Beverley P. Woolf
Esma Aïmeur
Roger Nkambou
Susanne Lajoie (Eds.)

LNCS 5091

Intelligent Tutoring Systems

9th International Conference, ITS 2008
Montreal, Canada, June 2008
Proceedings

Lecture Notes in Computer Science 5091

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Beverley P. Woolf Esma Aïmeur
Roger Nkambou Susanne Lajoie (Eds.)

Intelligent Tutoring Systems

9th International Conference, ITS 2008
Montreal, Canada, June 23–27, 2008
Proceedings

Springer

Volume Editors

Beverley P. Woolf
University of Massachusetts, Amherst, MA 01003-4610, USA
E-mail: bev@cs.umass.edu

Esma Aïmeur
Université de Montréal
C.P. 6128, Succ. Centre-Ville Montréal QC, H3C 3J7, Canada
E-mail: aimeur@iro.umontreal.ca

Roger Nkambou
University of Quebec at Montreal, Montreal, QC, H2X 3Y7, Canada
E-mail: nkambou.roger@uqam.ca

Susanne Lajoie
McGill University, Montreal, QC, H3A 1Y2, Canada
E-mail: susanne.lajoie@mcgill.ca

Library of Congress Control Number: 2008929517

CR Subject Classification (1998): K.3, I.2.6, H.5, J.1

LNCS Sublibrary: SL 2 – Programming and Software Engineering

ISSN 0302-9743
ISBN-10 3-540-69130-8 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-69130-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12282807 06/3180 5 4 3 2 1 0

Preface

The 9th International Conference on Intelligent Tutoring Systems (ITS 2008) was held June 23–27, 2008 in Montreal. This year we celebrated the 20th anniversary of the conference founded in 1988 in Montreal. We have had biennial conferences for most of the past 10 years around the world, including in Brazil, Taiwan, France, Canada, and the USA. These ITS conferences provide a forum for the interchange of ideas in all areas of computer science and human learning, a unique environment to exchange ideas and support new developments relevant for the future. The 2008 conference was a symbolic milestone that enabled us to look back at what has been achieved and what is currently being done, in order to face the challenges of tomorrow.

Much has changed in the last 20 years in terms of hardware, software, programmers, and education stakeholders. Technology is now networked, pervasive, and available anyplace and anytime. The potential exists to provide customized, ubiquitous guidance and instruction. However, much has remained the same and the need is just as great to model the learner, teaching strategies and domain knowledge. This year we saw an increase in research into student affect (motivation, boredom, and frustration), specifically attempts to detect student affect, while feedback studies considered which responses to provide given both student cognition and affect. Studies also looked at the impact on learning of positive feedback and politeness in feedback. New research was seen in data mining based on larger studies that use data from real students to diagnose effective learning and teaching. So much interest has been generated in this area that the first International Conference on Educational Data Mining was co-located with ITS 2008.

This year we received 207 submissions from six continents and accepted 63 full papers (30.4 %) and 61 short papers. Presented papers came from 20 countries, several of which have not been represented in previous ITS conferences. All accepted papers are published in this proceedings volume long papers are allotted ten pages and short papers three pages. We also present brief abstracts of the talks of our five invited speakers: Alan Collins, Julita Vassileva, Kurt VanLehn, Judy Kay, and Alan Lesgold. The conference also included seven workshops, interactive events, two tutorials, and a Young Researcher's Track.

The conference provided opportunities for the cross-fertilization of information and ideas from researchers working on interactive and adaptive learning environments for learners of all ages, for subject matter that spans the school curriculum (e.g., math, science, language learning), and for professional applications in industry, military, and medicine. Presented papers offered a rare professional opportunity for researchers to present cutting-edge research from a wide range of topics, including the fields of artificial intelligence, computer science, cognitive and learning sciences, psychology, and educational technology.

This year we instituted a meta-review process in which Senior Program Committee (PC) members managed three reviewers for each submitted paper and were able to engage in an e-mail discussion with reviewers for each paper. This resulted in more detailed reviews and enabled reviewers to consider and explore more deeply the reactions of other reviewers to each paper.

We thank the many, many people who helped make this conference possible. We especially thank our energetic PC, with over 100 members, including Senior PC, PC members and external reviewers who read numerous papers, managed other reviewers, and generally submitted their reviews on time. We thank the external reviewers who were recruited by PC members to assist us when we exhausted our initial cadre of reviewers. We thank the individual committees, including an Organizing, Program and Conference Committee along with the Chairs for Tutorial, Young Researchers, Demonstration, Poster, and Workshop activities. We are grateful to our longstanding International Steering Committee (14 members) who helped and guided us when decisions were needed. We are especially grateful to the General Chairs, Claude Frasson and Gilles Gauthier, who kept us on schedule and provided supportive advice. All these people are acknowledged in the next few pages and at http://gdac.dinfo.uqam.ca/its2008/.

Finally, we gratefully acknowledge Springer for its continuing support in publishing the proceedings of ITS 2008 and the generous support of our sponsors including University of Quebec at Montreal, McGill University, and the University of Montreal.

April 2008

Beverly Park Woolf
Esma Aïmeur
Roger Nkambou
Susanne Lajoie

Organization

Conference Committee

Conference Chairs	Susanne Lajoie (McGill University, Canada) Roger Nkambou (UQAM, Canada)
Program Chairs	Esma Aïmeur (University of Montreal, Canada) Beverly Park Woolf (University of Massachusetts, USA)
General Chairs	Claude Frasson (University of Montreal, Canada) Gilles Gauthier (UQAM, Canada)
Local Arrangements	Jacqueline Bourdeau (TELUQ, Canada)
Tutorials Chairs	Peter Brusilovsky (University of Pittsburg, USA) Valery Shute (ETS, USA)
Workshops Chairs	Roger Azevedo (University of Memphis, USA) Tak-Wai Chan (National Central University, Taiwan)
Posters Chairs	Cristina Conati (UBC, Canada) Judy Kay (University of Sydney, Australia)
Panels Chairs	Kenneth Koedinger (CMU, USA) Riichiro Mizoguchi (Osaka University, Japan)
Young Researchers	Guy Gouardères (University of Pau, France) Rosa Maria Vicari (Universidade Federal do Rio Grande do Sul, Brazil)
Demonstrations Chairs	Aude Dufresne (University of Montreal, Canada) Neil Heffernan (Worcester Polytechnic Institute, USA) André Mayers (University of Sherbrooke, Canada)
Sponsorship Chair	Daniel Dubois (UQAM, Canada)
Local Assistance	Line Marier, Marie-France Frasson, Nadia Amara, Mohamed Gaha, Sébastien Gambs, Philippe Fournier-Viger, Geneviève Gauthier, Usef Faghihi, Emmanuel Blanchard, Amal Zouaq, Valery Psyché, Hicham Hage

Senior Program Committee

Esma Aïmeur (University of Montreal, Canada)
Vincent Aleven (Carnegie Mellon University, USA)
Kevin Ashley (University of Pittsburgh, USA)
Ryan Shaun Joazeiro de Baker (Carnegie Mellon University, USA)
Ben du Boulay (University of Sussex, UK)
Bert Bredeweg (University of Amsterdam, The Netherlands)
Jacqueline Bourdeau (TELUQ-UQAM, Canada)
Paul Brna (Glasgow University, UK)
Peter Brusilovsky (University of Pittsburgh, USA)
Stefano Cerri (University of Montpellier, France)
Tak-Wai Chan (National Central University of Taiwan, ROC)
William Clancey (NASA/Ames Research Center, USA)
Cristina Conati (University of British Columbia, Canada)
Isabel Fernandez de Castro (University of the Basque Country, Spain)
Gerhard Fisher (University of Colorado, USA)
Claude Frasson (University of Montreal, Canada)
Gilles Gauthier (University of Quebec at Montreal, Canada)
Jim Greer (University of Saskatchewan, Canada)
Guy Gouardères (University of Pau, France)
Monique Grandbastien (Université de Nancy1, France)
Mitsuru Ikeda (Japan Advanced Inst. of Science and Technology, Japan)
Lewis Johnson (University of Southern California, USA)
Marc Kaltenbach (Bishop's University, Canada)
Judith Kay (University of Sydney, Australia)
James Lester (North Carolina University, USA)
Ken Koedinger (Carnegie Mellon University, USA)
Susanne Lajoie (McGill University, Canada)
Chee-Kit Looi (Information Technology Institute, Singapore)
Rosemary Luckin (University of Sussex, UK)
Gordon McCalla (University of Saskatchewan, Canada)
Tanja Mitrovic (University of Canterbury, New Zealand)
Riichiro Mizoguchi (University of Osaka, Japan)
Jack Mostow (Carnegie Mellon University, USA)
Roger Nkambou (University of Quebec at Montreal, Canada)
Toshio Okamoto (UEC, Japan)
Helen Pain (University of Edinburgh, UK)
Valerie Shute (Florida State University, USA)
Kurt Van Lehn (University of Pittsburgh, USA)
Julita Vassileva (University of Saskatchewan, Canada)
Felisa Verdejo (UNED, Spain)
Beverly Woolf (University of Massachusetts, USA)

Program Committee

Mohamed Abdelrazek (University of Cairo, Egypt)
Elizabeth Andre (University of Augsburg, Germany)
Roger Azevedo (University of Maryland, USA)
Joseph Beck (Worcester Polytechnic Institute, USA)
Gautam Biswas (Vanderbilt University, USA)
Ricardo Conejo (University of Malaga, Spain)
Cyrille Desmoulins (University of Grenoble, France)
Vladan Devedzic (University of Belgrade, Serbia)
Aude Dufresne (University of Montreal, Canada)
Ulrich Hoppe (University of Duisburg, Germany)
Paul Held (University of Erlangen-Nuremberg, Germany)
Neil Heffernan (Worcester Polytechnic Institute, USA)
Jean-Marc Labat (University of Paris 6, France)
Brent Martin (University of Canterbury, New Zealand)
Alessandro Micarelli (University of Rome, Italy)
Claus Moebus (University of Oldenburg, Germany)
Jean-François Nicaud (University of Nantes, France)
Khalid Rouane (University of Montreal, France)
Ana Paiva (University of Lisbon, Portugal)
Fabio Paraguaçu (University of Maceio, Brazil)
Jean-Pierre Pécuchet (INSA of Rouen, France)
Carolyn Rose (Carnegie Mellon University, USA)
Carole Redfield (St. Mary's University, USA)
Eileen Scanlon (Open University, UK)
Amy Soller (Institute for Defense, USA)
Akira Takeuchi (Kyushu Institute, Japan)
Pierre Tchounikine (University of Maine, France)
Gheorghe Tecuci (George Mason University, USA)
Wouter van Joolingen (University of Twente, The Netherlands)
Gerhard Weber (University of Freiburg, Germany)
Kalina Yacef (University of Sydney, Australia)

Steering Committee

Claude Frasson (University of Montreal, Canada) - Chair
Stefano Cerri (University of Montpellier II, France)
Isabel Fernandez-Castro (University of the Basque Country, Spain)
Gilles Gauthier (University of Quebec at Montreal, Canada)
Guy Gouardères (Université de Pau, France)
Mitsuru Ikeda (Japan Advanced Institute of Science and Technology, Japan)
Marc Kaltenbach (Bishop's University, Canada)
Judith Kay (University of Sidney, Australia)
Alan Lesgold (University of Pittsburgh, USA)
James Lester (North Carolina State University, USA)

Fabio Paraguaçu (Federal University of Alagoas, Brazil)
Elliot Soloway (University of Michigan, USA)
Daniel Suthers (University of Hawaii, USA)
Beverly Park Woolf (University of Massachusetts, USA)

Referees

Marie-Hélène Abel
Lisa Anthony
Nilufar Baghaei
Nicholas Balacheff
Scott Bateman
Emmanuel Blanchard
Denis Bouhineau
Christopher Brooks
Susan Bull
Hao Cen
Scotty Craig
Ben Daniel
Elisabeth Delozanne
Toby Dragon
Daniel Dubois
Sonia Faremo
Philippe Fournier-Viger
Sebastien Gambs
Abdelkader Gouaich
Nathalie Guin
Zinan Guo
Robert Hausmann
Hage Hicham
Tanner Jackson
Clement Jonquet
Panayiota Kendeou
Chad H. Lane
André Mayers
Ruddy Lelouche
Bernard Lefebvre
Vanda Luengo
Chas Murray
Tom Murray
Marie-H. Nienaltowski
Andrew Olney
Amy Ogan
Kaska Poraska-Pomsta
Michael Ringenberg
Ido Roll
Erin Walker
Amali Weerasinghe
Ruth Wylie
Diego Zapata-Rivera
Amal Zouaq

Sponsoring Institutions

ITS 2008 was organized by the Univeristy of Quebec at Montreal, McGill University and the University of Montreal in cooperation with ACM/SIGSCE, ACM/SIGAPP.fr, IEEE, IEEE Computer Society, AAAI, AIED Society, JSAI (Japanese Association for Artificial Intelligence), JSISE (Japanese Society for Information and Education), and CAAI (Chinese Association for Artificial Intelligence)

Table of Contents

Keynote Speaker Abstracts

Rethinking Education in the Age of Technology 1

Life-Long Learning, Learner Models and Augmented Cognition 3

Intelligent Training Systems: Lessons Learned from Building Before It Is Time 6

The Interaction Plateau: Answer-Based Tutoring < Step-Based Tutoring = Natural Tutoring 7

Social Learning Environments: New Challenges for AI in Education 8

Emotion and Affect

Self Versus Teacher Judgments of Learner Emotions During a Tutoring Session with AutoTutor 9

Towards Emotionally-Intelligent Pedagogical Agents 19

Viewing Student Affect and Learning through Classroom Observation and Physical Sensors 29

Comparing Learners' Affect While Using an Intelligent Tutoring System and a Simulation Problem Solving Game 40

What Are You Feeling? Investigating Student Affective States During Expert Human Tutoring Sessions 50

Responding to Student Uncertainty During Computer Tutoring: An Experimental Evaluation 60

Tutor Evaluation

How Does an Intelligent Learning Environment with Novel Design Affect the Students' Learning Results? 70

Learning Linked Lists: Experiments with the iList System 80

Re-evaluating LARGO in the Classroom: Are Diagrams Better Than Text for Teaching Argumentation Skills? 90

Automatic Multi-criteria Assessment of Open-Ended Questions: A Case Study in School Algebra 101

Why Tutored Problem Solving May be Better Than Example Study: Theoretical Implications from a Simulated-Student Study 111

A Case Study Empirical Comparison of Three Methods to Evaluate Tutorial Behaviors 122

Student Modeling

Children's Interactions with Inspectable and Negotiated Learner Models 132

Using Similarity Metrics for Matching Lifelong Learners 142

Developing a Computer-Supported Tutoring Interaction Component with Interaction Data Reuse 152

Machine Learning

Towards Collaborative Intelligent Tutors: Automated Recognition of Users' Strategies 162

Automatic Generation of Fine-Grained Representations of Learner Response Semantics 173

Automatic Construction of a Bug Library for Object-Oriented Novice Java Programmer Errors 184

Authoring Tools

Helping Teachers Build ITS with Domain Schema 194

Evaluating an Authoring Tool for Model-Tracing Intelligent Tutoring Systems 204

Open Community Authoring of Targeted Worked Example Problems ... 216

Agent Shell for the Development of Tutoring Systems for Expert Problem Solving Knowledge 228

Tutor Feedback and Intervention

Balancing Cognitive and Motivational Scaffolding in Tutorial Dialogue 239

Assessing the Impact of Positive Feedback in Constraint-Based Tutors 250

The Dynamics of Self-regulatory Processes within Self-and Externally Regulated Learning Episodes During Complex Science Learning with Hypermedia 260

The Politeness Effect in an Intelligent Foreign Language Tutoring System 270

Investigating the Relationship between Spatial Ability and Feedback Style in ITSs 281

Individualizing Tutoring with Learning Style Based Feedback 291

Use of Agent Prompts to Support Reflective Interaction in a Learning-by-Teaching Environment 302

A Standard Method of Developing User Interfaces for a Generic ITS Framework 312

Data Mining

Helping Teachers Handle the Flood of Data in Online Student Discussions 323

What's in a Cluster? Automatically Detecting Interesting Interactions in Student E-Discussions 333

Scaffolding On-Line Discussions with Past Discussions: An Analysis and Pilot Study of PedaBot 343

How Who Should Practice: Using Learning Decomposition to Evaluate the Efficacy of Different Types of Practice for Different Types of Students 353

How Does Students' Help-Seeking Behaviour Affect Learning? 363

Toward Automatic Hint Generation for Logic Proof Tutoring Using Historical Student Data 373

Does Help Help? Introducing the Bayesian Evaluation and Assessment Methodology 383

Using Knowledge Discovery Techniques to Support Tutoring in an Ill-Defined Domain 395

More Accurate Student Modeling through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing 406

E-Learning and Web-Based ITS

Interoperable Competencies Characterizing Learning Objects in Mathematics 416

Comparing Classroom Problem-Solving with No Feedback to Web-Based Homework Assistance 426

Harnessing Learner's Collective Intelligence: A Web2.0 Approach to E-Learning 438

Bridging the Gap between ITS and eLearning: Towards Learning Knowledge Objects 448

Natural Language Techniques and Dialogue

Semantic Cohesion and Learning 459

Dialogue Modes in Expert Tutoring 470

Seeing the Face and Observing the Actions: The Effects of Nonverbal Cues on Mediated Tutoring Dialogue 480

Affective Transitions in Narrative-Centered Learning Environments 490

Word Sense Disambiguation for Vocabulary Learning 500

Narrative Tutors and Games

Student Note-Taking in Narrative-Centered Learning Environments: Individual Differences and Learning Effects 510

Assessing Aptitude for Learning with a Serious Game for Foreign Language and Culture 520

Story-Based Learning: The Impact of Narrative on Learning Experiences and Outcomes 530

Semantic Web and Ontology

An Architecture for Combining Semantic Web Techniques with Intelligent Tutoring Systems 540

The Use of Ontologies to Structure and Support Interactions in LOR ... 551

Leveraging the Social Semantic Web in Intelligent Tutoring Systems 563

Structurization of Learning/Instructional Design Knowledge for Theory-Aware Authoring Systems 573

Expanding the Plausible Solution Space for Robustness in an Intelligent Tutoring System 583

Cognitive Models

Using Optimally Selected Drill Practice to Train Basic Facts 593

Eliminating the Gap between the High and Low Students through Meta-cognitive Strategy Instruction 603

Using Hidden Markov Models to Characterize Student Behaviors in Learning-by-Teaching Environments 614

To Tutor the Tutor: Adaptive Domain Support for Peer Tutoring 626

Collaboration

Shall We Explain? Augmenting Learning from Intelligent Tutoring Systems and Peer Collaboration 636

Theory-Driven Group Formation through Ontologies 646

Poster Papers

Self-assessment in Vocabulary Tutoring 656

Automatically Generating and Validating Reading-Check Questions 659

Dynamic Browsing of Audiovisual Lecture Recordings Based on Automated Speech Recognition 662

Agent-Based Framework for Affective Intelligent Tutoring Systems 665

Measuring the Perceived Difficulty of a Lecture Using Automatic Facial Expression Recognition 668

Minimal Feedback During Tutorial Dialogue 671

Can Students Edit Their Learner Model Appropriately? 674

When Is Assistance Helpful to Learning? Results in Combining Worked Examples and Intelligent Tutoring 677

Enabling Reputation-Based Trust in Privacy-Enhanced Learning Systems 681

Authoring Educational Games with Greenmind 684

An Experimental Use of Learning Environment for Problem-Posing as Sentence-Integration in Arithmetical Word Problems 687

Automatic Analyses of Cohesion and Coherence in Human Tutorial Dialogues During Hypermedia: A Comparison among Mental Model Jumpers 690

Interface Challenges for Mobile Tutoring Systems 693

Agora UCS Ubiquitous Collaborative Space 696

Adapte, a Tool for the Teacher to Personalize Activities 699

Framework for a Competency-Driven, Multi-viewpoint, and Evolving Learner Model 702

Use Chatbot CSIEC to Facilitate the Individual Learning in English Instruction: A Case Study 706

Using an Adaptive Collaboration Script to Promote Conceptual Chemistry Learning 709

Towards an Intelligent Emotional Detection in an E-Learning Environment 712

How Do We Get the Pieces to Talk? An Architecture to Support Interoperability between Educational Tools 715

Cognitive Load Estimation for Optimizing Learning within Intelligent Tutoring Systems 719

Investigating Learner Trust in Open Learner Models Using a 'Wizard of Oz' Approach 722

Personalized Learning Path Delivery: Models and Example of Application 725

Semi Automatic Generation of Didactic Resources from Existing Documents 728

An Evaluation of Intelligent Reading Tutors 731

An Intelligent Web-Based Learning System for Group Collaboration Using Contracts 734

An Adaptive and Customizable Feedback System for Intelligent Interactive Learning Systems 737

Detection of Learning Styles from Learner's Browsing Behavior During E-Learning Activities 740

Analyzing Learners' Self-organization in Terms of Co-construction, Co-operation and Co-ordination 743

Authoring Mobile Intelligent Tutoring Systems 746

XTutor: An Intelligent Tutor System for Science and Math Based on Excel 749

Tying Ontologies to Domain Contents for CSCL 752

One Exercise – Various Tutorial Strategies 755

Bi-directional Search for Bugs: A Tool for Accelerating Knowledge Acquisition for Equation-Based Tutoring Systems 758

Design of a System for Automated Generation of Problem Fields 763

Lessons Learned from Scaling Up a Web-Based Intelligent Tutoring System 766

Tailoring of Feedback in Web-Based Learning: The Role of Response Certitude in the Assessment 771

Trying to Reduce Bottom-Out Hinting: Will Telling Student How Many Hints They Have Left Help? 774

Leveraging C-Rater's Automated Scoring Capability for Providing Instructional Feedback for Short Constructed Responses 779

An Authoring Tool That Facilitates the Rapid Development of Dialogue Agents for Intelligent Tutoring Systems 784

Using an Emotional Intelligent Agent to Reduce Resistance to Change 787

Story Generation to Accelerate Math Problem Authoring for Practice and Assessment 790

Supporting the Guide on the SIDE 793

Comparing Two IRT Models for Conjunctive Skills 796

The Effect of Providing Error-Flagging Support During Testing 799

Cognitive Tutoring System with "Consciousness" 803

It's Not Easy Being Green: Supporting Collaborative "Green Design" Learning 807

Cognitive and Technical Artefacts for Supporting Reusing Learning Scenario Patterns 810

Integration of a Complex Learning Object in a Web-Based Interactive Learning System 813

Semantic Web Reasoning Tutoring Agent 816

An Affective Behavior Model for Intelligent Tutors 819

Decision Tree for Tracking Learner's Emotional State Predicted from His Electrical Brain Activity 822

Toward Supporting Collaborative Discussion in an Ill-Defined Domain 825

Author Index 829

Rethinking Education in the Age of Technology

Allan Collins

Learning Sciences, Northwestern University
Evanston, Illinois, 60208-0001
a-collins@northwestern.edu

All around us people are learning with the aid of new technologies: children are playing complex video games, workers are taking online courses to get an advanced degree, students are taking courses at commercial learning centers to prepare for tests, adults are consulting Wikipedia, etc. New technologies create learning opportunities that challenge traditional schools and colleges. These new learning niches enable people of all ages to pursue learning on their own terms. People around the world are taking their education out of school into homes, libraries, Internet cafes, and workplaces, where they can decide what they want to learn, when they want to learn, and how they want to learn.

The emergence of alternative venues for learning threatens the identification of learning with school. The tension between new forms of learning and old forms of schooling will not be resolved with the victory of one or the other. Rather, we see the seeds of a new education system forming in the rapid growth of new learning alternatives such as home schooling, learning centers, workplace learning, distance education, Internet cafes, educational television, computer-based learning environments, technical certification, and adult education. This does not mean that public schools are going to disappear, but their dominant role in education will diminish considerably.

The changes we see happening in education are neither all good nor all bad. We see many benefits to the kinds of education that technology affords, such as the ability of learners to pursue deeply topics of interest to them and to take responsibility for their own education. We also see many benefits in the successful history of traditional public schooling, which has provided extraordinary access to learning, status, and economic success for millions of students over the course of the past two centuries. But at the same time the roads to dystopia are also open. In particular, the new technologies can undermine both Thomas Jefferson's vision of educating citizens who can make sensible public policy decisions, and Horace Mann's vision of a society where everyone can succeed by obtaining a good education. Increasing the ability to personalize educational opportunities gives a natural advantage to those who can afford the services. Our fear is that citizenship and equity may be undermined by the fragmentation and customization afforded by the information revolution.

The developments described above are changing how people think about education. This rethinking will take many years to fully penetrate our understanding of the world and the society around us. Eventually when people and politicians become worried about what kids are learning or what adults don't know, their automatic reaction may not be "How can we improve the schools?" Instead they may ask, "How can we develop games to teach history?", "How can we make new technology resources

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 1–2, 2008.

available to more people?" or " What kinds of tools can support people to seek out information on their own?" These are all questions that push the envelop for improving education out of the schools and into new venues. The link between schooling and learning forces our conversation into institutional responses - we don't yet know how to ask wider questions when we think about improving education. To be successful, leaders will need to grasp these changes in a deep way and bring the government's resources to bear on the problems raised by the changes that are happening. They will have to build their vision of a new education system around these new understandings.

The rethinking that is necessary applies to many aspects of education and society. We are beginning to rethink the nature of learning, motivation, and what is important to learn. Further the nature of careers are changing and how people transition back and forth between learning and working. These changes demand a new kind of educational leadership and changing roles for government. New leaders will need to understand the affordances of the new technologies, and have a vision for education that will bring the new resources to everyone.

Life-Long Learning, Learner Models and Augmented Cognition

Judy Kay

CHAI: Computer human adapted interaction research group
School of Information Technologies
The University of Sydney, Australia
judy@it.usyd.edu.au

Abstract. Our field of Intelligent Tutoring Systems has long been inspired by the vision of achieving huge improvements in learning via expert personalised teaching. As we now see computers become ubiquitous and pervasive, we can broaden that vision to include new ways to learn what we need to know, when we need to know it, throughout our lives. In this 20th anniversary of the ITS conferences, we can see that the future will bring an ITS vision that is broadened to include augmented cognition, where systems provide, not only teaching, but also the means to augment our memory by facilitating access to information as needed, be that as mediated contact with other people or access to our own external memory, a collection of the things we want to be able to re-find or remember as needed.

Central to this vision is the life-long learner model because it bears the responsibility for modelling relevant aspects of the learner so that an ITS can help us access the information we need to meet our needs. This talk draws on the foundations of ITS work to create a view of the nature of that life-long learner model, the processes of life-long learner modelling and the ways that an ITS can make use of these. The talk illustrates the vision in terms of representations of learner models, user interface and other practical concerns such as privacy.

1 ITS as a Grand Challenge Problem

The ITS research community has been driven by the importance of the human need to learn and to access information. We now have a long track record of work towards understanding how to push the limits of technology in support of improved learning. This draws on both improved understanding of learning and human cognition and equally, on creating new ways to build software systems that are effective aids for learning.

More recently, there has been clear recognition of the importance of our vision and goals as well as the challenges in achieving them. In 2002, the Computing Research Association (CRA) identified five Grand Research Challenges in Computer Science and Engineering[1]. One of these,

[1] http://www.cra.org/grand.challenges/

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 3–5, 2008.

, matches the ITS goal of personalised teaching. Subsequently, the United Kingdom Computing Research Committee (UKCRC) identified nine Current Grand Challenges for Computing[2]. One of these GC8, , recognises the importance of the multidisciplinary research that is already a strong part of the ITS tradition. Another, GC3, is also closely aligned ITS research. In the last year, another peak body, the National Academy of Engineering identified 14 wide-ranging grand challenge problems. One of these is [3], which recognises the importance of research into technology to support instruction that "can be individualized based on learning styles, speeds, and interests to make learning more reliable". This, too, is directly aligned with the goals of the ITS community. This talk will explore two key aspects that are at the core of a research agenda that tackles these grand challenge research problems.

2 Life Long Learner Models

Learner models are at the heart of the personalisation of ITSs [1]. For life-long learning, we need to explore ways to build life-long learner models. These have the potential to track learning progress over long periods and across the range of sources of evidence about the learner's progress.

This talk will explore some of the issues that arise as we move towards such models. Some of these have already had considerable attention within our community. Notably, there has been wide recognition of the importance of interoperability, where there can be effective communication between ITSs. The talk will review approaches based on semantics and standardisation efforts and how these appear to provide some potential foundations for ensuring that a meaningful long term learner model can draw upon information that is harvested by the range of learning systems, as well as other software, that a person may use throughout their life. The talk will examine ways we will be able to make use of both conventional learning tools and environments, such as learner management systems (LMSs) as well as ITSs, with their especially rich information about the learners. The talk will explore alternative lines of research that can enable us to exploit the vast quantities of electronic traces of learner activity within conventional software. Taking the example of an LMS, we can, on the other hand, explore the challenges on enhancing it with learner models. Alternatively, we can make post-hoc interpretations of the vast data available from such tools, the electronic traces that learners leave through their interaction. These have huge potential to provide invaluable evidence for a rich life-long learner model. Another key is the human-in-the-loop approaches, particularly open, transparent and scrutable learner models. Our research agenda for life-long learner models must also make meaningful progress on privacy management for these models.

[2] http://www.ukcrc.org.uk/grand_challenges/current/index.cfm
[3] http://www.engineeringchallenges.org/cms/8996/9127.aspx

3 Life-Long Augmented Memories

The ubiquity and pervasive nature of computers has the potential to have important impact on the our learning needs and goals because we may be able to rely on technology to augment our memories. This talk will explore key directions for research which takes account of this ubiquitous nature of computing: approaches to just-in-time learning, delegation of [illegible] to the computer and ways that electronically mediated collaboration can support remembering by indirection, aided by other people. Our focus will be on the links between such augmented cognition and life-long learner models.

Reference

1. Self, J.: The defining characteristics of intelligent tutoring systems research: ITSs care, precisely. International Journal of Artificial Intelligence in Education 10(3-4), 350–364 (1999)

Intelligent Training Systems: Lessons Learned from Building Before It Is Time

Alan M. Lesgold

School of Education, University of Pittsburgh, Pittsburgh, PA 15217 USA
AL@pitt.edu

I discuss what I learned while developing five generations of intelligent coached apprenticeship systems somewhat earlier than hardware, basic software support, programmers, or users were ready for them. First, it was essential to remain focused on the central instructional principles driving our work. Second, we learned that the social issues in deploying novel systems trump any demonstrations of return on investment or efficacy. People only use what they are comfortable using. Third, we learned that being as free as possible of specific operating system or software commitments was absolutely necessary. Fourth, we learned that the fundamental role of coached apprenticeship systems is to efficiently provide the rare moments from real life that afford the chance to learn deep and transferable skills and knowledge. Fifth, we learned that developing intelligent coached environments affords opportunities for learning by teachers/trainers and designers of work processes as well as by students/trainees. Finally, we learned that capabilities for which we can have complete student models are exactly those destined to be taken over by machines, placing a premium on far transfer as the goal for high-end training/teaching systems.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, p. 6, 2008.

The Interaction Plateau: Answer-Based Tutoring < Step-Based Tutoring = Natural Tutoring

Kurt VanLehn

LRDC, University of Pittsburgh, Pittsburgh, PA 15260 USA
VanLehn@cs.pitt.edu

Face-to-face tutoring by an expert human tutor is widely thought to be more effective than intelligent tutoring systems (ITS), which are in turn thought to be more effective than computer-aided instruction (CAI), computer-based training (CBT), etc. The latter tutoring systems have students work out complex solutions on paper, then enter their answer into the tutor, which gives them feedback and hints on their answer. Thus, CAI, CBT, etc. are answer-based tutoring systems. This is a low level of interactivity, in that the student may make many inferences between the time they start the problem and when they first get feedback on their thinking. With a typical ITS, such as the Andes physics tutoring system, students enter every step of a complex solution into the tutor, which gives them feedback and hints, either immediately or when they have finished entering all the steps. These systems are step-based tutoring systems, because the feedback and hints are directed at steps rather than the final answer. They are moderately interactive, because students make a moderate number of inferences per step. When interacting face-to-face with a human tutor, students often talk aloud as they reason, and thus allow the tutor to hear and intervene at almost every inference made by the student. Thus, human tutoring is highly interactive. Natural language tutoring systems, such as Why2-Atlas and Cordillera, are engineered to act like human tutors, so they too are highly interactive. If we use "natural tutoring" to cover both human tutoring and natural language tutoring, then the three types of tutoring can be ordered:

answer-based tutoring < step-based tutoring < natural tutoring

This certainly holds for their degree of interactivity, as just argued. This is also thought to be the ordering for their learning effectiveness. Moreover, it is sometimes thought that higher interactivity affords or perhaps even causes higher learning gains.

This talk will debunk that myth. In particular, experiments with human and computer tutors usually find that learning gains are ordered this way:

answer-based tutoring < step-based tutoring =natural tutoring

Increasing interactivity beyond the step level appears to neither afford nor cause higher learning gains.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, p. 7, 2008.

Social Learning Environments: New Challenges for AI in Education

Julita Vassileva

Computer Science Department, University of Saskatchewan,
176 Thorvaldson Bldg., 110 Science Place, Saskatoon, SK, S7N 5A9 Canada
jiv@cs.usask.ca

Abstract. The advance of new social computing technologies (called often Web 2.0) brings new opportunities and challenges for eLearning. They allow us to leverage an endless amount of learning resources, repositories and people – learners, tutors, and teachers. By wise selection and combination of these resources, the "holy grail" of AI and Education can potentially be achieved – a personalized, adaptive learning environment. Yet there are many challenges along the way. To combine functionality offered by various applications, protocols are required (e.g. SOAP) and smooth interface integration (e.g. mash-ups). The resources are distributed and decentralized, created by different authors and organizations and following different semantic and annotation agreements. Imposing hard standards is not going to work, if we want to tap into a wide pool of user-contributed resources, which is the key feature of Web 2.0. Combining these resources requires shared meaning, even if just on a limited scale and time, for the purpose at hand. Community-based semantic agreements (ontologies) that are constantly evolving are one way to deal with this problem. User data is collected by many applications that create their own user models. Sharing this data brings many advantages for personalization, but also creates risks related to privacy. Mechanisms for combining user data and taking action need to be developed. Trust and reputation mechanisms and decentralized user modeling address this problem. Finding appropriate data and applications/services for a given learner at a given time is a big issue. Collaborative filtering is a well-established, relatively light-weight technique in areas that do not require interpreting complex user input. However, learning applications require complex user input. Complex models of learner knowledge need to be correlated, and the cold-start / sparse data problem is a serious hurdle. Finally, the most critical problem from my point of view is motivating stakeholders (authors, teachers, tutors, learners) to participate. Without their participation, the pool of resources and learners (peers and collaborators) to interact with will never reach the level of diversity necessary to ensure personalized, adaptive learning environments for a large number of learners. Designing incentive mechanisms for participation can be viewed as a kind of instructional planning, which can be successful in achieving certain levels and quality of participation. The talk provides an overview of these issues and research that addresses them.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, p. 8, 2008.

Self Versus Teacher Judgments of Learner Emotions During a Tutoring Session with AutoTutor

Sidney D'Mello[1], Roger Taylor[2], Kelly Davidson[1], and Art Graesser[1]

[1] Institute for Intelligent Systems, University of Memphis, Memphis, TN 38152 USA
{sdmello,kldavdsn,a-graesser}@memphis.edu
[2] Department of Teaching and Learning, Peabody College, Vanberbilt University
roger.s.taylor@vanderbilt.edu

Abstract. The relationship between emotions and learning was investigated by tracking the affective states that college students experienced while interacting with AutoTutor, an intelligent tutoring system with conversational dialogue. An emotionally responsive tutor would presumably facilitate learning, but this would only occur if learner emotions can be accurately identified. After a learning session with AutoTutor, the affective states of the learner were classified by the learner and two accomplished teachers. The classification of the teachers was not very reliable and did not match the learners self reports. This result suggests that accomplished teachers may be limited in detecting the affective states of learners. This paper discusses the implications of our findings for theories of expert tutoring and for alternate methodologies for establishing convergent validity of affect measurement.

1 Introduction

Researchers in the ITS community have always considered it important to develop a model of the learner. The model parameters can come from different sources, such as static trait measures that are extracted form learner self reports and dynamic measures that are induced from the stream of behaviors and thoughts of the learner during the course of learning. ITSs are expected to adapt their tutoring strategies to the learners' aptitude, personality, prior-knowledge, goals, progress, and a host of other parameters that presumably impact learning. It is also widely acknowledged that the scope of learner modeling need not be restricted to cognitive factors alone, because the affective states (emotions) of learners are inextricably bound to the cognitive states and ultimately linked to learning gains [1-4]. A person's affective response to an ITS can change, depending on their goals, preferences, expectations and knowledge state. For example, academic risk theory contrasts adventuresome learners who want to be challenged with difficult tasks, take risks of failure, and manage negative emotions when they occur, whereas cautious learners want to tackle easier tasks, take fewer risks, and minimize failure and the resulting negative emotions [5].

We know that events that arise during a tutoring session with an ITS cause learners to experience a variety of possible emotions that depend on the learning challenges, the amount of changes they experience, and whether important goals are blocked. Negative emotions such as confusion and frustration occur when learners confront

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 9–18, 2008.

contradictions, anomalous events, obstacles to goals, salient contrasts, perturbations, surprises, equivalent alternatives, and other stimuli or experiences that fail to match expectations [6-7]. Positive emotions (such as engagement, flow, delight, excitement and eureka) are experienced when tasks are completed, challenges are conquered, insights are unveiled, and major discoveries are made.

There is some evidence that there are significant relationships between affective states and learning gains. Kim [8] conducted a study which demonstrated that the interest and self-efficacy of a learner significantly increased when the learner was accompanied by a pedagogical agent that served as a virtual learning companion that was sensitive to the learner's affect. Linnenbrink and Pintrich [9] reported that the posttest scores of physics understanding decreased as a function of negative affect during learning. Graesser and colleagues have demonstrated that the affective state of confusion, where learners' are in a state of cognitive disequilibrium, with more heightened physiological arousal and with more intense thought, is positively correlated with learning [1], [3]. Of course, it is important to differentiate the state of being productively confused, which leads to learning and positive emotions, from being hopelessly confused, which has no pedagogical value. The affective state of flow, where the learner is so absorbed in the material that time and fatigue disappear [10], is positively correlated with learning, whereas prolonged experiences of boredom seem to negatively impact learning gains [1].

An affect-sensitive tutor would presumably enhance intelligent learning environments [3], [11-13]. Such an ITS would incorporate assessments of the students' cognitive, affective, and motivational states into its pedagogical strategies to keep students engaged, boost self-confidence, heighten interest, and presumably maximize learning. For example, if the learner is frustrated, the tutor would need to generate hints to advance the learner in constructing knowledge, and make supportive empathetic comments to enhance motivation. If the learner is bored, the tutor would need to present more engaging or challenging problems for the learner to work on. We are currently in the process of developing a version of AutoTutor that is sensitive to both the cognitive and affective states of learners [11], [6]. AutoTutor is an intelligent tutoring system that helps learners construct explanations by interacting with them in natural language and helping them use simulation environments [3].

At this point in science, we need to answer several questions about the role of emotions in deep learning before we can build a functional affect-sensitive ITS. One important question needs to be addressed by all theoretical frameworks and pedagogical practices that relate emotions and learning: How are affective states detected and classified?.

A first step is to explore a simple measurement question: How reliably can emotions be classified by humans and machines. An emotionally sensitive learning environment, whether it be human or computer, requires some degree of accuracy in classifying the learners' affect states. The emotion classifier need not be perfect, but it must have some degree of accuracy.

We have previously conducted a study that investigated the reliability by which emotions can be classified by the learners themselves versus peers and versus trained judges [14]. Our results supported a number of conclusions about emotion measurement by humans. First, the interrater reliability between the various pairs of judges

(self-peer, self-trained judge 1, self-trained judge2, peer-trained judge 1, peer-trained judge 2, trained judge1– trained judge 2) was quite low, with an average kappa of 0.18. Second, trained judges who are experienced in coding facial actions and tutorial dialogue provided affective judgments that were more reliable ($\kappa = .36$) and that matched the learners' self reports better than the judgments of untrained peers.

The overall low kappa scores between the various judges highlight the difficulty in measuring a complex construct such as emotion. It is illuminating to point out, however, that the kappas for the two trained judges in the Graesser et al [14] study are on par with data reported by other researchers who have assessed the reliability of emotion detection by [15-18]. Statisticians have sometimes claimed that kappa scores ranging from 0.4 – 0.6 are typically considered to be fair, 0.6 – 0.75 are good, and scores greater than 0.75 are excellent [19]. Based on this categorization, the kappa scores obtained in these studies would range from poor to fair. However, such claims of statisticians address the reliability of multiple judges or sensors when the phenomenon is more salient and when the researcher can assert that the decisions are clear-cut and decidable. The present research goal on emotions is very different. Our goal is to use the kappa score as an unbiased metric of the reliability of making affect decisions, knowing full well that such judgments are fuzzy, ill-defined, and possibly indeterminate.

Critics might attribute the low kappa scores achieved in previous studies to various inadequacies of our methodology. Predominant among these concerns is the lack of knowledge about emotions that people have in general, irrespective of whether the affect judges are the participants, their peers, the trained judges, and other researchers conducting field observations on affect. Perhaps people with heightened emotional expertise (i.e., knowledge, intelligence), such as social workers or FBI agents, would provide more accurate models of learners' emotions.

In this paper, we directly investigated the above criticism by measuring the degree to which people with presumably heightened emotion-detection expertise match the judgments of the learner. In particular, we assessed the reliability by which middle and high school teachers judged the emotions of the learner. The notion of teachers having heightened emotion-detection expertise emerges from diverse investigations of accomplished teachers and expert tutors. For example, Goleman [2] stated in his book, *Emotional Intelligence*, that expert teachers are able to recognize a student's emotional state and respond in an appropriate manner that has a positive impact on the learning process. Lepper and Woolverton [13] have claimed that it takes expertise in tutoring before accurate detection of learner emotions can be achieved. This requirement of expertise is apparently quite important because, according to Lepper and Woolverton [13], roughly half of expert tutors' interactions with the student are focused on affective elements. These important claims would be seriously limited if teachers are unable to detect the affective states of the learner. This question motivated the present study.

The present study tracked the affective states that college students experience while interacting with AutoTutor. We investigated the extent to which teachers can accurately identify the affective states of learners who interact with AutoTutor. This immediate objective feeds into the long-term goal of building a version of AutoTutor

that identifies and responds adaptively to the affective states of the learner. AutoTutor will never be able to adapt to the learner's emotions if it cannot detect the learner's emotions. Peer tutors and expert tutors similarly will be unable to adapt to the learner's emotions if they cannot identify such affective states.

2 Methods

The participants were 28 undergraduates at the University of Memphis who participated for extra course credit. After completing a pretest, participants interacted with AutoTutor for 32 minutes on one of three randomly assigned topics in computer literacy: hardware, Internet, or operating systems (see [3] for detailed information about AutoTutor). Two videos were recorded during the participant's interaction with AutoTutor. A video of the participant's face was recorded with a camera and a screen-capturing software program called Camtasia Studio was used to capture the audio and video of the participant's entire tutoring session.

Figure 1 depicts the experimental setup for the study. The participant interacted with the AutoTutor program on the center monitor, while the left and right monitors captured the participants body movements and face respectfully. During the interaction phase, the left and right monitors were turned off.

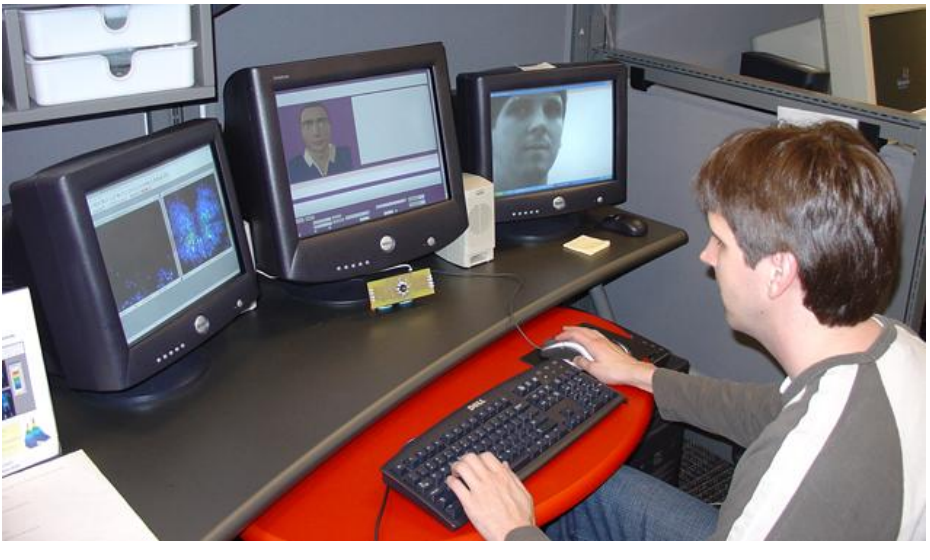


Fig. 1. Learner interacting with AutoTutor

After the tutorial session, participants completed a posttest on the learning of computer literacy (which is irrelevant data from the standpoint of the present study). Participants subsequently participated in a retrospective emotion judgment procedure. The videos of the participants' face and screen were synchronized and displayed to the participants (see middle and right monitors in Figure 1). The participants were

instructed to make judgments on what affective states were present at 20-second intervals; at each of these points, the video automatically paused (freeze-framed). Participants could also pause the videos at any time in between these 20-second points and make affective judgments at those points.

A list of the affective states and definitions was provided to the participants. The states were boredom, confusion, flow, frustration, delight, neutral and surprise, the emotions that were most frequently experienced during previous research with AutoTutor [1], [20].

In addition to the *self judgments* that were provided by the participants, two middle school teachers judged all of the sessions individually. The teachers were accomplished Master teachers in Memphis middle and high schools who were recognized for their accomplishments in motivating students and promoting student learning. Since affect judgment is a time consuming procedure, both teachers judged either the first half or the second half of each participants session. Specifically, for 14 randomly assigned participants, both teachers made affective judgments on the first half of the participants' AutoTutor session. Both teachers judged the second half of the remaining 14 sessions.

3 Results and Discussion

Interjudge reliability in judging emotions was computed using Cohen's kappa for the three possible pairs of judges (self vs. teacher1, self vs. teacher2, and teacher1 vs. teacher2). The observations included those judgments at the 20-second interval polling (N = 1459) and those in-between observations in which at least one judge observed an emotion in between two successive pollings (N = 329). Cohen's kappa scores were computed separately for each of the 28 learners.

We performed a repeated measures ANOVA, with the three judge pairs as within subject factors, and the order (first half vs. second half of participants session) as a between subject factor. There were statistically significant differences in kappa scores among the three judges, $F(2, 52) = 6.783$, $MSe = .01$, $p < .01$, partial $\eta^2 = .207$. Bonferroni post-hoc tests indicated that there were no significant differences in the kappa scores between the self and the teachers ($\kappa_{\text{self-teacher1}} = .076$, $\kappa_{\text{self-teacher2}} = .027$). However, kappa score between the self and teacher2 was significantly lower than the kappa between the two teachers ($\kappa_{\text{teacher1-teacher2}} = .123$). Furthermore, the interaction between judge pair and order was not significant $F(2, 52) < 1$, $p = .859$, indicating that kappa scores were the same irrespective of whether the judgments were made on the first or the second half of the learners' AutoTutor session.

These results support the conclusion that teachers are not particularly good at judging the learners emotions. Judgments provided by the two teachers were not very reliable (i.e. the teachers did not agree with each other) and did not match the learners' self reports. Before we accepted this conclusion too cavalierly, we examined whether the different judge types (self vs. teachers) are sensitive to a different set of emotions. We answered this question by examining the proportion of emotions reported by each judge. Table 1 presents means and standard deviations for the proportion scores that were computed individually for each of the 28 learners and 3 judges.

Table 1. Proportion of emotions observed by self and teachers

Emotion	Self		Teacher1		Teacher2		Mean Judges	
	Mean	Stdev	Mean	Stdev	Mean	Stdev	Mean	Stdev
Boredom	0.155	0.137	0.044	0.078	0.074	0.072	0.091	0.057
Confusion	0.186	0.149	0.065	0.051	0.188	0.119	0.146	0.070
Delight	0.031	0.048	0.019	0.027	0.009	0.021	0.020	0.011
Flow	0.192	0.173	0.634	0.107	0.575	0.187	0.467	0.240
Frustration	0.130	0.122	0.145	0.097	0.032	0.044	0.102	0.061
Neutral	0.284	0.248	0.074	0.060	0.108	0.107	0.155	0.113
Surprise	0.022	0.029	0.018	0.027	0.014	0.026	0.018	0.004

We performed a 3×7×2 factor repeated measures ANOVA on the proportions of emotions observed by the three judges. The two within subject factors were the affect judge with 3 levels (self, teacher1, and teacher2) and the emotion with 7 levels (boredom, confusion, delight, flow, frustration, neutral, surprise). The order (first half vs. second half of participants session) was included as a between subject factor. The proportion scores are constrained to add to 1.0 within order and judge, so it is not meaningful to consider the main effects of order and judge. However, the main effect of emotion and the remaining interactions are not constrained and therefore justifiable.

The main effect for emotion was statistically significant, $F(6, 156) = 118.455$, $MSe = .017$, $p < .001$, $\eta^2 = .820$, as was also the interactions between emotion × order, $F(6, 156) = 2.627$, $MSe = .017$, $p < .05$, partial $\eta^2 = .092$, judge × emotion, $F(12, 312) = 32.204$, $MSe = .012$, $p < .001$, partial $\eta^2 = .553$, and the three way interaction of judge × emotion × order, $F(12, 312) = 1.997$, $MSe = .012$, $p < .05$, $\eta^2 = .071$. Quite clearly, most of the variance is explained by the main effect of differences in emotions and by the judge × emotion interaction. Therefore, we performed follow up analyses of simple main effects between three judges within the seven emotions.

Bonferroni post-hoc tests indicated that the proportions of boredom, neutral, and frustration reported by the two teachers were statistically similar and quantitatively lower than the self judgments. Therefore, it appears that teachers have difficulty in detecting states such a boredom and neutral that are accompanied by a generally expressionless face that is devoid of diagnostic facial cues [21]. But what about frustration? This is arguably a state that is expressed through significant bodily arousal and animated facial expressions. We suspect that the difficulty experienced by the teachers in detecting frustration might be explained by the social display rules that people adhere to in expressing affect [22]. Social pressures may result in the learner disguising of negative emotions such as frustration, thus making it difficult for the teachers to detect this emotion.

It appears that the affective state of flow was detected at higher proportions by the two teachers than by the self. However, if self reports of affect are considered to be the ground-truth measure of affect, a majority of the instances of flow that were observed by the teachers would be considered to be false positives. It appears, that in the absence of sufficient facial and contextual cues, the teachers attribute the learners' emotions to the flow experience. This is clearly attributing too much to the learner.

Confusion, an emotion that is fundamental to deep learning [1], [3] has a facial imprint of a lowered brow and tightened eyelids [21]. This was detected at similar rates

by the self and teacher1, but at lower rates than teacher2. This pattern is plausible because judgments provided by teacher1 matched judgments provided by the participants at a somewhat higher rate, although not statistically significant, than teacher2.

Finally, delight and surprise were detected at similar rates by the self and the two teachers. Experiences of delight and surprise are rare, however (2% each when averaged across all judges), and are typically accompanied by highly animated facial activity [21]. Such salient constraints would explain why they were detected at similar rates by all the judges.

The low kappa scores between the self and the two teachers, coupled with the differences in the proportion of emotions experienced by the self and the teachers, suggest that the teachers tend to judge self classified experiences of boredom, confusion, frustration, and neutral as similar to the state of flow. This was verified by conducting a follow-up analyses that focused on isolating the source of errors in the teachers' judgments. Two confusion matrices were computed, each contrasting the self judgments with judgments by teacher1 and teacher 2. Table 2, presents an average of the two matrices.

An analysis on Table 2 revealed two clear sources of discrepancies between the self judgments and the judgments provided by the two teachers. First, the teachers appear to annotate several of the emotions as being in the state of flow or heightened engagement. For example, the teachers classified 41% of self diagnosed experiences of boredom as flow. This miscategorization is heightened for neutral, with 61% self reported neutral instances being classified as flow. The second source of classification errors occurs at instances where the teacher fails to make an emotion judgment, but the self provides a rating (see the None column). This occurs during instances when the learner makes a voluntary affect judgment, in between the 20 second stops, and the teachers fail to detect those points.

Table 2. Confusion matrix contrasting self judgments with average of teachers' judgments

Self	**Teachers Judgments**							
Judgments	Boredom	Confusion	Delight	Flow	Frustration	Neutral	Surprise	None
Boredom	0.13	0.10	0.00	0.41	0.08	0.07	0.01	0.25
Confusion	0.05	0.14	0.01	0.40	0.05	0.06	0.02	0.31
Delight	0.02	0.06	0.03	0.38	0.05	0.01	0.02	0.42
Flow	0.05	0.13	0.01	0.52	0.04	0.09	0.00	0.10
Frustration	0.03	0.08	0.01	0.46	0.05	0.05	0.01	0.28
Neutral	0.05	0.09	0.01	0.61	0.05	0.09	0.01	0.10
Surprise	0.02	0.04	0.05	0.19	0.05	0.05	0.00	0.56
None	0.02	0.03	0.01	0.07	0.04	0.01	0.01	0.83

4 General Discussion

An emotionally sensitive tutor, whether human or artificial, would presumably promote learning gains, engagement, and self-efficacy in the learner. Such a tutor should have different strategies and dialogue moves when the learner is confused or frustrated than when the learner is bored. However, both human and automated tutors can

be emotionally adaptive only if the emotions of the learner can be detected. The accuracy of the detection need not be perfect, but it should be approximately on target.

We have previously documented that trained judges who are experienced in coding facial actions and tutorial dialogue provide affective judgments that are more reliable and that match the learner's self reports better than the judgments of untrained peers. [14]. The results of this study support a number of additional conclusions about emotion detection by humans. It appears that accomplished teachers do not seem to be very adept at detecting the learners' emotions. Emotion judgments provided by the two teachers were not very reliable, i.e. the teachers did not agree with each other, and their judgments showed very little correspondence to the learner's self reports. In fact the degree to which the teachers affective judgment matched the self reports of the learner were on par with peer judges and were quantitatively lower than the trained judges. So untrained peers and accomplished teachers do not seem to be very proficient at judging the emotions of the learner.

It is possible that the assessments of learner affect provided by peers and teachers would be more accurate in naturalistic settings such as tutoring sessions or classrooms, where the judgments would occur in real time and the peers and teachers would have established a rapport with the students and have vested interests in their learning. These conditions are difficult to recreate in a laboratory, as it would be difficult to envision a scenario where the learner, a peer, trained judges, and teachers could simultaneously provide online emotion judgments. Nevertheless, our results suggest that, when presented with the identical stimulus (videos of the participants face and screen), judgments by the self and trained judges were more reliable than judgments by the peers and teachers.

It appears that each type of affect judge, be it the self, the untrained peer, the trained judges, or the accomplished teachers, bring a unique set of perspectives, standards, and experience to the affect judgment task. For example, it is reasonable to presume that participants tap into episodic memories of the interaction in addition to the prerecorded facial cues and contextual features when they retrospectively judge their own emotions (self judgments).

Unlike the self, the trained judges are not mindful of the episodic memory traces of the participants. However, they have been extensively trained on detecting subtle facial expressions with the Facial Action Coding System [22], and are more mindful of relevant facial features and transient facial movements. They also have considerable experience interacting with AutoTutor. Our results suggest that training on facial expressions (diagnostic assessment) coupled with knowledge on AutoTutor dialogue (predictive assessment), makes the trained judges robust affect detectors. The trained judges exhibit reliability (they agree with each other) as well as convergent validity (their judgments match self reports). Therefore, from a methodological perspective, retrospective affect judgments by the participant combined with offline ratings by trained judges, seems to be valuable protocol to establishing construct validity in emotion measurement, at least when compared to untrained observers, peers, and even teachers.

Affect sensitivity is an important requirement for ITSs that aspire to bridge the communicative gap between the highly expressive human and the socially challenged computer. Therefore, integrating sensing devices and automated affect classifiers is an important challenge for next generation ITSs that are attempting to broaden the

bandwidth of adaptivity to include the learners' cognitive, affective, and motivational states. Although, a handful of automated affect detection systems operate in an unsupervised fashion, supervised machine learning techniques are at the heart of most of the current affect detection systems. Consequently, providing accurate models of ground-truth for a complex construct such as emotion is an important requirement for such supervised affect classifiers. We hope to have scaffolded the development of automated affect-detection systems by providing a methodology to annotate the emotions of a learner in an ecologically valid setting (randomly selected participants rather than actors and the emotional expressions occurred naturally instead of being induced), and contrasting our methodology of self plus trained judgments with alternatives (peers, teachers, observers [1], and emote-aloud protocols [20]. We are currently developing such an emotion classifier with an eye for integrating it into an affect-sensitive version of AutoTutor. Whether an automated affect-sensitive AutoTutor has a positive impact on learning awaits future research and technological development.

Acknowledgements

Special thanks to Gloria Williams and Arlisha Darby, the two teachers who provided the affect judgments. We also acknowledge O'meed Entezari, Amy Witherspoon, Bethany McDaniel, and Jeremiah Sullins for their help with the data collection. This research was supported by the National Science Foundation (REC 0106965 and ITR 0325428) and the Institute of Education Sciences (R305B070349) Any opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF or IES.

References

1. Craig, S.D., Graesser, A.C., Sullins, J., Gholson, B.: Affect and learning: An exploratory look into the role of affect in learning. Journal of Educational Media 29, 241–250 (2004)
2. Goleman, D.: Emotional intelligence. Bantam Books, New York (1995)
3. Graesser, A.C., Chipman, P., King, B., McDaniel, B., D'Mello, S.: Emotions and Learning with AutoTutor. In: Luckin, R., et al. (eds.) 13th International Conference on Artificial Intelligence in Education (AIED 2007), pp. 569–571. IOS Press (2007)
4. Snow, R., Corno, L., Jackson, D.: Individual differences in affective and cognitive functions. In: Berliner, D.C., Calfee, R.C. (eds.) Handbook of educational psychology, pp. 243–310. Macmillan, New York (1996)
5. Meyer, D.K., Turner, J.C.: Reconceptualizing emotion and motivation to learn in classroom contexts. Educational Psychology Review 18, 377–390 (2006)
6. Graesser, A.C., Jackson, G.T., McDaniel, B.: AutoTutor holds conversations with learners that are responsive to their cognitive and emotional states. Educational Technology 47, 19–22 (2007)
7. Mandler, G.: Mind and emotion. Wiley, New York (1976)
8. Kim, Y.: Empathetic Virtual Peers Enhanced Learner Interest and Self-Efficacy. In: Workshop on Motivation and Affect in Educational Software at the 12th International Conference on Artificial Intelligentce in Education, Amsterdam, The Netherlands (2005)

9. Linnenbrink, E.A., Pintrich, P.R.: The role of motivational beliefs in conceptual change. In: Limon, M., Mason, L. (eds.) Reconsidering conceptual change: Issues in theory and practice, pp. 115–135. Kluwer Academic Publishers, Dordretch (2002)
10. Csikszentmihalyi, M.: Flow: The Psychology of Optimal Experience. Harper-Row, New York (1990)
11. D'Mello, S.K., Picard, R., Graesser, A.C.: Towards an Affect Sensitive AutoTutor. IEEE Intelligent Systems 22(4), 53–61 (2007)
12. Lepper., M.R., Chabay, R.W.: Socializing the intelligent tutor: Bringing empathy to computer tutors. In: Mandl, H., Lesgold, A. (eds.) Learning Issues for Intelligent Tutoring Systems, pp. 242–257. Erlbaum, Hillsdale (1988)
13. Lepper, M.R., Woolverton, M.: The wisdom of practice: Lessons learned from the study of highly effective tutors. In: Aronson, J. (ed.) Improving academic achievement: Impact of psychological factors on education, pp. 135–158. Academic Press, Orlando (2002)
14. Graesser, A.C., McDaniel, B., Chipman, P., Witherspoon, A., D'Mello, S., Gholson, B.: Detection Of Emotions During Learning With Autotutor. In: Proceedings of the 28th Annual Conference of the Cognitive Science Society, pp. 285–290. Erlbaum, Mahwah (2006)
15. Ang, J., Dhillon, R., Krupski, A., Shriberg, E., Stolcke, A.: Prosody-Based Automatic Detection Of Annoyance And Frustration In Human-Computer Dialog. In: Proceedings of the International Conference on Spoken Language Processing, Denver, pp. 2037–2039 (2002)
16. Grimm, M., Mower, E., Kroschel, K., Narayan, S.: Combining Categorical and Primitives-Based Emotion Recognition. In: 14th European Signal Processing Conference (EUSIPCO), Florence, Italy (2006)
17. Litman, D.J., Forbes-Riley, K.: Predicting Student Emotions In Computer-Human Tutoring Dialogues. In: Proceedings Of The 42nd Annual Meeting Of The Association For Computational Linguistics. Association for Computational Linguistics, East Stroudsburg, PA, pp. 352–359 (2004)
18. Shafran, I., Riley, M., Mohri, M.: Voice signatures. In: Proceedings IEEE Automatic Speech Recognition and Understanding Workshop, pp. 31–36. IEEE, Piscataway (2003)
19. Robson, C.: Real word research: A resource for social scientist and practitioner researchers. Blackwell, Oxford (1993)
20. D'Mello, S.K., Craig, S.D., Sullins, J., Graesser, A.C.: Predicting Affective States Through An Emote-Aloud Procedure From Autotutor's Mixed-Initiative Dialogue. International Journal Of Artificial Intelligence In Education 16, 3–28 (2006)
21. McDaniel, B.T., D'Mello, S.K., King, B.G., Chipman, P., Tapp, K., Graesser, A.C.: Facial Features for Affective State Detection in Learning Environments. In: McNamara, D.S., Trafton, J.G. (eds.) Proceedings of the 29th Annual Cognitive Science Society, pp. 467–472. Cognitive Science Society, Austin (2007)
22. Ekman, P., Friesen, W.V.: The facial action coding system: A technique for the measurement of facial movement. Consulting Psychologists Press, Palo Alto (1978)

Towards Emotionally-Intelligent Pedagogical Agents

Konstantin Zakharov[1], Antonija Mitrovic[1], and Lucy Johnston[2]

[1] Intelligent Computer Tutoring Group,
Department of Software Engineering and Computer Science
[2] Department of Psychology
University of Canterbury, Private Bag 4800, Christchurch 8140, New Zealand
{kon.zakharov,tanja.mitrovic,lucy.johnston}
@canterbury.ac.nz

Abstract. Research shows that emotions play an important role in learning. Human tutors are capable of identifying and responding to the affective states of their students; therefore, for ITSs to be truly affective, they should also be capable of tracking and appropriately responding to the emotional state of their users. We report on a project aimed at developing an affect-aware pedagogical agent persona for an ITS for teaching database design skills. We use the dimensional approach to affective modeling, and track the users' affective state along the valence dimension as identified from tracking the users' facial features. We describe the facial-feature tracking application we developed, as well as the set of rules that control the agent's behavior. The agent's response to the student's action depends on the student's cognitive state (as determined from the session history) as well as on the student's affective state. The experimental study of the agent shows the general preference towards the affective agent over the non-affective agent.

Keywords: facial feature tracking, affect recognition, emotional intelligence, affective pedagogical agents, evaluation.

1 Affective Gap in Intelligent Tutoring Systems

Computers have been implicitly designed without awareness of the affective communication channel. The lack of affective fit between technology and its users is particularly significant in Intelligent Tutoring Systems (ITSs): failing to acknowledge the complex interaction between the cognitive and affective processes ubiquitous in human activities, educational systems might never approach their full potential. Kort and Reilly [1] call for a re-engineering of the ITSs' pedagogy by shifting the focus of research towards expert teachers "*who are adept at recognizing the emotional state of learners, and, based upon their observations, take some action to scaffold learning in a positive manner*". In educational research the difference between learning performance under the *ideal* one-to-one tutoring conditions and other methods of instruction has been referred as the 2 Sigma problem [2]. It is very likely that the affective gap in ITSs can partially explain the 2 Sigma problem in the ITSs' context.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 19–28, 2008.

The semantic component of social interaction, most frequently taking the form of speech, is often accompanied by the affective interaction component, which is considered equally or sometimes even more important then the semantic component [3]. Although people in general are not always aware of how exactly their language, posture, facial expression and eye gaze convey their emotions, these underpin their interactions and navigation in the social world [4]. Recent research on affect recognition in computer-mediated environments opens new perspectives, although very little research has explored the ways in which a computer can be used to address the emotional state of its user in the learning context [5]. However, present-day ITS research is facing a wide range of interaction design and technical problems that arise during the development of affect-aware ITSs.

In this paper we present a pedagogical agent capable of active affective support, guided by the logic which integrates the learner's cognitive and affective states. Section 1 outlines the supporting research on cognitive and affective processes in the learning context. Section 2 presents our approach to affective state detection, while Section 3 describes its implementation. Section 4 presents the affective pedagogical agent we developed for EER-Tutor, an ITS for relational database design [6]. Section 5 describes the experiment and its outcomes. Finally, Section 6 concludes the paper by discussing our findings.

2 Affective Processes in the Learning Context

Prior research suggests a strong interaction between cognitive and affective processes in the human mind; in the educational context, stress, anxiety, and frustration experienced by a learner can severely degrade learning outcomes [7]. Researchers have been grappling with the question of how to define appropriate behavior within an interactive learning environment. Etiquette is highly context-dependent; consequently what may be appropriate in one context may be inappropriate in another. Generic HCI research emphasizes the need to avoid negative affective states such as frustration; frequently mentioned solutions include either (a) trying to determine and fix the problem causing the negative feelings, and/or (b) preemptively trying to prevent the problem from happening in the first place. However, there are some fundamental differences between general HCI etiquette and educational HCI etiquette. Learning from a computer is not just about ease of use; learning can be frustrating and difficult because it involves learners' exposure to errors and gaps in their thinking and knowledge [4].

Unfortunately, there is no cookbook defining all the rules for HHI (human-to-human interaction) that HCI and ITSs developers can simply implement; however, one simple rule of thumb suggested in the work of Mishra and Hershey [4] is to apply what has been found appropriate in HHI to the design of HCI. However, the feedback design in many computer-based educational systems is often based on the simplistic and erroneous assumption that praise is assumed to affect behavior positively, irrespective of context [4]. Recent studies with AutoTutor explore strategies to address boredom, frustration, flow and confusion[8]; AutoTutor detects affective states through conversational cues, posture and facial features.

Kort and Reilly [1] propose a model of constructive cognitive progress that relates learning and emotions in an evolving cycle of affective states. The model suggests

that this cycle, including its negative states, is natural to the learning process. The theory of flow, initially proposed by Csikzentmihalyi in 1990 [9], also attempts to tie together cognitive and affective processes. Flow is described as a mental state of operation in which people are fully immersed in what they doing; this state is characterized by a feeling of energized focus, full involvement, and success in the process. The study of flow has shown that conscious awareness of "flow zone" tends to diminish happiness and flow [9]. These findings suggest that conscious awareness of frustration, feeling of an impasse and other similar negative influences may diminish these states. In other words, affective self-awareness, fostered by affective support can assist users in mitigating the detrimental influences of negative affective states on their learning. Myers [10] describes two generic varieties of support for emotion regulation applicable in HHI: passive and active support. Passive support is used by people to manipulate moods, without necessarily addressing the emotions themselves. In contrast, active support occurs when people discuss or otherwise address their emotions directly as a means of managing them.

Bringing together the emotional self-regulation, Kort's theory of emotions in learning and the theory of flow, Burleson and Picard [11] implement an approach that uses affective agents in the role of peer learning companions to help learners develop meta-cognitive skills such as affective self-awareness for dealing with failure and frustration. In our research we adopt a similar approach by developing an affective agent playing the role of a caring tutor capable of offering active affective support.

If the pedagogical agents are to mimic the human tutors' affective behavior, the agents' designers need to endow them with social and emotional intelligence. Affective pedagogical agents should possess the knowledge of how to link the cognitive and affective experience of the learner in an attempt to meet the learner's needs; the agents should be able to choose an affective-behavioral strategy suitable for achieving the desired effect. Consequently, affective pedagogical agents need to embody a higher order of emotional behavior; they have to maintain the history and status of their own emotional state and that of the learners, and they have to have the capability of self-regulation of emotional state and support for the learner's emotional state [5].

3 Identifying Users' Affective States

There are two major theoretical approaches to the study of emotion: dimensional and categorical. Theorists who use the categorical approach to emotion attempt to define specific categories or types of emotions [12]. Research in this area suggests that there are a number of basic emotions (estimates range from three to more than 20) which combine to produce all the emotional states which people experience. The dimensional approach conceptualizes emotional space as having two or perhaps three underlying dimensions along which the entire range of human emotions can be arranged [13]. The most common dimensions are valence (ranging from happy to sad) and arousal (ranging from calm to excited).

In our research, we adopt the dimensional approach: the continuous nature of the valence dimension in this approach (versus the discrete states in the categorical approach) underpins the choices which determine the implementation of modeling of the agent's and user's emotions. The dimensional approach eliminates the need for

classifying the emotional states as belonging to certain categories; potentially, this resolves a number difficulties arising in emotion modeling—the label associated with a particular emotional display carries a lot less significance then the observed parameters of the emotional state.

Facial feature tracking techniques are based directly on action units listed in the Facial Action Coding System (FACS) [14]. Positive affective valence can be indexed through a decrease in the distance between the corner of the mouth on one side of a face—action unit #4 (Lip Corner Puller) activated by *Zygomaticus major*; this action results in the upward and outward turning of the mouth into a smile. Negative affective valence can be indexed through a decrease of the distance between the inner corners of eyebrows—action unit #12 (Brow Lowerer) activated by *Corrugator supercilii*; this results in the furrowed eyebrows look. We developed an algorithm for feature tracking which utilizes a combination of common image processing techniques, such as thresholding, integral projections, contour-tracing and Haar object classification; many of these operations are available though the OpenCV[1] library. Throughout the algorithm, the focus of attention is shifted among a number of regions of interest, determined on the basis of the anthropomorphic constraints describing human face geometry [15]. The algorithm relies on a few session-dependent threshold values for processing eye, brow and mouth regions. To accommodate lighting variations, the threshold values have to be chosen manually during algorithm calibration at the start of each tracking session. The feature detection algorithm includes five steps: (1) face region extraction; (2) iris detection; (3) outer eye corners detection; (4) mouth corners detection and (5) inner brow corners detection.

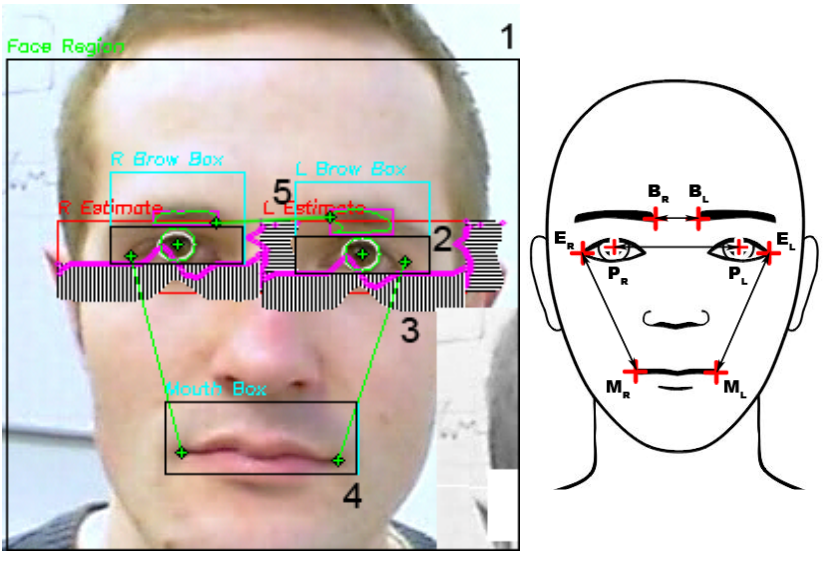


Fig. 1. The left image shows an example frame with the detected features. The right image shows the core features tracked by the feature-tracking algorithm.

[1] http://www.intel.com/technology/computing/opencv/—Open Source Computer Vision library.

Figure 1 shows facial features labeled according to the corresponding algorithm steps. Facial feature detection is the first stage of affect detection—the rest is based on the idea of facial animation parameter normalization described in [16]; in this technique the feature displacement is measured on the basis of a set of facial parameters for a neutral expression. At the start of every session, during calibration, the algorithm stores three parameters which are ratios of distances $E_RM_R + E_LM_L$ and B_RB_L to the distance between the pupils, P_RP_L shown in Figure 1. Real-time analysis of the differences between the ratios saved during calibration and the ratios calculated for each frame is the key to feature displacement. The algorithm does not attempt to determine affective state in every frame; rather our algorithm makes its decisions on the basis of observed changes throughout the session. Positive affective valence is indexed by the reduced distance between the corners of eye and mouth; negative valence is indexed by the reduced distance between the inner eyebrow corners. With every consecutive update received from the feature tracking code, the affective state is updated, registering transitions between negative, neutral and positive affective states.

4 Affective Pedagogical Agent for EER-Tutor

To accommodate the preferences of users, we created two female and two male Haptek[2] characters. The agents were designed to appear as young people approximately 20 to 30 years of age. Haptek's People Putty SDK allows for fine-grain control over the agent's features and behavior in a way which is consistent with the dimensional approach to emotion modeling. People Putty exposes a two-level API for controlling its characters' emotional appearance. On the lower level, the emotional appearance can be controlled through a set of parameters associated with the characters' facial features; these parameters define the position and shape of the eyebrows, the corners of the mouth, the overall position of the head and so on. We chose a subset of parameters to control the characters' appearance changes along the affective valence dimension ranging from sad to happy. Haptek characters use Microsoft Speech API-compatible Text-to-Speech (TTS) engines to generate verbal narrations along with realistic lip-sync movements. For the experimental studies we acquired two reportedly high-quality TTS Cepstral[3] voices—one male and one female.

The agent's persona is guided by a set of fifteen rules which implicitly encode the logic of session history appraisal. The rules assume that continuous lack of cognitive progress will be accompanied by a negative affective state, because the user will be dissatisfied with the progress of the current task; conversely, a satisfactory progress will result in a positive affective state. Each rule corresponds to a pedagogically-significant session state which requires the agent's response. For example there are rules requiring the agent to greet users when they sign on, submit a solution, ask for a new problem and so on. Each rule has a set of equivalent feedback messages determining the agent's verbal response; in addition, each rule includes a numeric value which triggers a change in the agent's affective appearance. For example, when the user reaches correct solution, along with a congratulatory message the agent responds with a cheerful smile. On the other hand, when the user is struggling with the solution

[2] http://www.haptek.com/—Haptek People Putty SDK site.
[3] http://www.cepstral.com/—Cepstral Text-to-Speech engires.

resulting in multiple submissions with errors, the agent's verbal response consists of the list of errors, along with an affective facial expression—the agent's face looks sad as if the agent is empathizing with the user (Figure 2). The agent's responses described above rely only on the appraisal of the cognitive state. The agent has its own affective module, which stores the current affective state; in the course of a session, the agent's affective state may be affected by the session events, but in the absence of the affect-triggering changes, the agent's affective state always gravitates towards the neutral state, as it is the case with human emotions.

The agent's affective awareness is intended to give the agent the capability of providing active affective support by addressing the user's feelings. The affective state appraisal is implicitly encoded in a set of rules corresponding to a subset of pedagogically-significant situations described above. The agent is capable of differentiating between positive and negative affective states; however, the agent addresses only steady negative affective states. The rationale for this approach is based simultaneously on the flow theory and on the model of cyclic flow of emotions in learning. The state of positive flow may be disrupted by making the subject aware of the flow; thus the agent does not need to interfere if there is no negative affect. When the user is happy with the state of the session, it is unlikely the agent's affective feedback will improve anything, even if the agent is beaming with happiness and enthusiasm; if anything, such an interference may break the mood or unnecessarily distract the user. On the other hand, making the subject aware of their negative state may distract them from their negative feelings and move them along towards their goal. Apart from

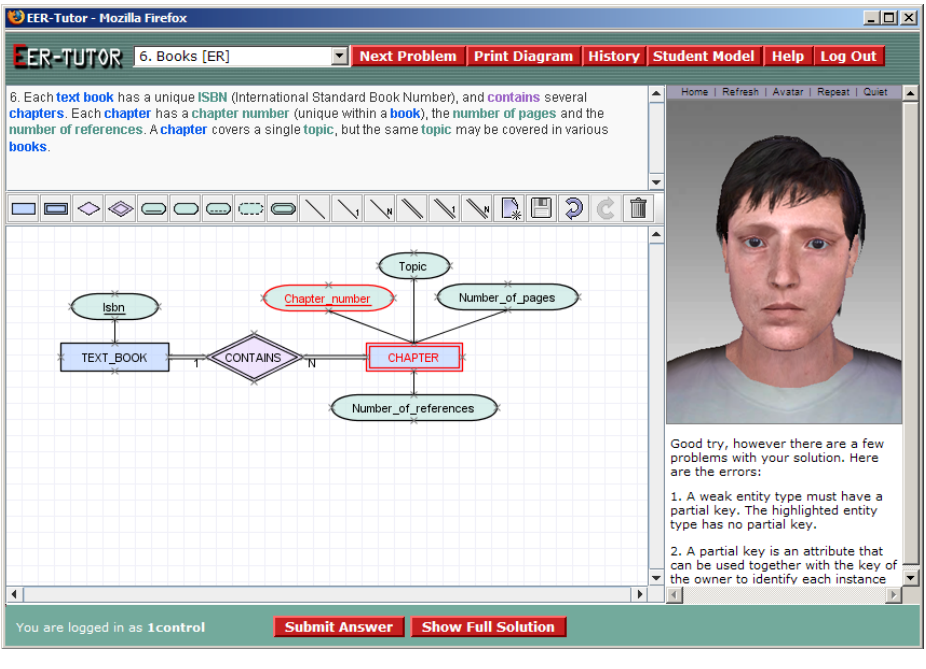


Fig. 2. The state of the agent after a few consecutive incorrect solution submissions

responding to user-generated interface events (i.e. solution submission), the agent can intervene with affect-oriented messages when the user's affective state degrades.

Certainly, the affect-oriented messages triggered by negative affective states run the risk of making a bad situation worse, because a user afflicted by negative feelings might consider any interruptions irritating. With this in mind, we tried to design our agent's behavior to be as unobtrusive as possible; the agent only provides affect-oriented content if the subject's facial feature tracking data indicates the dominance of the negative affective state. In our implementation, the interval to be taken into consideration is a configurable parameter; for the evaluation study it was set to two minutes. Thus the agent, while responding to interface events, such as solution submission, may add an affect-oriented message to its feedback only if the negative affect has been prevalent during the last two minutes and if the user did not receive affective feedback during that time. The same logic is applied to the agent's affective interjections in the absence of interface events. The following are examples of feedback messages used by the agent for affect-oriented feedback both for user-generated events and unsolicited affective interventions—these messages are intended to address the user's negative feelings and express empathy in a way suitable in the given context:

- "I'm sorry if you are feeling frustrated—it's just that some of the problems demand a lot of work."
- "I apologize if you feel negative about this practice session—some of the solutions are quite complex."
- "It does not look like you are not enjoying this practice session—but if you keep solving these problems, you will be better prepared for future assessment."

5 Experiment

In order to evaluate the affective agent, we performed a study in an introductory database course in March–April 2007. The experimental group had access to the affect-aware version of the agent, while the control group had the affect-unaware version of the agent. This version of the agent was guided by rules without affect-oriented content so the agent did not generate affective facial or verbal reactions, but always remained neutral. The task-oriented feedback for both conditions was identical. The participants were randomly allocated to the control and experimental conditions. All users were familiar with EER-Tutor, because the class was introduced to the sans-agent version of this system a week before the study.

The study was conducted as a series of individual sessions, one session per participant. A webcam for facial feature tracking was mounted on top of the monitor and aimed at the participant's face. We used the Logitech Quick-Cam Pro 5000 webcam, operating at the frame rate of 15 fps. at a resolution of 640×480px. For improving the accuracy of facial feature tracking, we ran the sessions in a controlled lighting environment—two 1000W video-studio lights were pointed away from the participant towards a white screen, which worked as a source of diffused white light. Participants wore head-phones to hear the agent's feedback.

The participants were expected to spend 45-minutes with EER-Tutor, while solving problems of their choice from EER-Tutor's curriculum at their own pace. Before each session the experiment convener provided a verbal description of the task. At the

end of the session, the participants were asked fill out the questionnaire. Finally the participant was debriefed on the nature of the experiment and the underlying research.

A total of 27 participants took part in the experiment—13 participants (11 male, 2 female) were allocated to the control and 14 (13 male, 1 female) to the experimental condition. The average age in the control and experimental conditions was 22 (s = 6.5) and 24 (s = 7.1) years respectively. We did not expect to observe significant difference between the conditions in the objective learning performance measures, because the sessions were short; therefore the between-condition comparison was made on the basis of the questionnaires responses.

There was no significant difference between the groups in reported levels of expertise in database modeling. The main positive outcome of the evaluation was determined by the responses to questions, which ranked the appropriateness of the agent's behavior and its usefulness: 64% of the experimental groups thought that the agent's behavior was appropriate, compared to only 30% of the control group; furthermore, 43% of the experimental group rated the agent as a useful addition to EER-Tutor, compared to the 15% of the control group. The affect-aware agent's behavior was rated higher than the affect-neutral agent in terms of both its behavior (Mann-Whitney U Test, $U = 57$, $N_C = 13$, $N_E = 14$, $p < 0.05$) and usefulness (Mann-Whitney U Test, $U = 56$, $N_C = 13$, $N_E = 14$,$p < 0.05$). There was no significant difference in the users' perception of learning and enjoyment levels with either version of the agent. The rankings of the participants' perceptions of the agents' emotional expressiveness did not reveal significant difference between the two conditions—30% in the control condition noticed the agent's emotions versus 15% in the experimental condition. This result is somewhat unexpected, because the affective facial expressions were generated only for the experimental condition.

Free-form questionnaire responses suggest that the participants received the affect-aware version with interest and approval; for example, one participant commented: "I liked when the avatar told me I shouldn't worry because of feeling uncomfortable about the question I was working on." Three participants, however, stated they felt annoyed when the agent misdiagnosed their affective state; one user commented: "The avatar kept asking me if I was feeling negative when I wasn't." Another comment suggests that we, as interaction designers, were not clear enough about the intention the agent was to communicate to the user: "I needed encouragement when I wasn't doing very well, but instead got a sad face"—this particular user clearly did not find the agent's empathy encouraging. In this situation, verbal expression of empathy combined with a positive facial expression could have had a better effect.

Verbalized feedback was enthusiastically welcomed by the participants; even though the questionnaire did not elicit comments on verbalized feedback, 33% of participants (approximately equal proportions for each condition) stated that verbal feedback was a useful addition to EER-Tutor because it helped the users to remain focused on the workspace and work faster. Only 11% stated that verbalized feedback was unnecessary or distracting. The participants were able to silence the agent and stop verbal feedback; only 20% used this feature, but all these participants turned the verbal feedback back on within one to four minutes. The participants' interest in the verbal feedback can be explained in the work of Nass and Brave [17], who offer the evidence that the awareness of non-human origin of speech is not enough for the "*brain to overcome the historically appropriate activation of social relationships by*

voice." These findings have a vast potential for the future development of affective pedagogical agents in ITSs.

It appears that the agent's presence raised the level of users' expectations associated with the EER-Tutor's ability to guide them and provide hints. This functionality is implicit in EER-Tutor, because at any time the list of errors can be obtained by submitting an incomplete solution; in the agent's presence, however, some users wanted the agent to take it upon itself to provide context-specific hints. We observed in some cases that the agent did not match the users' expectations in its ability to take control of the situation and provide context-specific unsolicited task-oriented hints and assistance when the participants were struggling with the task. For example, one user commented: "When it was obvious I was lost, the avatar didn't offer any tips or appropriate questions." Another user commented that without this ability the agent was "a helper that was not very helpful."

In general, approval of the pedagogical agent's presence in EER-Tutor dominates the questionnaire responses. The agents' uptake was not unanimous, but the evaluation results advocate the presence of affective pedagogical agents, with the affect-aware agent demonstrating superiority over its non-affective counterpart.

6 Conclusions and Discussion

The active affective support that we attempted to implement theoretically takes the interaction between the agent and learner to a new level—it brings the pedagogical agent closer to the learner. This opens a whole new horizon of social and ethical design issues associated with the human nature traits. The experiment results indicate a range of preferences associated with pedagogical agents and affective communication. Affective interaction is individually driven, and it is reasonable to suggest that in task-oriented environments affective communication carries less importance for certain learners. Also, some learners might not display emotions in front of a computer, or some users might display emotions differently; even though people do tend to treat computers and other digital media socially, it does not necessarily mean that people's responses in the HHI and HCI contexts are equivalent. On the other hand, some people are naturally more private about their feelings; such individuals might respond to the invasion of their private emotional space by a perceptive affective agent with a range of reactions from withdrawal to fear. Others might resent being reminded about their feelings when they are focusing on a cognitive task; in such situations, people might unconsciously refuse to acknowledge their feelings all together. Although the interplay of affective and cognitive processes always underpins learning outcomes, affective interaction sometimes may need to remain in the background; whatever the case, an ITS should let the user decide on the level of affective feedback, if any, thus leaving the user in control [18].

Affective state in learning environments and in HCI in general is known to be negatively influenced by the mismatch between the user's needs and the available functionality; inadequate interface implementations, system limitations, lack of flexibility, occurrences of errors and crashes—all these factors contribute to the affective state. In most cases, it is difficult or virtually impossible to filter out the affect generated by this kind of problems. These considerations add another level of complexity to the modeling of such ill-defined domains as the human-like emotional behavior.

At the same time, the inevitable and undisputed truth is that humans are affective beings, guided by a complex system of emotions, drives and needs. Some aspects of affect-recognition in HCI may forever remain ill-defined or hidden, just as in some HHI scenarios one can never be completely and utterly sure of the perceived experience. Affective agents may improve the learner's experience in a variety of ways, and these will be perceived differently by every individual learner; agents may ease frustration and may make the process more adaptive, interesting, intriguing or appealing. If affect-recognition and affective agents can attract more learners and improve learning outcomes, thus taking the ITS research a step closer to bridging the affective gap and possibly resolving the 2 Sigma problem, then this step is worth taking.

References

1. Kort, B., Reilly, R.: An Affective Module for an Intelligent Tutoring System. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 955–962. Springer, Heidelberg (2002)
2. Bloom, B.S.: The 2 Sigma Problem: The Search for Method of Group Instruction as Effective as One-to-One Tutoring. Educational Researcher 13, 4–16 (1984)
3. Bickmore, T.W.: Unspoken Rules of Spoken Interaction. Communications of the ACM 47, 38–44 (2004)
4. Mishra, P., Hershey, K.A.: Etiquette and the Design of Educational Technology. Communications of the ACM 47, 45–49 (2004)
5. Klein, J., Moon, Y., Picard, R.W.: This Computer Responds to User Frustration: Theory, Design, and Results. Interacting with Computers 14, 119–140 (2002)
6. Zakharov, K., Mitrović, A., Ohlsson, S.: Feedback Micro-engineering in EER-Tutor. In: Proc. AIED 2005, pp. 718–725 (2005)
7. Goleman, D.: Emotional Intelligence. Bantam Books, New York (1995)
8. D'Mello, S., Picard, R., Graesser, A.: Toward an Affect-Sensitive AutoTutor. IEEE Intelligent Systems 22, 53–61 (2007)
9. Csikzentmihalyi, M.: Flow: The Psychology of Optimal Experience. HarperPerennial, New York (1990)
10. Myers, D.G.: Psychology. Worth Publishers, New York (1998)
11. Burleson, W., Picard, R.W.: Gender-Specific Approaches to Developing Emotionally Intelligent Learning Companions. IEEE Intelligent Systems 22, 62–69 (2007)
12. Plutchik, R.: Emotions: A Psychoevolutionary Synthesis. Harper and Row, New York (1980)
13. Bradley, M.M., Lang, P.J.: Measuring Emotion: Behavior, Feeling and Physiology. In: Cognitive Neuroscience of Emotion, pp. 242–276. Oxford University Press, New York (2000)
14. Ekman, P., Friesen, W.V.: Facial Action Coding System. Consulting Psychologists Press (1978)
15. Tian, Y.-L., Kanade, T., Cohn, J.: Recognizing Action Units for Facial Expression Analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 23, 97–115 (2001)
16. Pandzic, I.S., Forchheimer, R.: MPEG-4 Facial Animation: The Standard, Implementation and Applications. John Wiley and Sons (2002)
17. Nass, C., Brave, S.: Wired for Speech: How Voice Activates and Advances the Human-Computer Relationship. The MIT Press (2005)
18. Kay, J.: Learner Control. User Modeling and User-Adapted Interaction 11, 111–127 (2001)

Viewing Student Affect and Learning through Classroom Observation and Physical Sensors

Toby Dragon[1], Ivon Arroyo[1], Beverly P. Woolf[1], Winslow Burleson[2], Rana el Kaliouby[3], and Hoda Eydgahi[4]

[1] Department of Computer Science, University of Massachusetts-Amherst
[2] Arts, Media and Engineering Program, Arizona State University
[3] Media Lab, Massachusetts Institute of Technology
[4] Department of Electrical Engineering, Massachusetts Institute of Technology
{dragon,arroyo,bev}@cs.umass.edu, winslow.burleson@asu.edu, kaliouby@media.mit.edu, hoda@mit.edu

Abstract. We describe technology to dynamically collect information about students' emotional state, including human observation and real-time multi-modal sensors. Our goal is to identify physical behaviors that are linked to emotional states, and then identify how these emotional states are linked to student learning. This involves quantitative field observations in the classroom in which researchers record the behavior of students who are using intelligent tutors. We study the specific elements of learner's behavior and expression that could be observed by sensors. The long-term goal is to dynamically predict student performance, detect a need for intervention, and determine which interventions are most successful for individual students and the learning context (problem and emotional state).

1 Introduction and Previous Work

The obvious next frontier in computational instruction is to systematically examine the relationship(s) between student affective and learning outcome (performance) [18]. Human emotion is completely intertwined with cognition in guiding rational behavior, including memory and decision-making [18,11,16,5]. Students' emotion towards learning can have a drastic effect on their learning experience [10]. An instructor who establishes emotional and social connections with a student in addition to cognitive understanding enhances the learning experience. Responding to a learner's emotion, understanding her at a deep level, and recognizing her affect (e.g. bored, frustrated or disengaged) are key elements of quality teaching. If computer tutors are to interact naturally with humans, they need to recognize affect and express social competencies. This research attempts to understand how students express emotion, detect these emotions, and quantify emotional variables.

Previous projects have produced computational tutors that recognized and responded to models of emotion (e.g., self-efficacy and empathy [15]). Projects have tackled the sensing and modeling of emotion in learning and educational gaming environments [14, 17]. A dynamic decision network was used to measure student emotional state based on variables such as heart rate, skin conductance and eyebrow

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 29–39, 2008.

position [7]. Studies have evaluated the impact of affective interface agents on both affective and motivational outcomes based factors (e.g., gender, ethnicity). Lack of engagement was shown empirically to correlate with a decrease in learning [4]. In this study, however the tutor elicited negative feelings from students, in part because it blocked those who were presumed to be gaming the system [1]. Most prior work on emotion recognition has focused on deliberately expressed emotions within a laboratory setting and not in natural situations such as classroom learning. Many of earlier systems did not use fully adaptive learning environments and some were games. The research described here takes the next step by integrating emotion detection within an intelligent tutor as part of learning in a natural classroom setting.

2 Overall Plan

The long-term goal of this research is to dynamically collect information about students' emotional state in order to predict performance, detect a need for intervention, and determine which interventions are most successful for individual students and context (problem, emotional state). To accomplish these tasks, we implement emotion detection within an existing tutor in three phases: classroom observations, the use of physiologic sensors, and software algorithms (e.g., machine learning). We triangulate among these approaches to resolve toward agreement (with the realization that we may be far away from realizing any consensual agreement). This paper describes the first two methods for detection of emotion; classroom observations and a sensor platform.

In the first phase of this research human observation in the classroom approximated the type of information the sensors would collect, and corroborated what sensor information indicates about students' emotional state. Classroom observations are a useful exploratory strategy because human observers can intuitively discern high-level behaviors and make appropriate judgments on limited information that may be difficult to automatically decide from raw sensor data.

In the second phase we evaluate low-cost portable and readily deployable sensors that dynamically detect emotion using the theoretical basis formed from classroom observations. Sensors are can collect constant streams of data in parallel, allowing for much more consistent observation than a human ever could accomplish. They are also increasingly inexpensive and fast at processing/collecting data. Thus, human observations identify behaviors that are worth observing and then sensors gather this behavioral data in bulk. We will evaluate the effectiveness of sensors in predicting student emotional state, and use reinforcement-learning techniques to decide which interventions are most successful for students in certain emotional states.

3 Classroom Observations

Our goal in the first phase of this research was to observe student behavior and identify variables that represented 1) emotions and desirable/undesirable states linked to student learning, and 2) physical behaviors linked to emotion states. This involved quantitative field observations in the classroom in which researchers recorded the

behavior of students using intelligent tutors. Observations by multiple observers, using this method, have had high inter-rater reliability and report relatively low impact on student behavior once students are used to the observer's presence [4]. Researchers observed students using the Wayang Mathematics Tutor, a tutor that prepares 12-16 year old students for the mathematics section of standardized exams [2]. The tutor, which has been used by a thousand of students represents mathematic skills and recognizes which skills a student has learned. It shows students their progress and offers them a choice of problem difficulty.

3.1 Experimental Design

The study included thirty four (34) students in a public school in urban Holyoke, MA, split into 3 different classes. Students took a pretest survey to evaluate their attitudes towards math (self-concept and value) and goal (learning vs. performance) orientation [10], as well as a mathematics pretest with multiple problems to evaluate diverse concepts taught within the Wayang Outpost math tutoring software. Students used the tutoring software during a period of 3 weeks and were then given a posttest. While students used the Wayang software, three researchers coded behavioral variables and subjective variables, such as valence of the student's emotion. Researchers were trained during several sessions to code these variables by observing videos of students using Wayang. Coders rotated around the classroom, coding one student at a time. Observation periods lasted for approximately 15 seconds, with the following 15 seconds to confirm the observation. Because students may have experienced several behaviors/emotions during one time period (e.g., the student was seen forward and then back on the chair), we coded the first state seen, but the second one was coded and taken account later in the analysis.

Behavioral and Task-Based Variables. Researchers coded physical behavior (chair and head posture, movement, face gestures) and looked for expressed affect in specific facial expressions (smile, frown, nod) and verbal behavior (loud comments, talk with others). They also coded whether a student appeared to be on- or off-task. The process of identifying this behavior is obviously somewhat subjective and noisy (i.e. a student may look to be on task when they are not). Students were marked as being off-task when they were clearly not using the software appropriately. This includes not looking at the screen, using other programs on the computer, staring blankly at the screen without taking any action, conversing with peers about other subject matter, etc [4]. On-task students might be reading/thinking about the problem, talking to a friend about the problem, or writing a solution on paper. Off-task students are not concentrated/engaged on learning and this is undesirable for learning.

Emotional Indicators. Because it is often difficult to distinguish one emotion from another, we limited the conventional emotional terms to four categories of emotions that result from the combination of two indicators: (i) *valence* (positive or negative nature of the emotion/energy the student seemed to be expressing) and (ii) *arousal* or level of physical activity. These emotion indicators are used to express the four basic emotions in Table 1, and are consistent with early research on emotions [20]. However, our concern was that this emotional state variable might not be correlated to learning without also considering on-task or off-task behavior. It is highly desirable for a student to experience a state of joy/excitement when she is on-task, but if the

Table 1. Desirable State Variables and Possible Emotion Indicators

Valence	Arousal	On/Off task	Example Student Behavior	Desirability value	
+	+	On	Aha moment, yes! That's it!	2	Highly Desirable
+	-–	On	Concentrated on problem-solving	2	Highly Desirable
-–	+	On	Frustrated with tutoring software,	1	Maybe desirable
-–	-–	On	Yawning, zoned out within software	0	Not desirable
+	+	Off	Laughing with friend	0	Not desirable
+	-–	Off	Very focused but on other software	0	Not desirable
-–	+	Off	Angry quarrel with friend	0	Not desirable
-–	-–	Off	Zoned out, or sleeping	0	Not desirable

student tends to be joyful while off-task, the emotion variable will not correlate strongly with optimal learning. Thus, we created another variable, *Desirability Value*, which is both task- and emotion-dependent (on/off-task, valence and arousal), see Table 1. The values reflect the fact that being off-task is undesirable, but also that being tired/bored (negative valence, negative arousal) while being on-task is also not desirable, as the student may give up. Frustration while being on-task is not necessarily negative; learning episodes often have productive moments of frustration. Finally, states of positive valence while being on-task are highly desirable, whether accompanied by high arousal or by low levels of arousal where students experience high mental activity without significant observable emotional expression.

3.2 Results

We evaluated correlations among the frequency of behaviors, task and emotional state variables. Correlations were computed between global emotion indicators and intermediate emotion/task-based state variables. Then we analyzed the correlation between these state-based variables and student behaviors. Students were detected to be on-task 76% of the time, slightly lower than previous findings regarding off/on-task behavior with software learning environments [3].

Table 2 shows the frequencies of different emotional states. Note that negative valence emotions were observed only 8% of the time. This could be largely due to the fact that a neutral or indiscernible valence was coded as positive. Table 2 shows that 73% highly desirable states were observed, 3% medium desirable states, and 24% non-desirable states.

Table 2. Frequency of Emotion Indicators and Desirable Learning States

Emotion indicators: Valence & Arousal	Frequency	Percent
+ valence & --arousal (concentrated, satisfied)	148	58%
+ valence & + arousal (excited, joyful, actively engaged)	85	34%
- valence & +arousal (frustrated, angry)	16	6%
- valence & --arousal (bored, tired)	5	2%
Total	254	100%

Desirable State	Frequency	Percent
Highly desirable	181	73%
Not desirable	61	24%
Medium Desirable	7	3%

Correlations Between Emotion Indicators and Learning/Attitudes. We analyzed whether we can use emotional indicators and other state variables to predict learning and motivation, the variables we want to optimize.

Valence. Valence (or student energy) was significantly correlated to pretest math score (N=34, R=.499, p=.003). This suggests that students who are good in math to begin with, also have substantially more positive emotions while using the software, or at least less unpleasant emotions (e.g. boredom, frustration). Valence was also positively correlated to posttest learning orientation (N=30, R=.499, p<.01), but not to pretest learning orientation, suggesting that having positive valence during the tutoring session may instill higher *learning orientation* goals at posttest time. A similar effect happened for posttest self-concept and valence (R=.48, p<0.01) where students who had higher valence emotions had higher posttest self-concept scores. Thus, the presence of *positive* or *negative* emotions can help predict more general attitudes towards math at posttest time.

Arousal. Arousal (or student activity) was negatively correlated with pre-tutor learning orientation (N=30, R=-.373, p<0.05), suggesting that students who are *performance-oriented* (characterized by a desire to be positively evaluated by others) are more likely to be physically active or 'aroused', as opposed to those who are *learning oriented,* who tend to express less physical activity.

Emotion (Valence + Arousal). Our emotional scale was correlated with pretest self-concept (R=.385, p<0.05) and posttest learning orientation (R=.463, p<.05), suggesting that the presence of four types of emotions (determined by combinations of valence and arousal) can help predict more general attitudes towards learning math.

On/Off task. Being on-task is significantly correlated to posttest self-concept in math (N=30, R=.442, p=.02), but not to pretest self-concept in math, suggesting that being on-task is not a result of an incoming high self-concept in math. However, it indicates that being on-task may generate better self-concept after using the tutor. There is a significant correlation between math posttest performance and being on-task (R=.640, p<.018). Again, being on-task is not correlated with math pretest performance, meaning that prior math knowledge will not predict students' tendencies towards on or off-task behavior. Instead, being on-task seems to lead to higher posttest scores, again implying that being engaged with the tutoring system is part of the reason for achieving higher posttest scores. This is consistent with past research results on on/off task behavior [3]. If we can encourage students to be on-task, we will foster better attitudes for math and higher posttest scores.

Desirable Learning State. Similar significant correlations were found for this variable as on/off task (i.e., it predicted posttest scores and posttest self-concept in math to a similar extent as on/off task behavior). If we can encourage students to be in our desirable learning states (Table 1), we will also foster better attitudes for math and higher posttest scores.

Correlations Between Emotional/Task-Based States and Behavior. Several correlations were discovered among student behavior (chair, head and hand position), emotion indicators (valence and arousal) and the desirability value, see Table 3. Clearly, a high positive correlation exists for arousal and chair movement since we defined arousal by physical activity. Meanwhile, valence is not linked to chair movement, meaning that students do not express their positive or negative emotions with chair

movement. A negative correlation exists for desirable state and being on-task, meaning that students are in a more desirable learning state (and more on-task) when they don't move so much in the chair.

Other interesting findings (some not shown) are that students with positive valence emotions tend to sit in the middle of the chair, instead of being towards the side, the front or the back of the chair. Last, students leaning on their hands correlated negatively with arousal –as leaning is a fairly inactive posture. It is not that obvious though that students in a state of positive valence also tend to lean on their hands.

Table 3. Pearson correlations among student behavior (chair, head and hand position), emotion indicators (valence and arousal), the desirability value and student talk

	VALENCE	AROUSAL	ON TASK?	Desirability Value	TALK
Chair Movement N =	-.467 (0.46*) 252	.420 (.000***) 252	-.140 (.027*) 249	-.154 (.015*) 247	-----
CHAIR Middle N =	.148 (.018*) 252	.107 (090) 252	-.002 (.974) 249	-.003 (.967) 247	-----
HEAD MOVE	-.224 (.000***) 249	.345 (.000***) 249	-.417 (.000***) 246	-.435 (.000***) 244	
HEAD SIDE	-.195 (.002**) 254	.247 (.000***) 254	-.325 (.000***) 251	-.337 (.000***) 249	-----
HEAD MOVE SIDE N =	-.270 (.000***) 249	.230 (.000***) 249	-.422 (.000***) 246	-.443 (.000***) 244	-----
HEAD MIDDLE N =	.202 (.000***) 254	-.186 (.000***) 254	.427 (.000***) 251	.436 (.000***) 249	-----
HEAD UP N =	-.097 (.123) 254	.062 (.326) 254	-.214 (.001**) 251	-.235 (.000***) 249	-----
TALK N =	-.117 (.064) 251	.304 (.000***) 251	-.644 (.000***) 251	-.628 (.000***) 249	-----
SOUND N =	-.075 (.248) 242	.370 (.000***) 242	-.388 (.000***) 241	-.379 (.000***) 239	-----
SMILE N =	-.086 (.185) 240	.313 (.000***) 240	-.430 (.000***) 237	-.420 (.000***) 235	.485 (.000***) 237
NEUTRAL N =	.142 (028*) 240	-.238 (.000***) 240	.395 (.000***) 237	.409 (.000***) 235	-.285 (.000***) 237
SOUND N =	-.075 (.248) 242	.370 (.000***) 242	-.388 (.000***) 241	-.379 (.000***) 239	.533 (.000***) 241

*** Correlation is significant at the 0.001 level (2-tailed); ** Correlation is significant at the 0.01 level (2-tailed); * Correlation is significant at the 0.05 level (2-tailed).

Head movement was correlated with negative valence, high arousal, off-task behavior and non-desirable states. This implies that students move their heads when they feel negative emotions, when being off-task and in a non-desirable learning state. When students are in such unproductive learning state, and when they are off-task, they tend to move their heads to the side. Also, students tend to move their head to the side when they have negative feelings. It is possible that students avoid the computer screen when they don't feel good about the software or the learning situation. At the same time, having their head in the middle had the opposite effect: it was correlated with positive valence, low arousal, on-task behavior, and desirable state for learning.

Students holding their head up indicates off-task behavior and an undesirable state for learning, while holding their head down is not (possibly because many students

tend to work on paper on their desk). Again, head up could be an indication of screen avoidance. Talking and environmental sound are both correlated to high arousal and positive emotion, although they are associated with off-task behavior and undesirable states. This means that students tend to have off-task talk, which seems reasonable for a system that does not encourage on-task collaboration with a partner.

It seems obvious that frowning is related to having a negative valence emotion. However, frowning doesn't appear to be a good predictor of being on-task or being in a desirable learning state (not shown). A smile on the face does predict off-task behavior (R=-.430 with on-task) and undesirable state for learning (R=-.420), Table 3. Surprisingly, smiling was not linked to valence, but it is positively correlated with arousal and talk (students probably moved and talked with friends while they smiled). The opposite effect happened for a neutral face: it was positively correlated to desirable learning state and on-task behavior. A neutral face was linked to positive valence, most likely because we coded seeing a neutral emotion as positive valence. A neutral face was an indicator that the student was not moving (negative arousal) and not talking. Last, an environmental sound that is louder than background noise was a good predictor of talking (R=.533) suggesting that a microphone that senses for odd sounds can detect if a student is talking with good accuracy, which in turn was evidence for a non-desirable state for learning within the software, see Table 3.

4 Sensor Technology

Fig. 1. Pressure mouse sensor

These human observations in the classroom are continuing as a way to understand the impact of student emotions on learning. Yet these student emotions can be detected automatically by intelligent tutors, which can then also respond dynamically with appropriate interventions. In order to establish a social and emotional connection with students, tutors should recognize students' affect and respond to them at a deep level. Towards this end, our goal in the second phase was to automate the observation process using sensors. We have developed a low cost multimodal sensor platform that is being integrated into the Wayang Tutor and evaluated in classrooms. The platform includes a custom produced Pressure Mouse, a Wireless BlueTooth Skin Conductance sensor, a Posture Analysis Seat, and a Facial Expression System. This platform expands on an earlier one at an order of magnitude reduction in the overall cost. The sensors are developed from an earlier system that had several sensors in common with AutoTutor [9]. Pre-production prototypes of each sensor have been developed and we are producing thirty sets of these sensor platforms for simultaneous use in classrooms. The intent is to provide a better understanding of student behavior and affect and to determine the contribution of each sensor to the modeling of affect [14].

Pressure mouse. A pressure mouse is used to detect the increasing amounts of pressure users place on their mice related to their increased levels of frustration. The pressure mouse system has six force sensitive resistor sensors and an embedded microprocessor, Figure 1. It uses the standard communication channel of a USB mouse for pointing and clicking functions and then in parallel uses a second channel, a serial communications port, to provide pressure data at 20ms intervals from each of the six sensors. Pressure sensors located under the mouse button measure the force of the users click in addition to their overall pressure across the surface of the mouse.

Posture Analysis Seat. We have developed and are now testing a low-cost/low resolution pressure sensitive seat cushion and back pad with an incorporated accelerometer to measure elements of a student's posture and activity, Figure 2. This system captures many student movements relevant to education that were previously captured by the TekScan system, that used an extremely expensive Posture Analysis Seat, developed for medical and automotive applications [19]. The previous system used pattern recognition techniques while watching natural behaviors to *learn* which behaviors tended to accompany states such as interest and boredom. We are now developing similar algorithms based on the new low-cost posture analysis chair.

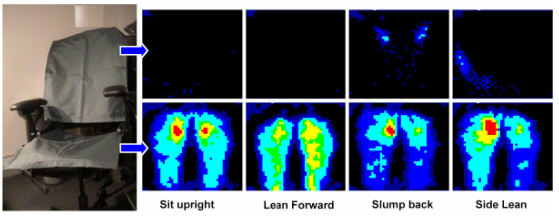


Fig. 2. Posture State Chair Sensor. The previous sensor resulted in posture recognition (89-97% accurate). And classification of high/low interest and break taking (69-83% accurate) [14].

Wireless skin conductance. A wireless version of a earlier glove that sensed conductance was developed by Carson Reynolds and Marc Strauss at the MIT Media Lab, in collaboration with Gary McDarby, at Media Lab Europe, see Figure 3. While the skin conductance signal is not valenced (i.e. does not describe how positive or negative the affective state is) it is strongly correlated with arousal. High levels of arousal tend to accompany significant and attention-getting events [6].

Facial Expression Camera. A person's mental state is not directly available to an observer; instead it is inferred from a range of non-verbal cues including facial expressions. We are using a facial expression recognition system that incorporates a computational model of mind reading as a framework for machine perception and mental state recognition [12]. This facial action analysis is based on a combination of bottom-up vision-based processing of the face (e.g. head nod or smile) with top-down predictions of mental state models (e.g. interest and confusion) to interpret the meaning underlying head and facial signals over time [12]. A multilevel, probabilistic architecture (using dynamic Bayesian networks) mimics the hierarchical manner

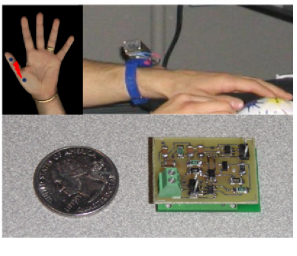


Fig. 3. Wireless Skin Conductance Sensor

with which people perceive facial and other human behavior [21] and handles the uncertainty inherent in the process of attributing mental states to others. The output probabilities represent a rich modality that technology can use to represent a person's state and respond accordingly. The resulting visual system infers mental states of people from head gestures and facial expressions in a video stream in real-time. At 30 fps, the inference system locates and tracks 24 feature points on the face and uses motion, shape and color deformations of these features to identify 20 facial and head movements (e.g., head pitch, lip corner pull) and 11 communicative gestures (e.g., head nod, smile, eyebrow flash) [21]. Dynamic Bayesian networks model these head and facial movements over time, and infer the student's "hidden" affective-cognitive state.

5 Discussion and Future Work

This paper described the use of human observations and wireless sensors to detect student emotions, learning, and attitudes towards learning. We identified emotion indicators (valence and arousal) that combined with on and off-task variables to represent desirable/undesirable states linked with student learning, as well as physical behaviors linked to emotional states. This was achieved through quantitative field observations in the classroom in which researchers recorded the behaviour of students using intelligent tutors. We described correlations between low-level observations (i.e. chair movement) and higher-level observations (valence, arousal, on-off task behavior) and then between these higher-level observations and student learning and attitudes. Through these links, we propose that low-level sensor information can tell us about emotion indicators and other state-variables linked to learning. Sensors can provide information about how students perform and information about when students are in non-productive states so that the tutor can provide appropriate interventions. In turn, sensors can also inform us whether the given interventions are working or not. With this goal in mind, low cost portable sensors are being used in natural classroom settings. Thus, once we know which variables are useful predictors of learning and affective outcomes, these sensors can replace the human observers and predict students' emotional states related to learning.

Table 4. Guide to interpreting sensor data and predicting learning

⇐⇐ **Predict student learning and attitude** ⇐⇐			
Desirable Learning States	**Emotion/task indicators**	**Biologic indicators**	**Sensors to use**
Most desirable (Joy, Aha moment, Concentrated Actively engaged)	+ Valence AND On-task	Lean on hand; Little chair/head movement; Sit in middle of chair; Head in middle; Neutral face;	Chair sensors Camera
Medium desirable (Frustrated, angry)	-- Valence + Arousal	Head movement; Chair movement; Squeezing of mouse	Camera, Pressure mouse; Chair Sensors
Least desirable (Bored, tired)	Off-task OR -- Valence -- Arousal	Talking; Large chair movement; Head movement; Head to side or head up; Smile	Skin conductance; Camera; Chair sensors; Microphone
⇒⇒ **Detect strong and weak student learning behavior** ⇒⇒			

This paper unveiled several interesting findings: 1) observed fluctuating states of emotion and on/off task behavior help predict posttest performance and attitudes/motivation; 2) student states are expressed with specific behaviors that can be automatically detected with sensors; and 3) a mechanism for strong/weak learning behavior detection was identified. As a result of these findings we identify how sensors can predict and reflect student learning, see Table 4. Moving from right to left sensor readings and emotion/biologic indicators are used to predict student learning and other motivational variables; moving from left to right indicates how strong/weak learning and attitudes are expressed and detected by sensors.

Future work consists of using these behaviors to predict emotions and desirable/undesirable learning states that would in turn help us predict learning and attitudes towards learning mathematics. The long-term goal is to dynamically collect information about students' emotional state and predict student states, and in turn predict posttest performance in real time. Moreover, because certain states such as negative valence and high levels of arousal are unproductive for post-tutor assessments of learning/attitudes, such states will lead to the selection of an intervention. At that point we must also decide which interventions are most successful for individual students and context (e.g. topic, emotional state). Finally, we intend to resolve the nature of data from different sensors. The camera provides very high-level judgments as it uses its own inference engine to decide emotional states, whereas all other sensors provide relatively raw data. We are engaged in the development of machine learning algorithms that relate these data sets to learners' diverse emotional states. Using all of these techniques, we plan to recognize and help students cope with states of negative valence and support their return to on-task behavior.

Acknowledgement. We would like to thank Roopesh Konda and Joshua Richman for their work on this project. The material is based upon work supported by the National Science Foundation, IIS Grant #0705554. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Aleven, V., Roll, I., McLaren, B., Ryu, E.J., Koedinger, K.: An architecture to combine meta-cognitive and cognitive tutoring: Pilot testing the Help Tutor. In: Proceedings of the 12th International Conference on AIED (2005)
2. Arroyo, I., Ferguson, K., Johns, J., Dragon, T., Meheranian, H., Fisher, D., Barto, A., Mahadevan, S., Woolf, B.P.: Repairing Disengagement with Non-Invasive Interventions. In: Proceedings of the 13th International Conference of AIED. IOS Press (2007)
3. Baker, R.S.J.d.: Modeling and Understanding Students' Off-Task Behavior in Intelligent Tutoring Systems. In: Proceedings of ACM CHI 2007. Computer-Human Interaction (2007)
4. Baker, R.S.J.d., Corbett, A., Koedinger, K.: Detecting Student Misuse of Intelligent Tutoring Systems. In: Proceedings of the Seventh International Conference on Intelligent Tutoring Systems, pp. 531–540 (2004)
5. Block, J.: On the relation between IQ, impulsivity and deliquency. Journal of American Psychology 104, 395–398 (1995)

6. Boucsein, W.: Electrodermal activity. Plenum Press, New York (1992)
7. Conati, C.: Probabilistic assessment of users' emotions in educational games. Journal of Applied Artificial Intelligence, special issue on Merging Cognition and Affect in HCI 16(7-8), 555–575 (2002)
8. Cytowic, R.E.: Synesthesia: A union of the senses. Springer, New York (1989)
9. D'Mello, S.K., Picard, R., Graesser, A.C.: Toward an affect-sensitive AutoTutor. IEEE Intelligent Systems 22(4), 53–61 (2007)
10. Dweck, C.S.: Self-theories: Their role in motivation, personality and development. The Psychology Press, Philadelphia (1999)
11. Goleman, D.: Emotional intelligence: Why it can matter more than IQ. Bantam, New York (1995)
12. Kaliouby, R., Robinson, P.: Real-time Inference of Complex Mental States from Facial Expressions and Head Gestures. In: Real-Time Vision for HCI, pp. 181–200. Spring-Verlag (2005)
13. Kapoor, A., Mota, S., Picard, R.W.: Towards a learning companion that recognises affect. In: AAAI Fall Symposium 2001, North Falmouth (2001)
14. Kapoor, A., Picard, R.W., Ivanov, Y.: Probabilistic Combination of Multiple Modalities to Detect Interest. In: International Conference on Pattern Recognition, August 2004, Cambridge (2004)
15. McQuiggan, S., Lester, J.: Diagnosing Self-Efficacy in Intelligent Tutoring Systems: An Empirical Study. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053. Springer, Heidelberg (2006)
16. Norman, D.A.: Twelve issues for cognitive science. In: Perspectives on cognitive science, pp. 265–295. Erlbaum, Hillsdale (1981)
17. Sheldon-Biddle, E., Malone, L., McBride, D.: Objective measurement of student affect to optimize automated instruction. In: Proceedings of Workshop on Modelling User Attitudes and Affect, User Modeling 2003(2003)
18. Shute, V.J.: Focus on formative feedback. ETS Research Report, #RR-07-11, Princeton (2006)
19. Tekscan: Tekscan Body Pressure Measurement System User's Manual. Tekscan Inc., South Boston, MA, USA (1997)
20. Wundt, W.: Outlines of Psychology. 2nd rev. English ed. Williams &Norgate, London (1902)
21. Zacks., J.M., Tversky, B., Iyer, G.: Perceiving, remembering, and communicating structure in events. Journal of Experimental Psychology: General 130, 29–58 (2001)

Comparing Learners' Affect While Using an Intelligent Tutoring System and a Simulation Problem Solving Game

Ma. Mercedes T. Rodrigo[1], Ryan S.J.d. Baker[2], Sidney D'Mello[3], Ma. Celeste T. Gonzalez[4], Maria C.V. Lagud[4], Sheryl A.L. Lim[1], Alexis F. Macapanpan[4], Sheila A.M.S. Pascua[4], Jerry Q. Santillano[1], Jessica O. Sugay[1,4], Sinath Tep[1], and Norma J.B. Viehland[4]

[1] Department of Information Systems & Computer Science, Ateneo de Manila University
[2] Human-Computer Interaction Institute, Carnegie Mellon University
[3] Institute for Intelligent Systems, University of Memphis
[4] Education Department, Ateneo de Manila University
mrodrigo@ateneo.edu, rsbaker@cs.cmu.edu

Abstract. We compare the affect associated with an intelligent tutoring environment, Aplusix, and a simulations problem solving game, The Incredible Machine, to determine whether students experience significantly better affect in an educational game than in an ITS. We find that affect was, on the whole, better in Aplusix than it was in The Incredible Machine. Students experienced significantly less boredom and frustration and more flow while using Aplusix. This implies that, while aspects unique to games (e.g. fantasy and competition) may make games more fun, the interactivity and challenge common to both games and ITSs may play a larger role in making both types of systems affectively positive learning environments.

1 Introduction

Games are fun. The same adolescents who are often reluctant to put significant time into their studies are often enthusiastically willing to put dozens of hours into playing modern computer games [6]. In recent years, researchers have suggested that embedding games into education can be a way to improve students' affect, interest, and motivation towards education, and in turn improve their learning. Some educational games have successfully built upon competition, curiosity, challenge, and fantasy to make learning more enjoyable, increase students' desire to learn, and complete more difficult work than with traditional educational materials [1,8,16]. However, there is also evidence that educational games may not have entirely positive effects on learners' affect and motivation. Bragg [5] found that students exhibited negative attitudes towards the use of games as the main instructional method for learning mathematics. Similarly, Vogel [24] argues that games and simulations that fail to make seamless connections between the subject matter and the game play will also inhibit learners' engagement and motivation.

While it is commonly believed that educational games will lead to better affect than non-gamelike learning environments, the evidence supporting this belief is not yet

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 40–49, 2008.

conclusive. In many cases, educational games have been studied in relation to relatively weak comparison conditions, such as paper worksheets with no feedback [16] and games with game features ablated [8]. Furthermore, it has been found that intelligent tutoring systems lead to significantly improved affect and motivation as compared to traditional, non-computerized learning contexts [22], though not necessarily to expert human tutors. Intelligent tutors generally lack game-like features like competition and fantasy, but share in common with games features such as instant feedback, and measures of continual progress. It is possible that the additional motivational features of educational games lead to more positive affect than intelligent tutors (i.e. more delight and engagement, and less frustration and boredom), but it is also possible that the largest motivational benefits come from the interactivity that both games and intelligent tutors share.

The differences among effects of educational games and intelligent tutoring systems on students' usage choices are also not yet fully studied. Consider hint abuse and systematic guessing, behaviors categorized as gaming the system, i.e. "attempting to succeed in an educational environment by exploiting properties of the system rather than by learning the material and trying to use that knowledge to answer correctly" [3]. As Rodrigo et al [20] discussed, students generally know that gaming behavior is undesirable in intelligent tutoring systems, as the primary goal is to learn the domain content – and students demonstrate this belief by hiding this behavior from their teachers. By contrast, there may be a perception that since games are primarily for fun, it is acceptable to use them in any fashion; hence, students may game the system more often in educational games than in intelligent tutoring systems.

In this paper, we compare the affect associated with an intelligent tutoring environment, Aplusix II: Algebra Learning Assistant [17,18] (http://aplusix.imag.fr/), and a simulation problem solving game, The Incredible Machine: Even More Contraptions [20]. Earlier studies [4,20] collected affect and usage data on students' affective states when using The Incredible Machine. We collect similar data for Aplusix, within similar populations and following virtually identical data collection and analysis procedures. By comparing these two data sets, we can determine whether students experience significantly better affect in an educational game than in an intelligent tutor, and in turn study which aspects of educational games explain their positive effects on student affect.

1.1 Descriptions of the Learning Environments

As mentioned in the introduction, affect and usage data were gathered from participants using two different interactive learning environments: the Incredible Machine and Aplusix.

The Incredible Machine [21], called TIM for short, is a simulation game where students complete a series of logical "Rube Goldberg" puzzles. In each puzzle, the student is given a limited set of objects, including mechanical tools like gears, pulleys, and scissors; more active objects like electrical generators and vacuums; and even animals. The student must combine these objects in a creative fashion to accomplish each puzzle's goal. Objectives range from relatively straightforward goals, such as lighting a candle, to more complex goals such as making a mouse run. If a student is stuck, he or she can ask for a hint; hint messages display where items should be

located in a correct solution to the current problem (without displaying which items should be placed in each location). TIM is thought to be highly entertaining, having won multiple awards for its innovative gameplay, most recently including an award for best casual mobile game at the 5th Annual Spike TV Video Game Awards in 2007. Hence, if an intelligent tutor can produce comparable levels of affect as those produced by this game, the ITS can be considered highly motivating. A screenshot from TIM is shown in Figure 1.

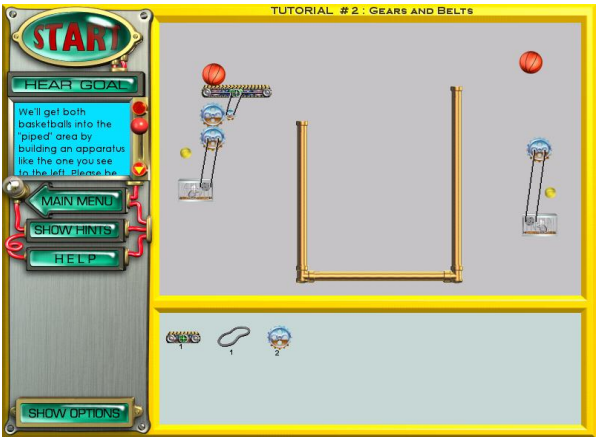


Fig. 1. A screen shot from The Incredible Machine: Even More Contraptions (TIM)

Aplusix [18,19] (http://aplusix.imag.fr/) is an intelligent tutoring system for mathematics. Topics are grouped into six categories (numerical calculation, expansion and simplification, factorization, solving equations, solving inequations, and solving systems), with four to nine levels of difficulty each. Aplusix presents the student with an arithmetic or algebraic problem from a problem set chosen by the student and allows the student to solve the problem one step at a time, as he or she would using a paper and pen. At each step, Aplusix displays equivalence feedback: two black parallel bars mean that the current step is equivalent to the previous step, two red parallel bars with an X mean that the current step is not equivalent to the previous step (see Figure 2). This informs the student about the state of the problem in order to guide him or her towards the final solution. Students can end the exercise when they believe they are done. Aplusix then tells the student whether errors still exist along the solution path or whether the solution is not in its simplest form yet. The student has the option of looking at the solution, a "bottom out" hint with the final answer. Hence, Aplusix both reifies student thinking and gives instant feedback, two key characteristics of modern intelligent tutoring systems [cf. 2]. However, Aplusix lacks one game-like feature found in many intelligent tutoring systems – indications of the probability that students have learned relevant skills, in the form of "skill bars". It has been suggested that students view skill bars as being like points in games, and that skill bars give students the perception of progress and encourage competition between students [22], although, in a lab study, Jackson & Graesser [14] did not find evidence that progress-only skill bars improve motivation.

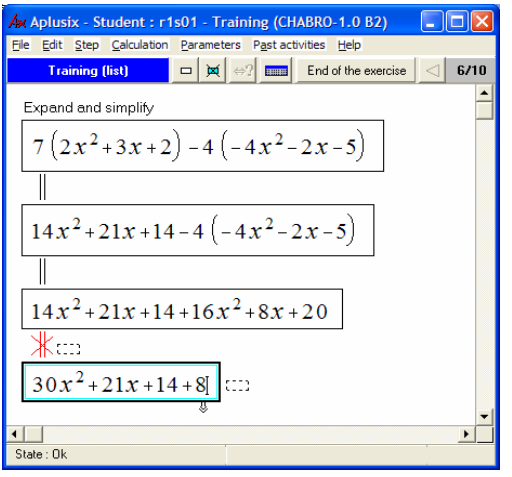


Fig. 2. A screen shot from Aplusix: Algebra Learning Assistant

These two systems provide a strong comparison between intelligent tutoring systems and games. TIM has won awards for its enjoyable gameplay; Aplusix can be considered a fairly traditional intelligent tutoring system, as it includes the continual feedback and reification of student thinking that is characteristic of most intelligent tutoring systems, but lacks skill bars, which some researchers think lend intelligent tutors a game-like feel. Hence, the two systems are good representatives of their respective classes, and similarities or differences in learner affect between the two systems will be representative of similarities or differences in affect between games and ITSs in general. It is worth noting that the two systems do not cover the same educational material, as TIM covers general problem-solving skill while Aplusix covers algebra; this possible confound will be considered in the discussion section.

2 Methods

The data gathering procedures for the two environments was very similar. The subsequent section discusses the profile of the participants, the observers, the coding procedures, and the inter-rater reliability of the observations.

The participants for the TIM study were students in a private high school in Quezon City (Metro Manila), the Philippines. Student ages ranged from 14 to 19, with an average and modal age of 16. Thirty-six students participated in this study (17 female, 19 male). The participants in the Aplusix study were first and second year high school students from four schools within Metro Manila and one school in Cavite, a province south of Manila. Students' age ranged from 12 to 15 with an average age of 13.5 and a modal age of 14 (high school begins earlier in the Philippines than in many other industrialized nations). One hundred and forty students participated in the Aplusix study (83 female, 57 male). The participants in both studies were computer-literate. However, none of them had previously used either TIM or Aplusix. The sample of participants did not overlap between studies.

Each student used TIM for ten minutes, and each student used Aplusix for 45 minutes. The different time spent in each system is a potential confound. In specific, this difference might lead to greater boredom or frustration within the Aplusix system (because students may experience more boredom or frustration later in a learning session) – if either of these effects is found, it may be due to differences between the studies rather than differences between the systems. Students used the software in small groups (9 for The Incredible Machine, 10 for Aplusix), one student per computer, during their class time. Each student's affect was observed several times as he or she used the learning software.

The observations were carried out by a team of six observers, working in pairs. The observers were Masters students in Education or Computer Science, and all but one had prior teaching experience. The set of observers was overlapping but not identical between systems. TIM was studied in 2006 [20] Aplusix was studied in 2007. Each observation lasted twenty seconds, and was conducted using peripheral vision, i.e. observers stood diagonally behind or in front of the student being observed and avoided looking at the student directly [cf. 3], in order to make it less clear when an observation was occurring. If two distinct affective states were seen during an observation, only the first affective state observed was coded; similarly, if two distinct behaviors were seen during an observation, only the first behavior observed was coded. Any behavior by a student other than the student currently being observed was not coded. Each pair of observers was assigned to a small number of students and alternated between them – more observers participated in the TIM study than the Aplusix study, thus a greater amount of time passed between observations in Aplusix (180 seconds) than The Incredible Machine (40 seconds).

In the studies, both affect and behavior were coded. The observers trained for the task through a series of pre-observation discussions on the meaning of the affective and behavior categories. Observations were conducted according to a guide that gave examples of actions, utterances, facial expressions, or body language that would imply an affective state, and observers practiced the coding categories during a pilot observation period prior to the studies. The guide was based on earlier work by [3,11], and is discussed in detail in [20]. The affective categories coded were boredom, confusion, delight, surprise, frustration, flow, and neutral, in line with earlier research by D'Mello et al [11] suggesting that these states are most relevant to students' affective experiences within an Intelligent Tutoring System. "Flow" refers to full immersion in an activity; the participant is focused on a task to the point that he or she is unaware of the passage of time [cf. 10]. The behavior categories coded were on-task, on-task conversation, off-task conversation, off-task solitary behavior, inactivity, and gaming the system; in both systems, gaming behavior consisted of systematic guessing – such as trying an object in every possible place in TIM – and use of help features to arrive at a solution without engaging in problem-solving.

706 observations were collected in TIM, for an average of 19.6 observations per student. Inter-rater reliability was acceptably high across all observations — Cohen's [7] κ=0.71 for usage observations, κ=0.63 for observations of affective state. Thirteen pairs of observations were collected per student in Aplusix, totaling 3,640 observations in all. Inter-rater reliability was again acceptably high: Cohen's κ=0.78 for usage observations, κ=0.63 for observations of affective state.

3 Results

3.1 Prevalence of Affective States

The most common affective state in both Aplusix and TIM was flow, occurring 62% of the time in TIM and 68% of the time in Aplusix. The difference in the prevalence of flow between environments was marginally statistically significant, t(174)= -1.66, two-tailed p=0.10, for a two-tailed, two-sample t-test with pooled variance.

The second most common affective state in both environments was confusion, occurring 11% of the time in TIM and 13% of the time in Aplusix. The difference in the prevalence of confusion between environments was also not statistically significant, t(174)= 0.73, two-tailed p=0.46. Delight was also not significantly different between environments, t(174) = 0.55, two-tailed p=0.58.

However, the frequency of two negative affective states was significantly different between systems. Frustration was more common in TIM (6%) than Aplusix (2%), t(174)=3.25, two-tailed p=0.001. Boredom was also more common in TIM (7%) than Aplusix (3%), t(174)=2.27, two-tailed p=0.02.

The overall pattern of results (shown in Figure 3) is that the affective experiences were, on the whole, more positive within Aplusix than TIM, with the effect more pronounced among negative affective states than positive affective states.

3.2 Prevalence of Negative Usage Behaviors

Gaming the system occurred in both Aplusix and The Incredible Machine. The average student gamed the system 1.4% of the time in Aplusix, about half of the prevalence in previous observations of gaming behavior in Cognitive Tutors [cf. 3]; the average student gamed the system 7.5% of the time in The Incredible Machine, about double the prevalence in previous observations of gaming behavior in Cognitive Tutors. The difference between the prevalence of gaming in the two environments was statistically significant, $t(174)=4.72$, $p<0.0001$, for a two-tailed two-sample t-test with pooled variance.

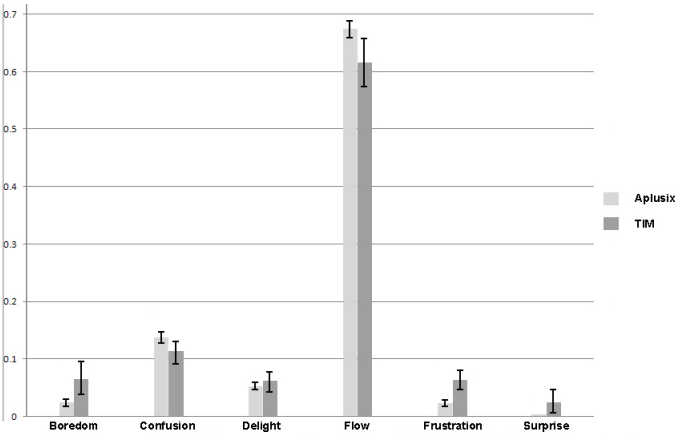


Fig. 3. Affective categories' prevalence of occurrence (standard error bars shown)

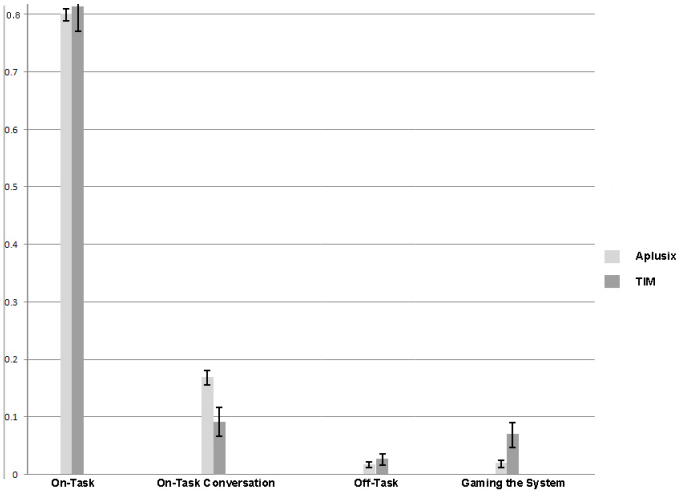


Fig. 4. Usage categories' prevalence of occurrence (standard error bars shown)

There was the appearance of a difference in the prevalence of off-task behavior between the two environments, with students being off-task 2.2% of the time in TIM and 1.3% of the time in Aplusix, but this difference was not statistically significant, $t(174)= 1.25$, two-tailed $p=0.21$.

The time spent on-task, working with the system, within the two environments, was almost identical: 80.9% on-task in TIM, 79.9% on-task in Aplusix, $t(174)=0.33$, two-tailed $p=0.74$. However, the time spent on-task, talking to another student or the teacher, was significantly higher in Aplusix (17.3%) than TIM (9.4%), $t(174)= -3.14$, two-tailed $p=0.001$. Hence, the overall pattern of results (shown in Figure 4) is that students spent significantly more time gaming the system in TIM, and significantly more time in on-task conversation in Aplusix.

4 Discussion and Conclusions

In this paper we have asked: are educational games associated with better affect because they are games, or simply because they are highly interactive learning environments? We investigated that question by comparing the incidence of positive and negative affective states and usage behaviors in an intelligent tutoring system, Aplusix, and a simulation problem solving game, The Incredible Machine.

Considering the high popularity of The Incredible Machine as a game, it would be reasonable to expect students using that environment to experience more positive affect, and less negative affect than students using an intelligent tutoring system such as Aplusix. At the same time, it might be reasonable to expect more students to game the system when playing The Incredible Machine than Aplusix, since by its very nature a game may encourage gaming the system relative to an intelligent tutor.

The evidence from our research partially aligns with these expectations. There is indeed more gaming the system in TIM than Aplusix; however, surprisingly, affect

was on the whole better within Aplusix than TIM – there was significantly more boredom and frustration in TIM, and a less flow.

This suggests that a well-designed intelligent tutoring system can lead to equally positive – or even more positive – affect than an educational game. In turn, this suggests that while factors such as fantasy may make games more fun [cf. 8], the interactivity and challenge common to both games and intelligent tutors may play a larger role in making games affectively positive learning environments.

The results in this paper are not fully definitive, however, for four reasons. First, there are a number of differences between the two studies. Although the two studies were conducted by the same research group with a single methodology, TIM and Aplusix cover different subject matter and the studies were conducted with samples recruited in different years (and differing subtly, demographically) rather than with random assignment within a single population. This is not a fatal flaw for the study presented here, but does suggest that its result should be replicated before being treated as proven truth (as, in fact, all research results should be). Second, TIM and Aplusix differ pedagogically from each other in a number of ways. In comparing an intelligent tutor to an educational game, multiple substantial differences between environments are unavoidable; games have several characteristics that distinguish them from other types of interactive environments [19], as do intelligent tutoring systems [23]. A comparison that varied on only one factor would not fairly represent one type of environment or the other; however, determining *which* factors lead to the largest positive improvements on student affect and behavioral choice will be key. Third, TIM and Aplusix differed substantially in terms of curricular relevance. While TIM fostered problem solving skills in general, Aplusix focused specifically on Algebra, a subject that the participants were studying at the time. Participants may have perceived Aplusix as relevant to the larger goal of getting good grades in mathematics, motivating them to invest more effort and attention when using the software [cf. 15]. Finally, affect's impact on learning can be counterintuitive. Positive affect in some cases appears to reduce perseverance and increase distraction [12]. On the other hand, the affective state of confusion, sometimes considered negative, has been shown to promote deep thinking [9].

In recent years, there has been rapidly increasing interest in educational games. Some of this interest has been based on the hypothesis that games will lead to better affect than existing learning environments [cf. 8,13]. However, in the research reported here, we have found that a traditional intelligent tutoring system can produce equally good – or better – affect as an award-winning educational game. The key question, therefore, appears not to be which type of learning environment is better, but how we can leverage the best practices developed by each of these design communities in order to develop a new generation of engaging and educationally effective learning environments.

Acknowledgements

We thank Jean-Francois Nicaud of the Laboratoire d'Informatique de Grenoble for the use of Aplusix. We also thank the Ateneo de Manila High School, Kostka School of Quezon City, School of the Holy Spirit of Quezon City, St. Alphonsus Liguori Integrated

School and St. Paul's College Pasig for their participation in the studies conducted. The work presented in this paper was made possible in part by a grant from the Ateneo de Manila University and by the Pittsburgh Science of Learning Center which is funded by the National Science Foundation, award number SBE-0354420.

References

1. Alessi, S.M., Trollip, S.R.: Multimedia for Learning: Methods and Development, 3rd edn. Allyn & Bacon, Needham Heights (2001)
2. Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive tutors: Lessons learned. The Journal of the Learning Sciences 4(2), 167–207 (1995)
3. Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z.: Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game the System". In: Proceedings of ACM CHI 2004: Computer-Human Interaction, pp. 383–390 (2004)
4. Baker, R.S.J.d., Rodrigo, M.M.T., Xolocotzin, U.: The dynamics of affective transi-tions in simulation problem-solving environments. In: Paiva, A., Prada, R., Picard, R.W. (eds.) ACII 2007. LNCS, vol. 4738, pp. 666–677. Springer, Heidelberg (2007)
5. Bragg, L.: Students' conflicting attitudes towards games as a vehicle for learning mathematics: A methodological dilemma. Mathematics Education Research Journal 19(1), 29–44 (2007)
6. Brown, J.S.: New learning environments for the 21st century. The Magazine of Higher Learning 38(5), 18–24 (2005)
7. Cohen, J.: A Coefficient of Agreement for Nominal Scales. Educational and Psychological Measurement 20, 37–46 (1960)
8. Cordova, D.I., Lepper, M.R.: Intrinsic motivation and the process of learning: Beneficial effects of contextualization, personalization, and choice. Journal of Educational Psychology 88, 715–730 (1996)
9. Craig, S.D., Graesser, A.C., Sullins, J., Gholson, B.: Affect & learning: an exploratory look into the role of affect in learning with AutoTutor. Journal of Educational Media 29(3), 241–250 (2004)
10. Csikszentmihalyi, M.: Flow: The Psychology of Optimal Experience. Harper and Row, New York (1990)
11. D'Mello, S.K., Craig, S.D., Witherspoon, A., McDaniel, B., Graesser, A.: Integrating affect sensors in an intelligent tutoring system. In: Affective Interactions: The Computer in the Affective Loop Workshop. In conjunction with International conference on Intelligent User Interfaces, pp. 7–13 (2005)
12. Dreisbach, G., Goschke, T.: How positive affect modulates cognitive control: Reduced perseveration at the cost of increased distractability. Journal of Experimental Psychology: Learning, Memory, and Cognition 30(2), 343–353 (2004)
13. Gee, J.P.: What video games have to teach us about learning and literacy. Palgrave Macmillan, New York (2003); Keller, C.: Motivational Design of Instruction. In: Reige-luth, C.(ed.) Instructional Design Theories and Models: An Overview of their current status. Lawrence Erlbaum Associates, Mahwah (1983)
14. Jackson, G.T., Graesser, A.C.: Content Matters: An investigation of feedback categories within an ITS. In: Luckin, R., Koedinger, K., Greer, J. (eds.) Artificial Intelli-gence in Education: Building Technology Rich Learning Contexts that Work. IOS Press, Amsterdam (2007)

15. Keller, J.M.: Strategies for Stimulating the Motivation to Learn. Performance and Instruction Journal 28(8), 1–7 (1987)
16. Lee, J., Luchini, K., Michael, B., Norris, C., Solloway, E.: More than just fun and games: Assessing the value of educational video games in the classroom. In: Proceedings of ACM SIGCHI 2004, pp. 1375–1378 (2004)
17. Nicaud, J.F., Bouhineau, D., Chaachoua, H.: Mixing microworld and CAS features in building computer systems that help student learn algebra. International Journal of Computers for Mathematical Learning 9(2), 169–211 (2004)
18. Nicaud, J.F., Bouhineau, D., Mezerette, S., Andre, N.: Aplusix II [Computer software] (2007)
19. Prensky, M.: Digital Game-Based Learning. Paragon House, St. Paul (2007)
20. Rodrigo, M.M.T., Baker, R.S.J.d., Lagud, M.C.V., Lim, S.A.L., Macapanpan, A.F., Pascua, S.A.M.S., Santillano, J.Q., Sevilla, L.R.S., Sugay, J.O., Tep, S., Viehland, N.J.B.: Affect and usage choices in simulation problem-solving environments. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Artificial Intelligence in Education: Building Technology Rich Learning Contexts that Work. IOS Press, Amsterdam (2007)
21. Sierra Online, The Incredible Machine: Even More Contraptions [Computer Software] (2001)
22. Schofield, J.W.: Computers and Classroom Culture. Cambridge University Press, Cambridge (1995)
23. VanLehn, K.: The behavior of intelligent tutoring systems. International Journal of Artificial Intelligence in Education 16(3), 227–265 (2006)
24. Vogel, J.J., Greenwood-Ericksen, A., Cannon-Bowers, J., Bowers, C.A.: Using virtual reality with and without gaming attributes for academic achievement. Journal of Research on Technology in Education 39(1), 105–118 (2006)

What Are You Feeling? Investigating Student Affective States During Expert Human Tutoring Sessions

Blair Lehman[1], Melanie Matthews[1], Sidney D'Mello[2], and Natalie Person[1]

[1] Department of Psychology, Rhodes College, 2000 North Parkway,
Memphis, TN 38112 USA
{lehba,matmv,person}@rhodes.edu
[2] Institute for Intelligent Systems, The University of Memphis,
365 Innovation Drive, Memphis, TN 38152 USA
sdmello@memphis.edu

Abstract. One-to-one tutoring is an extremely effective method for producing learning gains in students and for contributing to greater understanding and positive attitudes towards learning. However, learning inevitably involves failure and a host of positive and negative affective states. In an attempt to explore the link between emotions and learning this research has collected data on student affective states and engagement levels during high stakes learning in one-to-one expert tutoring sessions. Our results indicate that only the affective states of confusion, happiness, anxious, and frustration occurred at significant levels. We also investigated the extent to which expert tutors adapt their pedagogical and motivational strategies in response to learners' affective and cognitive states.

1 Introduction

It is widely acknowledged that one-to-one human tutoring is a powerful method for promoting active knowledge construction, increased conceptual understanding, augmented self-efficacy, and a more positive learning attitude – all factors that foster engagement and ultimately impact learning gains [1, 2]. Furthermore, it is also documented that accomplished (or expert human tutors) have a higher impact on learning than unaccomplished tutors (novices), and Intelligent Tutoring Systems (ITSs) [1, 2, 3]. However, while it is not feasible for every student to have access to an expert human tutor, ITSs are available to anyone with a computer. Therefore, one plausible solution is to model ITSs after expert human tutors, a task that requires a detailed understanding of expert tutoring strategies. So what do expert human tutors do? Lepper and Woolverton [4] have claimed that individualization, immediacy, and interactivity are the three major factors that enable expert tutors to be more effective than traditional learning in the classroom. Through modeling and monitoring student knowledge, tutors have the ability to adapt to the specific needs of individual students [5]. In addition to cognitive scaffolds, it has been claimed that expert tutors also provide motivational and emotional support for students in social, affective, and emotional ways [6].

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 50–59, 2008.

However, the story of expert tutoring is not restricted to pedagogical and motivational strategies; emotions (or affective states) play an important role in positive learning outcomes [7, 8, 9]. While certain emotional states such as the *flow* experience, [7] where the student is completely absorbed in the learning process, or confusion, where the learner experiences cognitive disequilibrium and is forced to think, are positively correlated with learning [10], other states such as frustration, boredom, anxiety, and despair can negatively impact learning [8]. For example, learners who are unsure of their ability often avoid tasks or give up when they encounter difficulties [11]. Thus, emotional states that are associated with low self-efficacy, such as feelings of anxiety, interfere with learning because the student is no longer fully motivated or engaged with the material.

Research on the cognitive and motivational strategies of tutors is quite extensive [5, 12, 13, 14, 15, 16], whereas empirical research on the affective dimension in tutoring is considerably more sparse and scattered. Furthermore, it is unclear whether expert human tutors divert more attention to respond to students' cognitive states or motivational and affective states. For example, Cromley and Azevedo [17] studied the practices of more and less experienced reading tutors to determine if there was a priority placed on cognitive or motivational scaffolding. More experienced tutors were found to use significantly less forms of motivational scaffolding and significantly more forms of cognitive scaffolding than less experienced tutors.

A very different depiction of expert tutoring emerges from the INSPIRE model proposed by Lepper and Woolverton [4]. According to this model the most effective tutors are highly knowledgeable about their domain and pedagogical strategies, develop a rapport with their students, utilize a Socratic method, plan for effective use of time, are indirect in their feedback, encourage articulation of acquired knowledge, and use a variety of techniques to maintain student engagement (p. 145-150). By utilizing a Socratic Method tutors are allowing students to construct their own knowledge within a highly organized framework for the progression of the session. Tutors are also creating a learning environment in which students feel comfortable and can develop greater confidence and self-efficacy. These highly effective tutors are very attentive to all of the student's needs and respond in ways to support both learning gains and affective experiences.

Our interest in the affective dimension of expert tutoring comes from a desire to build ITSs that are based on the pedagogical, motivational, and affective strategies of expert tutors. In particular, we seek answers to the following questions. What are the student emotions that occur during expert tutoring sessions? How do expert tutors adapt their pedagogical and motivational strategies to incorporate students' affective states? In this paper we describe a study that collected data on student affective states during 40 one-to-one expert tutoring sessions. We focused on a list of affective states that was obtained by investigating current theories of emotion [18, 19, 20], research on learning [7, 8, 21], and empirical research [10, 22, 23, 24]. The affective states were confusion, frustration, anxious, anger, fear, sadness, disgust, contempt, surprise, happiness, eureka, and curiosity. In addition to student affect, student engagement was also investigated as a separate construct with four levels: disengagement, socially attending, actively attending, and full engagement.

2 Method

Participants
Our participants consisted of eight expert tutors and 29 students. Tutors and students had been working together prior to the start of this study. Expert status for a tutor was defined as licensed to teach at the secondary level, having five or more years of tutoring experience, being recommended by local school administrators, and working for a professional tutoring agency. Some students had two sessions with the same expert tutor. The unit of analysis in this study was the tutor-student dyad. The subjects studied were algebra, geometry, physics, chemistry, basic math, and standardized test preparation.

Procedure
Tutors and students were given an informed consent form to read and sign. The session lasted approximately one hour**.** All sessions were videotaped with a camera that was positioned at a great enough distance to not disturb the tutoring session but still close enough to record sound and visual data. The researcher left the room during the tutoring session.

Data Treatment
The videos were digitized and then transcribed. They were then coded with respect to student cognitive states, tutor pedagogical and motivational strategies, and student affective states and engagement levels.

Coding Student Affective States and Engagement Levels – Student affective states for each one hour session were coded along two dimensions: Ekman's six "basic" emotions [18] and a set of learning-centered affective states [8, 23, 25, 22, 24]. The list of affective states with definitions appears in Table 1. Affective events were considered to be specific instances of the tutoring session where emotions could be detected through facial movements, perceivable paralinguistic cues of speech, and gross body movements. When an affective state was perceived, the engagement level of the student was also recorded with respect to four engagement levels as illustrated in Table 1. Engagement Levels were considered to be the degree to which a student has invested mental resources to the topic during perceivable affective states. We computed reliabilities using Cohen's Kappa for the affective states that occurred at a significant level in each analysis. Kappas for the five affective states that occurred consistently were: happiness (.80), confusion (.65), frustration (.72), anxious (.68), and contempt (.66). Sessions were divided evenly between two coders after sufficient levels of reliability were achieved, such that each coder was responsible for individually coding half of the expert tutoring sessions.

Coding Tutor Pedagogical and Motivational Moves – The coding scheme developed by Person et al. [26] was used to code 14 pedagogical and 10 motivational tutor moves. The pedagogical moves were developed from past research on the ways in which tutors aid students in problem solving and knowledge construction [5, 17, 27, 28, 29, 13, 30]. Some of these dialogue moves include direct instruction, simplified problem, hint, and comprehension gauging question. From research on the practices of expert tutors, the motivational strategies that have been used by expert tutors include such dialogue moves as positive feedback, negative feedback, humor, and solidarity statement [4].

Table 1. Definitions of Affective States and Engagement Levels

State	Definition
Learning-Centered	
Confusion	poor comprehension of material, attempts to resolve erroneous belief
Frustration	difficulty with the material and an inability to fully grasp the material
Anxious	nervousness, anxiety, negative self-efficacy, embarrassment
Contempt	annoyance and/or irritation with another person
Eureka	sudden realization about the material, a ha! moment
Curiosity	desire to acquire more knowledge or learn the material more deeply
Basic-Emotions	
Anger	negative affect toward material or person to an extreme degree
Fear	feelings of panic and/or extreme feelings of worry
Sadness	feelings of melancholy, beyond negative self-efficacy
Disgust	annoyance and/or irritation with the material and/or their abilities
Surprise	genuinely does not expect an outcomes or feedback
Happiness	satisfaction with performance, feelings of pleasure about the material
Engagement Level	
Disengagement	bored, uninterested in the topic being discussed
Socially Attending	attends to conversational conventions, only acknowledges tutor speech
Actively Attending	attends to content of the conversation, content-driven responses
Full Engagement	every mental resource is invested in the current topic, in a flow state

3 Results and Discussion

Proportion of Affective States that Occurred During Tutoring

We examined the proportion of occurrences of each of the affective states and engagement levels (see Table 2). A Repeated Measures ANOVA indicated that there was a significant difference in the various affective states experienced by the students, $F(11,308) = 51.11$, $Mse = .01$, $p < .001$, (partial eta-square) = .646. Bonferroni posthoc tests confirmed that confusion, anxious, and happiness were the common emotions that learners experienced during an expert tutoring session. Incidences of contempt, eureka, and curiosity were rare, and with the exception of happiness, all of the "basic" emotions almost never occurred. Incidences of frustration were less than confusion, anxious, and happiness but greater than the other emotions.

The affective profiles of the students during expert tutoring sessions is quite consistent with previous research on student emotions while they interacted with AutoTutor, a dialogue based ITS. D'Mello, Graesser, and colleagues have previously reported that confusion reigns supreme during deep learning activities of complex science topics, while incidences of contempt, eureka, curiosity, anger, and surprise are rare [22, 23, 24]. Confusion has been found to be a facilitator of learning [10, 24], where students are forced to think. Therefore, the recurrent appearances of confusion during the tutoring session demonstrate a high potential for students to learn. The high incidence of anxiety may be traced to students struggling with academic material who are seeking the help of a tutor in a high stakes learning situation. Anxiety includes negative feelings of self-efficacy, embarrassment, and being overwhelmed – states that resonate with difficulty in the subject matter. Students may enter the tutoring session with

Table 2. Descriptives of Student Affective States & Engagement Levels

Affective States

Non-Basic Emotions	Mean	Stdev	Basic Emotions	Mean	Stdev
Confusion	0.38	0.19	Anger	0.00	0.01
Frustration	0.05	0.04	Fear	0.00	0.00
Anxious	0.20	0.12	Sadness	0.00	0.00
Contempt	0.02	0.04	Disgust	0.01	0.04
Eureka	0.01	0.03	Happiness	0.29	0.22
Curiosity	0.01	0.02	Surprise	0.02	0.03
Sum	0.67		Sum	0.32	

Engagement Levels

Engagement	Mean	Stdev
Disengagement	0.00	0.01
Socially Attending	0.23	0.18
Actively Attending	0.77	0.18
Full Engagement	0.00	0.00

low self-efficacy from past experiences of failure with the material and may have feelings of embarrassment from having to seek out the help of a tutor.

Incidences of frustration were lower than what was expected from students in need of help from a tutor. The rate of occurrence of frustration documented in this study with expert tutors was consistent with earlier findings [22, 23, 24]. We suspect that the reduced rates of frustration may lie in the social display rules that people adhere to in expressive affect [31]. Social pressures may result in the disguising of negative emotions such as frustration, thus making it difficult for judges to detect this emotion. The perceived status difference between the student and the expert tutor, coupled with their lack of knowledge and heightened anxiety may supplement the desire to disguise frustration. Another important finding is that happiness was the only "basic" Ekman [18] emotion that reliably occurred. A paired sample t-test confirmed that the basic emotions occurred at a significantly lower rate than the other emotions, $t(28) = 4.305$, $p < .001$. This is yet another finding that challenges the significance of these "basic" emotions to learning and raises concern of the adequacy of basing an entire theory of emotions on the six "basic" emotions [18].

Levels of Student Engagement

An examination of student engagement levels indicated that learners were socially and actively attending for the majority of the tutoring session, $F(3, 84) = 167.405$, $Mse = .023$, $p < .001$, (partial eta-square) = .857. Experiences of complete disengagement (boredom) or full engagement (flow state) were rare, as could be expected due to the interactive nature of these tutoring sessions. The finding that students were actively attending for 77% of the tutoring session, while the remaining 23% of the time involved social attending, can be readily explained by considering the anatomy of a typical tutoring session. Social attending occurs when students cannot provide content-rich answers and only contribute by maintaining social conventions. Thus socially attending to the session is likely to occur for at least a portion of the session.

Given the nature of a human one-to-one tutoring session, actively attending appears to be the highest level of engagement that a student can achieve when interacting with both the tutor and the material. This bodes well for potential learning gains since students are focusing on the material and displaying their knowledge and misconceptions to the tutor.

Exploring Tutor Responses to Student Affective States

We investigated the extent to which expert tutors adapted their pedagogical and motivational strategies in response to the affective states of the students. In particular, if student experienced emotion E at turn t, we computed the probability that the tutor deployed dialogue move M at turn t+1, which is functionally equivalent to the conditional probability $\Pr\{M_{t+1}|E_t\}$. Our analyses focused on the most common tutor dialogue moves which were conversational okay, positive feedback, off-topic conversation, prompt, simplified problem, direct instruction, and comprehension gauging question [26]. For each frequent dialogue move, repeated measures ANOVAs were conducted to determine whether there were significant differences in deployment when considering student affective state.

The ANOVAs indicated that there were significant differences in the deployment of positive feedback ($F(1, 7) = 75.384$, $Mse = .691$, $p < .001$), off-topic conversation ($F(1,7) = 52.727$, $Mse = 1.312$, $p < .001$), direct instructions ($F(1,7) = 149.053$, $Mse = 2.945$, $p < .001$), and simplified problems ($F(1,7) = 37.115$, $Mse = .191$, $p < .001$) in response to the students' affect. However, prompting, conversational OK's and comprehension gauging questions were deployed independent of learners' affective states.

Table 3. Descriptives of Tutor Dialogue Move Given Student Affective State

Tutor Move at turn t+1	**Student Affective State at turn t**							
	Confusion		**Frustration**		**Anxious**		**Happiness**	
	M	*SD*	*M*	*SD*	*M*	*SD*	*M*	*SD*
Ok	0.149	0.099	0.187	0.156	0.17	0.129	0.104	0.123
Positive Feedback	0.225	0.082	0.051	0.06	0.2	0.136	0.112	0.106
Off-Topic	0.133	0.062	0.087	0.072	0.243	0.164	0.347	0.189
Prompt	0.051	0.057	0.006	0.017	0.037	0.035	0.001	0.002
Simplified Problem	0.153	0.081	0.048	0.063	0.051	0.048	0.057	0.042
Direct Instruction	0.383	0.112	0.213	0.107	0.3	0.138	0.316	0.111
Comprehension Gauging Question	0.029	0.018	0.026	0.059	0.063	0.061	0.059	0.044

Post-hoc tests on for the significant ANOVA's revealed the following patterns in the data. In the interest of brevity, we report the major findings only. It appears that expert tutors were more likely to provide positive feedback when students were confused than when frustrated. This finding is consistent with predictions by theories on impasse-driven learning [32]. These theories postulate that opportunities for learning

occur when students experience an impasse, defined as "when a student realizes that he or she lacks a complete understanding of a specific piece of knowledge" (p. 220). This is consistent with the affective state of confusion and explains why tutors provide more positive feedback to encourage students during this critical moment in learning. By scaffolding both the cognitive and affective state of the student, the tutor allows the impasse to occur but circumvents the shift to an unproductive affective state by maintaining a positive milieu.

While off-topic conversation during a tutoring session may seem to impede the effective construction of knowledge, it has been found to be an effective strategy and viewed as important by students [33, 34]. We found tutors to be more likely to utilize off-topic conversation when students were happy or anxious, which suggests that this dialogue move is being strategically deployed for two different purposes. Catt, Miller, and Schallenkamp [35] investigated the importance of instructor-student rapport and found it to be vital in maintaining good communication and producing greater learning gains. When students are happy tutors may take advantage of that situation and build a sense of trust and solidarity through conversation about day-to-day events. During states of anxiety, the tutor may use a temporary change in topic to relieve those negative feelings.

Expert tutors were found to provide students with a simplified problem more frequently when students were confused than during frustration or happiness. This finding is intuitively plausible and suggests that positive feedback coupled with a simplified problem seem to be the motivational and pedagogical strategies that expert tutors deploy in response to student confusion. Expert tutors are more likely to give direct instruction when students are anxious than if they are frustrated. The affective state of anxious includes several different feelings such as anxiety, nervousness, negative self-efficacy, worry, and being overwhelmed, and tutors may attribute these feelings to a lack of knowledge. Tutors might use direct instruction as a strategy to fill in these knowledge deficits in an attempt to alleviate such negative feelings instead of calling direct attention to the student's uncertainties.

4 Conclusion

The last decade has witnessed a surge in research that investigates the role of emotions in complex learning. We hope to have expanded this body of knowledge by identifying the affective states which students experience during expert tutoring sessions and the ways in which expert human tutors strategically respond to these states. Our findings suggest that with the exception of happiness, it is not the "basic" emotions that are prominent during learning but the affective states of confusion, frustration, and anxiety. Furthermore, we suspect that most of the experiences of happiness might in fact be states of contentness. Experiences of absolute happiness might be rarer than our data suggests. However, further research would be required to test this hypothesis.

The analysis of student engagement levels indicated that learners were socially attending and actively attending, but were rarely bored or in a state of flow; whereas learners experience both of these states while learning with ITSs [10, 25, 23, 24]. Taken together, these results highlight both the positive and negative aspects of

human and artificial tutoring sessions, because while it is beneficial to avoid boredom, flow is a state that is highly correlated with learning [10]. Of the pedagogical and motivational tutor dialogue moves that occur frequently [26], tutors only strategically deployed positive feedback, off-topic conversation, simplified problem, and direct instruction when responding to those affective states that occurred at significant levels. However, these results should be interpreted with a modicum of caution since the learners' affective states were not coded by the tutors themselves.

The next step is to use the findings from the present study to scaffold the development of ITSs that are capable of sensing and responding to student affect. Success in this endeavor depends upon adequately addressing three major issues: (1) what are the student affective states that occur during learning, (2) how can ITSs automatically detect these states, and (3) how should assessments of learner's affect influence pedagogical and motivational strategies of ITSs. The next step in this line of research is to begin constructing affect sensitive ITSs that are informed by the strategies of expert human tutors. Given the inextricable link between cognition and emotion, it is our position that modeling ITSs after human tutors will prove especially effective if emotions are taken into consideration. By implementing tutor pedagogical and motivational strategies in conjunction with affect sensitivity, ITSs will be able to produce heightened engagement, lower attrition, and increased self-efficacy, all factors that lead to positive learning gains.

Acknowledgments

This research was supported by a grant awarded by the U. S. Office of Naval Research (N00014-05-1-0241). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research.

References

1. Bloom, B.S.: The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher 13, 4–16 (1984)
2. Cohen, P.A., Kulik, J.A., Kulik, C.C.: Educational outcomes of tutoring: a meta-analysis of findings. American Educational Research Journal 19, 237–248 (1982)
3. Corbett, A., Anderson, J., Graesser, A., Koedinger, K., van Lehn, K.: Third generation computer tutors: Learn from or ignore human tutors? In: Proceedings of the 1999 Conference of Computer-Human Interaction, pp. 85–86. ACM Press, New York (1999)
4. Lepper, M.R., Woolverton, M.: The wisdom of practice: Lessons learned from highly effective tutors. In: Aronson, J. (ed.) Improving Academic Achievement: Impact of Psychological Factors on Education, pp. 135–158. Academic Press, San Diego (2002)
5. Chi, M.T.H., Siler, S.A., Jeong, H.: Can tutors monitor students' understanding accurately? Cognition and Instruction 22(3), 363–387 (2004)
6. del Soldato, T., du Boulay, B.: Implementation of motivational tactics in tutoring systems. Journal of Artificial Intelligence in Education 6(4), 337–378 (1995)
7. Csikszentmihalyi, M.: The domain of creativity. In: Runco, M., Albert, R. (eds.) Theories of Creativity, pp. 190–212. Sage Publications, Inc., Thousand Oaks (1990)

8. Kort, B., Reilly, R., Picard, R.W.: An affective model of interplay between emotions and learning: Reengineering educational pedagogy-building a learning companion. In: proceedings of ICALT 2001 (2001)
9. Stein, N.L., Levine, J.L.: Making sense out of emotion: The representation and use of goal-structured knowledge. In: Kessen, W., Ortony, A., Craik, F. (eds.) Memories, Thoughts, and Emotions: Essays in Honor of George Mandler, pp. 295–322. Lawrence Erlbaum Associates, Inc., England (1991)
10. Craig, S.D., Graesser, A.C., Sullins, J., Gholson, B.: Affect and learning: An exploratory look into the role of affect in learning. Journal of Educational Media 29, 241–250 (2004)
11. Snow, R.E., Corno, L., Jackson, D.: Individual differences in affective and conative functions. In: Berliner, D., Calfee, R. (eds.) Handbook of Educational Psychology, pp. 243–310. Macmillian Library Reference, New York (1996)
12. du Boulay, B., Luckin, R.: Modeling human teaching tactics and strategies for tutoring systems. International Journal of Artificial Intelligence in Education 12(3), 235–256 (2001)
13. Lajoie, S.P., Faremo, S., Wiseman, J.: Identifying human tutoring strategies for effective instruction in internal medicine. International Journal of Artificial Intelligence and Education 12(3), 293–309 (2001)
14. Palinscar, A., Brown, A.: Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. Cognition and Instruction 1, 117–175 (1984)
15. Lepper, M.R., Aspinwall, L.G., Mumme, D.L.: Self-perception and social perception processes in tutoring: Subtle social control strategies of expert tutors. In: Olson, J., Zanna, M. (eds.) Self-Inference Processes: The Ontario Symposium, pp. 217–237. Lawrence Earlbaum Associates, Inc., Hillsdale (1990)
16. Lepper, M.R., Woolverton, M., Mumme, D.L.: Motivational techniques of expert human tutors: Lessons for the design of computer-based tutors. In: Lajoie, S., Derry, S. (eds.) Computers as Cognitive Tools, pp. 75–105. Lawrence Earlbaum Associates, Inc., Hillsdale (1993)
17. Cromley, J.G., Azevedo, R.: What do reading tutors do? A naturalistic study of more and less experienced tutors in reading. Discourse Processes 40(2), 83–113 (2005)
18. Ekman, P.: Universal facial expressions in emotion. Studia Psychologica 15(2), 140–147 (1973)
19. Mandler, G.: Another theory of emotion claims too much and specifies too little. Current Psychology of Cognition 4(1), 84–87 (1984)
20. Scherer, K.: Appraisal considered as a process of multilevel sequential checking. In: Scherer, K., Schorr, A., Johnstone, T. (eds.) Appraisal Processes in Emotion: Theory, Methods, Research, pp. 92–120. Oxford University Press, New York (2001)
21. Meyer, D.K., Turner, J.C.: Re-conceptualizing emotion and motivation to learn in classroom contexts. Educational Psychology Review 18(4), 377–390 (2006)
22. D'Mello, S.K., Craig, S.D., Gholson, B., Franklin, S., Picard, R.W., Graesser, A.C.: Integrating affect sensors in an intelligent tutoring system. In: Affective Interactions: The Computer in the Affective Loop Workshop at 2005 International Conference on Intelligent User Interfaces, pp. 7–13. AMC Press, New York (2005)
23. Graesser, A.C., McDaniel, B., Chipman, P., Witherspoon, A., D'Mello, S., Gholson, B.: Detection Of Emotions During Learning With Autotutor. In: Proceedings of the 28<Superscript>th</Superscript> Annual Conference of the Cognitive Science Society, pp. 285–290. Erlbaum, Mahwah (2006)
24. Graesser, A.C., D'Mello, S.K., Chipman, P., King, B., McDaniel, B.: Exploring relationships between affect and learning with AutoTutor. In: Supplementary proceedings of the 13th International Conference on Artificial Intelligence in Education, pp. 16–23. IOS Press, Amsterdam (2007)

25. D'Mello, S.K., Picard, R.W., Graesser, A.C.: Towards an affect sensitive autotutor. Special Issue on Intelligent Education Systems – IEEE Intelligent Systems 22(4), 53–61 (2007)
26. Person, N., Lehman, B., Ozbun, R.: Pedagogical and motivational dialogue moves used by expert tutors. In: Presented at the 17th Annual Meeting of the Society for Text and Discourse, Glasgow, Scotland (2007)
27. Graesser, A.C., Person, N.K.: Question asking during tutoring. American Educational Research Journal 31(1), 104–137 (1994)
28. Graesser, A.C., Person, N.K., Magliano, J.P.: Collaborative dialogue patterns in naturalistic one-to-one tutoring. Applied Cognitive Psychology 9(6), 495–522 (1995)
29. Graesser, A.C., Wiemer-Hastings, P., Weimer-Hastings, K.: Constructing inferences and relations during text comprehension. In: Sanders, T., Schilperoord, J., Spooren, W. (eds.) Text Representation: Linguistic and Psycholinguistic Aspects, pp. 249–271. John Benjamins Publishing Company, Amsterdam (2001)
30. McNamara, D.S., Levinstein, I.B., Boothum, C.: iStart: Interactive strategy training for active reading and thinking. Behavior Research Methods, Instruments & Computers 36(2), 222–233 (2004)
31. Ekman, P., Friesen, W.V.: Nonverbal leakage and clues to deception. Psychiatry: Journal for the Study of Interpersonal Processes 32(1), 88–106 (1969)
32. VanLehn, K., Siler, S., Murray, C., Yamauchi, T., Baggett, W.: Why do only some events cause learning during human tutoring? Cognition and Instruction 21(3), 209–249 (2003)
33. Gruenwald, J.P., Ackerman, L.: A modified delphi approach for the development of student evaluations of faculty teaching. Journal of Marketing Education 8, 32–38 (1986)
34. Faranda, W.T., Clark, I.: Student observations of outstanding teaching: Implications for marketing educators. Journal of Marketing Education 26(3), 271–282 (2004)
35. Catt, S., Miller, D., Schallenkamp, K.: You are the key: Communicate for learning effectiveness. Education 127(3), 369–377 (2007)

Responding to Student Uncertainty During Computer Tutoring: An Experimental Evaluation

Kate Forbes-Riley, Diane Litman, and Mihai Rotaru

University of Pittsburgh, 3939 O'Hara St., Pittsburgh, PA 15260
forbesk,litman,mrotaru@cs.pitt.edu

Abstract. This paper evaluates dialogue-based student performance in a controlled experiment using versions of a tutoring system with and without automatic adaptation to the student affective state of uncertainty. Our performance metrics include correctness, uncertainty, and learning impasse severities, which are measured in a "test" dialogue after the tutoring treatment. Although these metrics did not significantly differ across conditions when considering all student answers in our test dialogue, we found significant differences in specific types of student answers, and these differences suggest that our uncertainty adaptation does have a positive benefit on student performance.

1 Introduction

In recent years, tutoring researchers have shown increasing interest in the interplay between student affect and learning (e.g. [1,2,3]). Numerous tutoring dialogue system researchers are investigating the hypothesis that student performance can be improved by automatically detecting and adapting to affective states (e.g., [4,5,6,7]). Student uncertainty is one state of primary interest due to its theorized relationship to correctness and learning. Researchers hypothesize that uncertainty can signal to the tutor that there is an opportunity for learning to occur, and that experiencing uncertainty can motivate a student to engage in learning (e.g. [6,8,9]). Moreover, correlational studies have shown a link between uncertainty and learning (e.g. [6]). However, few controlled experiments have investigated the performance impact of uncertainty adaptations in computer tutoring; most computer tutors respond based only on student correctness.

Based on this prior research, we hypothesized that responding to uncertainty - in addition to correctness - should improve student performance. We tested this hypothesis in a controlled experiment using adaptive and non-adaptive versions of a spoken dialogue tutoring system. Uncertainty and correctness were manually annotated in real-time by a human "Wizard". The experiment had three conditions. In the experimental condition, the system provided additional knowledge at places of uncertainty. In one control condition, the system did not provide this knowledge after uncertainty; in a second control condition the system provided this knowledge randomly.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 60–69, 2008.

Section 2 of this paper describes the experiment.[1] Section 3 presents a comparison of student performance metrics across condition. Section 4 discusses the implications of these results. Section 5 explains how we used these results to improve the design of a larger version of this experiment that is now underway.

2 The Experiment

In prior work we developed ITSPOKE (**I**ntelligent **T**utoring **SPOKE**n dialogue system) [11], a spoken dialogue tutor that is built on top of the Why2-Atlas text-based tutor [12] and tutors 5 qualitative physics problems. The spoken dialogues have a Question - Answer - Response format, implemented with a finite state dialogue manager. ITSPOKE responses (states) depend only on the correctness of the student answer (transitions between states). If the answer is correct, ITSPOKE moves on to the next question. ITSPOKE responses to incorrect answers take two forms: 1) For incorrect answers to easier questions, ITSPOKE provides the correct answer with a brief statement of reasoning. 2) For incorrect answers to harder questions, ITSPOKE engages the student in a **remediation subdialogue**, containing questions that walk the student through the more complex line of reasoning required for the correct answer.

2.1 Adaptive Wizard-of-Oz Spoken Dialogue Tutoring System

We've begun enhancing ITSPOKE to automatically respond to student affect[2] over and above correctness. For two reasons, we have initially targeted uncertainty. First, uncertainty occurred more than other affective states in our prior ITSPOKE dialogues [14]. Second, uncertainty is of primary interest to tutoring researchers due to its theorized relationship to learning (e.g. [6,8,9]). In [8], VanLehn et al. view uncertainty and incorrectness as signalling "learning impasses": opportunities for the student to learn the material about which s/he is uncertain or incorrect. From this view we derived a specific uncertainty adaptation hypothesis to test in a controlled experiment: Responding to uncertainty *in the same way as* incorrectness will improve student performance, by providing students with the knowledge needed to resolve their uncertainty impasses.

Implementing this adaptation involved changing the next state transitions in the finite state dialogue manager; instead of transitioning based only on the correctness of the answer, the transition is based on the answer's combined correctness and uncertainty value. More specifically, our uncertainty adaptation consisted of treating all uncertain+correct answers as if they were incorrect (note that uncertain+incorrect answers are already treated as incorrect).

[1] [10] describes the resulting publicly available Uncertainty Corpus in detail.

[2] We use "affect" to cover emotions and attitudes. Some argue for separating them, but some speech researchers find the narrow sense of "emotion" too restrictive since it excludes speech where emotion is not full-blown, including arousal and attitude [13]. Some tutoring researchers also combine emotion and attitude (e.g. [5,7]).

For an initial investigation into the impact of this adaptation on student performance, we implemented it in a Wizard of Oz (WOZ) version of ITSPOKE that tutors only one physics problem (as opposed to five). In this WOZ, a few system components are replaced by a human "wizard": The wizard performs speech recognition, correctness annotation, and uncertainty annotation, for each student answer. In this way, we tested the adaptation hypothesis without any potentially negative impact of automated versions of these tasks. Upon hearing each student answer, the Wizard annotates if it is correct or uncertain. These distinctions are binary: a "correct" answer may be partially or fully correct, and a "nonuncertain" answer may be certain or neutral for certainty.[3]

2.2 Experimental Design

The experiment had 3 conditions, designed to test whether our uncertainty adaptation improved student performance. For use in these 3 conditions, the dialogue manager was parameterized, so that it could adapt contingently on the student state of uncertain+correct as discussed above, or randomly, or not at all.

In the **experimental condition**, the dialogue manager adapted to uncertainty by treating all uncertain+correct student answers as incorrect.

In the **normal control condition,** the dialogue manager did not adapt to uncertainty (it was merely logged); it treated only incorrect answers as incorrect. In other words, this condition corresponds to the original system.

In the **random control condition**, the dialogue manager did not respond to uncertainty (it was merely logged), but it did treat a percentage of random correct answers as incorrect. This condition was included to control for the additional tutoring dialogue given to students in the experimental condition. The percentage was toggled to be statistically the same as the percentage of answers adapted to in the experimental condition (approximately 20%).

Note that the uncertainty adaptation for uncertain+correct answers was always preceded by the correctness feedback "Okay"; all other answers received the regular ITSPOKE correctness feedback. "Okay" was intended to mirror this feedback behavior. In particular, ITSPOKE is fully automated and uses somewhat vague correctness feedback (e.g. "Fine" and "Well..." for recognized correct and incorrect answers, respectively) because it may misrecognize an answer.

Figures 1-2 illustrate how system responses differ across conditions. In both figures, **STUDENT**$_1$ is *uncertain+correct*. In Figure 1 (experimental), **TUTOR**$_2$ says "Okay", then asks an additional remediation question to remediate the student's uncertainty. In Figure 2 (normal control),**TUTOR**$_2$ says "Good" and the remediation question is not asked. Section 3 discusses the tutor turn labels.

The experimental procedure was as follows. Each subject: 1) Read a small document of background physics material; 2) took a pretest of 7 fill in the blank

[3] In prior ITSPOKE corpora, this wizard displayed interannotator agreement of 0.85 Kappa on labeling binary correctness, and 0.62 Kappa on binary uncertainty [14]. Other possible approaches to labeling student affect include self-report (e.g. [4]).

TUTOR$_1$: What will the velocity of the object be a second after that (where the initial velocity is 9.8m/s and the acceleration is 9.8m/s^2)? *[Depth 1]*

STUDENT$_1$: Uh. 19.6 m/s? *[uncertain+correct]*

TUTOR$_2$: Okay. As we have seen, if a falling object has an acceleration of 9.8m/s^2, that means that its velocity changes by 9.8m/s every second. So if a second after it began falling the object's velocity is 9.8m/s, a second later its velocity will be 9.8m/s+9.8m/s=19.6m/s. So what will its velocity be a second after it is 19.6m/s? *[Depth 2+]*

STUDENT$_2$: Um. I'm thinking. 29.4 m/s. *[nonuncertain+correct]*

TUTOR$_3$: Right. So at every point in time during the fall of the man and his keys, how do their velocities compare with each other? *[Depth 1]*

Fig. 1. Corpus Excerpt: Experimental Condition

TUTOR$_1$: What will the velocity of the object be a second after that (where the initial velocity is 9.8m/s and the acceleration is 9.8m/s^2)? *[Depth 1]*

STUDENT$_1$: 19.6 m/s? *[uncertain+correct]*

TUTOR$_2$: Good. So at every point in time during the fall of the man and his keys, how do their velocities compare with each other? *[Depth 1]*

Fig. 2. Corresponding Corpus Excerpt: Normal Control Condition

questions; 3) used a web/voice interface to work through one physics *training* problem with a version of the WOZ (depending on condition); 4) took a posttest isomorphic to the pretest; 5) worked through a *test* problem isomorphic to the training problem with the *non-adaptive* WOZ (from the normal condition). Note that unlike the posttest, completing the test problem yielded a new dialogue.

Subjects were native English speakers who had not taken college physics. 60 subjects were randomly assigned to the 3 conditions (20 per condition), except conditions were gender-balanced. After the experiment, we found that 3 subjects in the experimental condition had no correct+uncertain answers and so never received the adaptation; 2 subjects in the random condition had no correct answers randomly selected for adaptation. These subjects were reclassified into the normal condition for our performance analysis.

3 Comparing Dialogue-Based Performance Metrics

We hypothesized that the training problem might be too short to yield significant differences between conditions in learning as measured by our pretest and posttest. This expectation was borne out; a two-way ANOVA with condition by repeated test measures design showed a significant main effect for test phase, ($F(1,57) = 33.919$, $p = 0.000$, $MSe = 0.032$), indicating students learned

overall, but there was no significant interaction effect between condition and test phase, indicating that amount of learning was not dependent on condition. One-way ANOVAs with post-hoc Tukey indicated no significant difference between conditions in raw (post-pre) or normalized ((post-pre)/(1-pre)) learning gain.

Thus, we used the test problem as an additional test of how the uncertainty adaptation in the training problem impacted student answers to the isomorphic questions in the test problem (where all students used the *non-adaptive* system, thereby receiving the same "test"). Below we analyze differences between conditions in dialogue-based performance metrics extracted from the test problem.

3.1 Comparing Impasse State Severities

In order to resolve a learning impasse, the student must first perceive that an impasse exists. Incorrectness and uncertainty differ in terms of this perception. Incorrectness simply indicates that the student has reached an impasse, while uncertainty - in a correct or incorrect answer - indicates that the student perceives s/he has reached an impasse. Based on this distinction, we associated each of our four answer combinations of uncertainty (**U**, **nonU**) and correctness (**I**, **C**) in the test problem with a scalar value from 3 to 0, as shown in Figure 3.

We hypothesized that these scalar values correspond to the severity of the student's current learning impasse state with respect to the test question, after receiving tutoring about the question in the training problem. Thus, 0 is a state in which the student is not experiencing an impasse, because s/he is correct and not uncertain about it. 3 is a state in which the student is experiencing the most severe type of impasse, because s/he is incorrect and not aware of it. 2 and 1 are states of lesser severity: the student is incorrect but aware that s/he might be, and the student is correct but uncertain about it, respectively.

Nominal State:	InonU	IU	CU	CnonU
Scalar State:	3	2	1	0
Severity Ranking:	most	less	least	none

Fig. 3. Different Impasse State Severities

After assigning a scalar state to each answer in the test problem, we computed a total and average impasse state severity per student. For example, suppose Figure 1 constituted our dataset for one student. The two student turns are labeled *uncertain+correct* and *nonuncertain+correct*, corresponding to scalar values 1 and 0, respectively. Thus the total = 1 (1+0), and the average = 0.5 (1/2).

We hypothesized that the experimental condition would show significantly lower total and average impasse severity in the test problem, because the uncertainty adaptation helped resolve more impasses during training. The "Means" columns in Table 1 show the means per condition. As expected, the experimental condition had lower total and average severity than the random condition, and random was lower than the normal condition. However, a one-way ANOVA with post-hoc Tukey showed no significant differences or trends ($p > 0.10$).

Table 1. Means and Correlations for Total and Average Impasse Severity

Metric	Means			Correlation (60)	
	Expmntl (17)	NormCtrl (25)	RandCtrl (18)	R	p
Tot. Impasse Severity	6.76	7.36	7.28	-0.38	0.003
Ave. Impasse Severity	0.38	0.42	0.41	-0.41	0.001

Despite this, we still hypothesized that lower impasse severities in the test problem are better, from a learning perspective. To support this, we computed a partial Pearson's correlation over all 60 students between both total and average impasse severity and posttest score, controlled for pretest score (pretest and posttest are significantly correlated in our data). The last two columns in Table 1 show the results. As shown, both total and average severity are significantly negatively correlated with learning, suggesting that lower impasse severities in the test problem are related to increased learning. We thus continue to use this hypothesis in our interpretation of results in the next sections.

3.2 Comparing Questions Originally Answered Correct+Uncertain

To further examine the impact of the uncertainty adaptation, we investigated student answers to those tutor questions that were asked in the training problem, answered as correct+uncertain, and then repeated in the test problem. In other words, we investigated student performance on the intended target of the uncertainty adaptation: the correct+uncertain (CU) answers. Note that these answers were all adapted to in the experimental condition, some were adapted to in the random condition, and none were adapted to in the normal condition.

The goal of our uncertainty adaptation was to increase correctness and decrease uncertainty in the test problem. In terms of these two dimensions combined, the goal was to decrease the frequency of the more severe nominal impasse states in Figure 3. Thus for each student's answers, we computed a total and percent of answers labeled with each (nominal) impasse severity (InonU, IU, CU, CnonU), as well as of correct (C) and nonuncertain (nonU) answers. For example, suppose both tutor questions in Figure 1 were originally answered CU in the training problem and are now repeated in the test problem. The totals then are: C=2, nonU=1, InonU=0, IU=0, CU=1, CnonU=1. The percents are: C=100%, nonU=50%, InonU=0%, IU=0%, CU=50%, CnonU=50%.

We hypothesized that the totals and percents in the experimental condition would be lower for InonU and IU, and higher for C, nonU, CU, and CnonU, because the uncertainty adaptation would have helped resolve impasses about these questions (or would have helped increase correctness and decrease uncertainty independently of each other). To test this hypothesis we ran a one-way ANOVA with post-hoc Tukey for each of the 12 metrics. Table 2 only shows metrics yielding significant differences or trends ($p<0.1$). The first column indicates these are answers to repeated questions originally answered CU (CU → ...).

Table 2. Means and Differences for Answers to Questions Originally Answered CU

Metric	Condition	Mean	Diff	p
Tot. CU → C	Expmntl	4.53	> NormCtrl	0.07
	NormCtrl	2.64		
	RandCtrl	5.11	> NormCtrl	0.01
Pct. CU → C	Expmntl	96.20%	> NormCtrl	0.09
	NormCtrl	76.50%		
	RandCtrl	91.06%		
Tot. CU → nonU	Expmntl	3.47		
	NormCtrl	2.32		
	RandCtrl	4.00	> NormCtrl	0.03
Tot. CU → CnonU	Expmntl	3.35		
	NormCtrl	2.20		
	RandCtrl	3.89	> NormCtrl	0.02

The remaining columns list the condition, its mean, the condition with which a difference is found, the direction of this difference (> or <), and its significance.

The first two results suggest that (significantly or as a trend) CU answers are more likely to stay correct in the test problem if they receive the uncertainty adaptation in the training problem. Put another way, CU answers are more likely to become incorrect during testing if the uncertainty adaptation is not received during training. The last two results suggest that the uncertainty adaptation reduces uncertainty in both the experimental and random conditions; however, only in the random condition do these results reach significance.

3.3 Comparing Answers at Different Dialogue Depths

We next tested whether the differences observed for answers to repeated questions generalized to all student answers in the test problem. However, one-way ANOVAs with post-hoc Tukey indicated no differences between conditions ($p > 0.10$) for any of the metrics (totals and percents for each nominal impasse state severity, for correct answers, and for uncertain answers).

We hypothesized that this lack of generalization might be due to the fact that student answers in remediation subdialogues can behave differently than those in the top-level dialogue, as we've shown in prior work [11]. As discussed in Section 2, the top-level dialogue is driven by correct answers to questions about the main problem topics, while a remediation subdialogue about a main topic is initiated by an incorrect answer to a top-level question. Thus as a final analysis, we distinguished these two answer types, which we refer to as "Depth 1" and "Depth 2+" answers. We computed the same metrics as above for each answer type and ran a one-way ANOVA with post-hoc Tukey for each metric. We found a trend for more Depth2+ answers to be CU in the experimental condition, as compared to the normal condition. More generally, the means for total and percent CU at Depth2+ were highest in the experimental condition, and lowest in the normal control condition. These results thus suggest that the uncertainty

adaptation helped increase correctness, but did not help decrease uncertainty, specifically regarding remediation questions. We hope to find firmer evidence of this when we repeat this type of analysis using data from the ongoing study discussed in Section 5.

4 Discussion and Related Work

Overall, our results in this paper are encouraging but inconclusive as to the benefit of our uncertainty adaptation on student performance. We hypothesize that two experimental design issues may have prevented larger differences between conditions. First, the training problem was likely too short. On average, it lasted 15 minutes, contained 20 student turns, and only 4 student turns on average received the adaptation in the experimental and random conditions. Second, the correctness feedback, "Okay", which preceded the uncertainty adaptation, was likely too vague. During the experiment, the wizard observed that uncertain+correct students were often confused by this feedback. We believe that the vagueness of "Okay" may have left these uncertain students ignorant as to whether their answer was correct. This vagueness may have been less noticeable to the random students, because roughly half of the time they were not uncertain when receiving the adaptation. This may explain why our analyses show little reduction in uncertainty in the experimental condition. Although resolving these issues should yield larger performance increases in the experimental condition, it still may not tease apart differences with the random condition. For one thing, *some* CU answers in the random condition receive the adaptation. A solution might be to only adapt to CnonU answers randomly; however, this too might benefit performance, by increasing the certainty of those answers (i.e., a CnonU answer may be neutral or certain). We assume it would not benefit performance to adapt to *every* correct answer, as this gives an identical response to incorrect and correct answers (except for correctness feedback).

Another complication is that it is not clear what is the best way to handle the fact that not all subjects in the two adaptive conditions actually received the adaptation. Although we moved into the normal condition the 5 subjects who didn't receive the adaptation, this is not necessarily the best solution because it can introduce sample bias; however, note that both before and after moving the subjects, the conditions had no significant difference in the total number or percent of correct answers in the test problem. Alternative approaches are also problematic. Removing the 5 subjects, as in [10], can also bias the samples. Retaining the subjects can yield ambiguous performance metrics. For example, for these 5 subjects, the metric %CU → CnonU would have to be set to 0 or left undefined because the denominator is 0 (# training CU), but if set to 0, then the value has another interpretation where this denominator is nonzero but the numerator is 0 (# training CU → testing CnonU). Note finally that if we use the Bonferroni correction, then the p-value required for a trend in Table 2 is 0.1/12 = 0.01. While this corrects for spurious results due to chance (type I errors), it can allow actual results to be overlooked (type II errors). We thus emphasize

that our results are exploratory and suggest specific hypotheses to be tested in our performance analysis of a larger experiment now underway (Section 5).

Determining when to adapt based on uncertainty is still an open question. To our knowledge only one other controlled experiment has tested uncertainty adaptations in spoken dialogue tutoring. In [5], Pon-Barry et al. implemented and evaluated two human tutor responses to uncertain answers (correct and incorrect) in the SCoT-DC tutor. In their "random" condition, the adaptations were used after all answers. They found significantly increased learning in this random condition as compared to a normal condition, but not in the experimental condition, where the adaptations were used only after uncertainty. Although most other work targeting uncertainty in the tutoring system community has involved correlational studies (e.g. [6]), there are other examples of adaptive tutoring systems developed or in development, which recognize affect and respond with various forms of empathy or politeness (e.g. [2,3,15]).

5 Conclusion and Current Directions

We presented one of the first experimental evaluations of student performance in a dialogue-based tutoring system that automatically adapts to student uncertainty. Our performance metrics include correctness, uncertainty, and learning impasse severity, which is a novel metric combining these two dimensions. These were measured in a test problem dialogue after the training dialogue. Though not conclusive, our results suggest that the uncertainty adaptation does have a positive benefit on student performance. In particular, correct+uncertain answers are more likely to become incorrect in the test problem if the uncertainty adaptation is not received during training, but only in the random condition are these answers also more likely to become nonuncertain. While learning impasse severity didn't differ significantly across conditions, it did significantly negatively correlate with student learning.

We hypothesized that two experimental design issues may have prevented more performance benefits of the uncertainty adaptation: short tutoring treatment and vague correctness feedback. We are now conducting a larger version of this experiment that resolves these issues. For this new experiment, we have implemented the uncertainty adaptation for all five ITSPOKE physics problems (rather than one); students are tutored for approximately an hour before taking the posttest, and thus are more likely to benefit from the uncertainty adaptation. In addition, we have replaced the vague "Okay" feedback with phrases that are clearly indicative of correctness (e.g. "That's correct").

Acknowledgments

This work was done as part of the Pittsburgh Science of Learning Center, funded by National Science Foundation (NSF) award #SBE-0354420. This work is also funded by NSF awards #0631930 & #0428472. We thank the ITSPOKE Group.

References

1. Workshop on Modeling and Scaffolding Affective Experiences to Impact Learning: Supplementary Proceedings of the 13th International Conference of Artificial Intelligence in Education (AIED), Marina Del Ray, CA, Online proceedings (July 2007), http://www.informatics.sussex.ac.uk/users/gr20/aied07/index.html
2. Wang, N., Johnson, W., Rizzo, P., Shaw, E., Mayer, R.: Experimental evaluation of polite interaction tactics for pedagogical agents. In: Proceedings of Intelligent User Interface Conference (IUI), pp. 12–19 (2005)
3. Hall, L., Woods, S., Sobral, D., Paiva, A., Dautenhahn, K., Wolke, D., Newall, L.: Designing empathic agents: Adults vs. kids. In: Proceedings of the Intelligent Tutoring Systems Conference (ITS), Maceio, Brazil, pp. 604–613 (2004)
4. McQuiggan, S., Mott, B., Lester, J.: Modeling self-efficacy in intelligent tutoring systems: An inductive approach. User Modeling and User-Adapted Interaction (UMUAI) 18(1-2), 81–123 (2008)
5. Pon-Barry, H., Schultz, K., Bratt, E.O., Clark, B., Peters, S.: Responding to student uncertainty in spoken tutorial dialogue systems. International Journal of Artificial Intelligence in Education 16, 171–194 (2006)
6. Craig, S., Graesser, A., Sullins, J., Gholson, B.: Affect and learning: an exploratory look into the role of affect in learning with AutoTutor. Journal of Educational Media 29(3), 241–250 (2004)
7. Bhatt, K., Evens, M., Argamon, S.: Hedged responses and expressions of affect in human/human and human/computer tutorial interactions. In: Proceedings of Cognitive Science (CogSci), Chicago, USA, pp. 114–119 (2004)
8. VanLehn, K., Siler, S., Murray, C.: Why do only some events cause learning during human tutoring? Cognition and Instruction 21(3), 209–249 (2003)
9. Kort, B., Reilly, R., Picard, R.: An affective model of interplay between emotions and learning: Reengineering educational pedagogy-building a learning companion. In: Okamoto, T., Hartley, R., Kinshuk, J., Klus, P. (eds.) Proceedings IEEE International Conference on Advanced Learning Technology: Issues, Achievements and Challenges, Madison, WI, pp. 43–48 (2001)
10. Forbes-Riley, K., Litman, D., Silliman, S., Purandare, A.: Uncertainty corpus: Resource to study user affect in complex spoken dialogue systems. In: Proceedings 6th Language Resources and Evaluation Conference (LREC), Marrakech, Morocco (May 2008)
11. Forbes-Riley, K., Rotaru, M., Litman, D.: The relative impact of student affect on performance models in a spoken dialogue tutoring system. User Modeling and User-Adapted Interaction 18(1-2), 11–43 (2008)
12. VanLehn, K., Jordan, P.W., Rosé, C.P., Bhembe, D., Böttner, M., Gaydos, A., Makatchev, M., Pappuswamy, U., Ringenberg, M., Roque, A., Siler, S., Srivastava, R., Wilson, R.: The architecture of Why2-Atlas: A coach for qualitative physics essay writing. In: Proceedings of Intelligent Tutoring Systems (2002)
13. Cowie, R., Cornelius, R.R.: Describing the emotional states that are expressed in speech. Speech Communication 40(1-2), 5–32 (2003)
14. Forbes-Riley, K., Litman, D.: Analyzing dependencies between student certainness states and tutor responses in a spoken dialogue corpus. In: Dybkjaer, L., Minker, W. (eds.) Recent Trends in Discourse and Dialogue, pp. 275–304. Springer (2008)
15. Burleson, W., Picard, R.: Affective agents: Sustaining motivation to learn through failure and a state of stuck. In: Social and Emotional Intelligence in Learning Environments Workshop at the Intelligent Tutoring Systems Conference (ITS), Maceio, Brazil (2004)

How Does an Intelligent Learning Environment with Novel Design Affect the Students' Learning Results?

Marina Lepp

University of Tartu, Institute of Computer Science
J. Liivi 2 – 306, 50409, Tartu, Estonia
marina.lepp@ut.ee

Abstract. We present an intelligent learning environment, T-algebra, for step-by-step solving of algebra problems using a novel design of step dialogue, which combines two known approaches: conversion by rules and entering the result. Each solution step in T-algebra consists of three stages: selection of the transformation rule, marking the parts of expression, entering the result of the operation. The designed dialogue enables the student to make the same mistakes as on paper and to receive understandable feedback about mistakes. The evaluation demonstrated that even a brief use of T-algebra affects the results of learning. The students who used T-algebra did better on consecutive paper test than the students who did not use T-algebra. Furthermore, T-algebra tends to affect specific error types, i.e., after using T-algebra the students make fewer mistakes of certain type on paper as well.

1 Introduction

Learning environments for step-by-step solving of expression manipulation problems (inc. linear equations) have been designed for a long time. Even the earlier environments could be divided into two groups according to the type of dialogue they use: rule-based or command-based environments (EXPRESSIONS [21], ALGREBRALAND [4], DISSOLVE [14]) and input-based environments (BUGGY/DEBUGGY system [5, 6], LMS [19], EMMA [16], Algebra tutor [2]). This division is still applicable today.

Rule-based environments (such as MathXpert [3], AlgeBrain [1], Cognitive algebra tutor [7], E-tutor: An Equation Solving Tutor [17]) are based on the principle that the student selects the transformation rule and in some cases a part of the expression; the transformation itself is made by the computer. In such environments, the student learns and practices the solution algorithm, but the learning of performing algorithm steps is passive, because the computer performs more work than the user. In addition, the student is not given the possibility to make certain mistakes; many typical mistakes are simply impossible.

Input-based environments (like Aplusix [13], Treefrog [20]) use paper-and-pencil-like dialogue design where a transformation step consists mainly of entering the next line. The student has the possibility to perform whatever steps and as much as he/she wants in one step and to make arbitrary mistakes. Yet such programs usually do not handle the solution algorithms of different types of problems and do not provide a precise diagnosis of the errors made.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 70–79, 2008.

This article presents the intelligent learning environment T-algebra with a novel design of step dialogue [12, 15]. The design is novel, because it combines two known approaches: rule-based and input-based environments (conversion by rules is supplemented by entering the result). By choosing the rule, the appropriate parts of the expression and entering the result of the operation, the student can learn the algorithms and their steps and make mistakes in the same way as on paper. The proposed dialogue enables the program to check the knowledge and skills of the student, to diagnose errors and to offer feedback.

Before distribution of T-algebra to all Estonian schools, we organized an experiment to clarify how the program affects the learning results. 126 students of 7th grade (about 13 years old) from four different Estonian schools participated in the experiment. Pre-test-posttest control-group design [8] was used in research. Pre-test and post-test were solved on paper in both (experimental and control) groups. Between these tests the control group received traditional instruction using paper and pencil, while the experimental group received experimental instruction using T-algebra. This experiment compared T-algebra with the paper-and-pencil method, not with other learning environments.

The second part of this article presents an overview of the T-algebra environment and error diagnosis principles in T-algebra. The third section describes the conditions of the experiment. The fourth part summarizes the pre-test and post-test results of the experimental and control groups, compares them, presents interesting findings of this comparison and answers the question: How does an interactive learning environment with novel design affect the students' learning results? The article also examines the types of errors in post-test in experimental and control groups.

2 Description of T-algebra Environment

T-algebra is an interactive learning environment for step-by-step solving of school algebra problems, including linear equations. Each solution step in T-algebra consists of

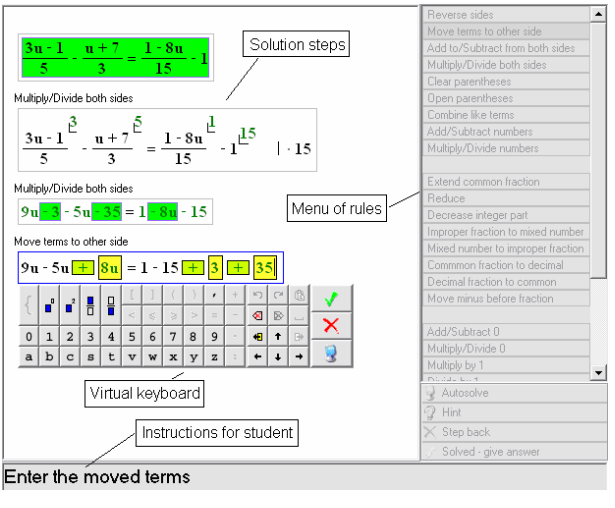


Fig. 1. The problem-solution window of the T-algebra program

three stages: selection of the transformation rule, marking the parts of expression, entering the result of the operation. The presented scheme improves the ability of the program to check the student's solutions, respond to the errors made by the student and give advice when the student is at loss. The program monitors whether the student works according to the algorithm, and supports it with the respective dialogue, diagnoses transformation errors, offers advice and, if necessary, performs the next stage of the step by itself.

The problem solution window of T-algebra is shown on Figure 1. The main part of the window contains solution steps and a virtual keyboard that can be used for active input. On the right side is the menu of possible actions. The lower part includes instructions for the student in this particular situation.

Figure 2 demonstrates performance of one step in the program (applying the rule *Move terms to other side*).

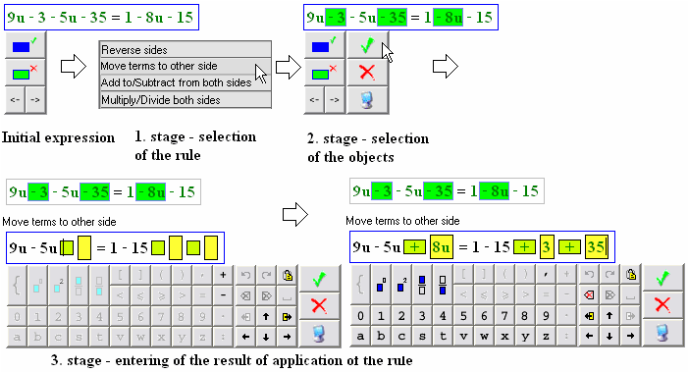


Fig. 2. Example of one step in T-algebra

In T-algebra, the student is left the possibility to make mistakes at all three stages of the step. If a mistake can be made then T-algebra can respond to it as well. First, the student could err in choosing the rule. If the application of the selected rule is impossible, the program does not immediately inform the student about the error, because the student will not find suitable objects for applying this rule or will make an error by choosing unsuitable objects. This gives the student a chance to correct the error without assistance.

Second, the student can make mistakes in marking the parts of expression. The program performs a number of different checks, like syntactical correctness, compatibility, position, etc. When wrong parts have been selected, the program does not permit to continue.

The input stage has the largest selection of potential mistakes, because the student must apply the rule for the marked parts and enter the result. The program tries to determine whether the student has made a standard error, which occurs often in student solutions (for example, not changing the sign of a moved term is a common mistake made by Estonian students). If the mistake is in the set of standard mistakes (some studies have been conducted to collect the students' mistakes made on paper [9, 11, 18]) then T-algebra is able to diagnose it and offer an appropriate error message (Fig. 3). Besides standard mistakes, T-algebra can also check the non-equivalence of equations.

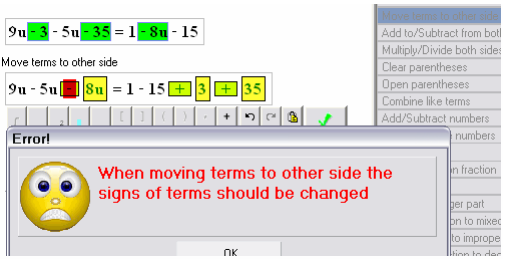


Fig. 3. Error message displayed when entering the result

3 Description of Experiment

The study was carried out in the winter 2007. Seven classes (126 students) of 7th grade (13 years old) from four different Estonian schools participated in the experiment. Classes from two schools, where there was more than one 7th grade class, were divided into experimental classes and control classes. The remaining two schools participated as experimental classes. After the division, we had 2 control classes and 5 experimental classes. Classes from the schools with more than one 7th grade class were taught by the same teacher.

The topic of linear equations was chosen for the experiment and the experiment began when the topic had been explained and practiced in the schools. The experiment consisted of four 45-minute sessions. In the first session, the students solved a pre-test on paper. In the next two sessions, the students practiced solving the problems of the same topic (linear equations). The experimental group practiced solving these problems with T-algebra, while the control group practiced solving exactly the same problems using traditional instruction technology – paper and pencil. In the last session, the students solved a post-test on paper. Teachers had exact instructions what, when and how to do and the same materials (pre-test, problems for practicing and post-test) were prepared for all teachers.

The pre-test was solved in both groups using paper and pencil. During the pre-test, the students could not use any assistance materials. The test contained 17 problems (6 types of problems) and it was possible to earn 39 points in total. Several examples of the problems (with maximum points) are listed below:

- Check if number *5* is solution of equation $10-2(3y-1)=2y-7(y-1)$ (3 p.);
- Reverse equation sides: $24=3y+7$ (1 p.);
- Divide equation sides by variable coefficient: $1{,}3n=-3{,}9$ (1 p.);
- Multiply both sides of the equation by common denominator: $\frac{x+3}{9}-\frac{3x+1}{12}=\frac{2x+5}{4}+\frac{x-5}{6}$ (2 p.);
- Move all variable terms to the left side and all constant terms to the right side and then combine like terms: $5-7x+4-x=3x-9+x+11$ (2 p.);
- Solve an equation: $5y-2(3y+4)=7-4y$ (5 p.).

The students were graded according to the following scale:

5: 35.5 – 39 points (91 - 100 %)
4: 27.5 – 35 points (71 - 90 %)
3: 19.5 – 27 points (51 - 70 %)
2: 11.5 – 19 points (31 - 50 %)
1: 0.0 – 11 points (0 - 30 %)

During the next two sessions (mathematics lessons), the experimental group practiced solving similar problems using T-algebra in computer class. The practice took place immediately after the pre-test in the next mathematics lesson and linear equations were not taught in the ordinary class between pre- and post-tests. The students had seen and tried T-algebra before when learning other topics, so the teachers did not have to explain the environment to the students. After the second lesson the students saved their solutions, the teachers collected them and sent to us for examination.

While the experimental group practiced solving in T-algebra, the control group solved exactly the same problems using paper and pencil. During the sessions the students solved the problems in their notebook and one of them wrote the solution on the blackboard. The teacher highlighted and corrected mistakes in the solutions on the blackboard, but did not explain anything new and did not correct solutions in the notebooks.

During the fourth consecutive session, both groups solved a post-test using paper and pencil. The arrangement of the post-test was the same as in pre-test and similar types of problems were used. Again, the students could not use any assistance materials, least of all the corrected pre-test.

After the experiment we collected the papers of the pre- and post-tests and the files with solutions in T-algebra for analysis.

4 Results of Experiment

After an analysis of papers and files, the students who had missed at least one session were excluded and the work of 115 students remained. The tests were analyzed further and the students whose pre-test result was less than 11 points were excluded, because in the preconditions of the experiment we assumed that the topic had been taught to the students, i.e., the students should be able to score at least 30 % of the points. We wanted to evaluate how T-algebra affects practicing after the topic has been explained by the teacher, not how it influences learning new material. While all other students had some basic knowledge about linear equations, these students (who scored under 30 %) did not. The work of 106 students remained after this step; 76 of them had participated in the experimental group and 30 in the control. Table 1 shows the results (average number of points) of pre- and post-test in both (experimental and control) groups. As we can see, the average number of points in pre-test is almost equal in experimental and control groups. The difference is not statistically significant (unpaired *t*-test $t = 0.0368$, $p = 0.97$) and the groups can be considered as equal.

Table 1. Results (average number of points) of the tests

	Experimental	**Control**
Pre-test	29.4	29.3
Post-test	31.3	29.9

Table 1 indicates that the knowledge of students from experimental group is statistically significantly improved (paired *t*-test $t = 3.571$, $p < 0.01$), but no statistically significant difference (improvement) can be found in the points earned by the control group (paired *t*-test $t = 1.2024$, $p > 0.05$). Effect size (using Cohen's *d*) is 0.179. This implies that even a brief use (2 lessons) of T-algebra affects the results of learning.

Table 2 shows the percentages of students from the experimental group with different grades in pre- and post-tests. As we can see, the percentage of students with the highest grade has grown. The percentage of students with grade 4 remained the same while the percentage of students with low grades (3 and 2) has decreased.

Table 2. Division of students (from the experimental group) by pre- and post-test grades

	Students with grade 5	**Students with grade 4**	**Students with grade 3**	**Students with grade 2**
Pre-test	25 %	38 %	24 %	13 %
Post-test	39.5 %	38 %	14.5 %	8 %

Checking the post-test of the experimental group, we noticed that one experimental student emulated the writing style of T-algebra. The operation *Multiply/Divide both sides* has a slightly different appearance in Estonian textbooks and in students notebooks from that used in T-algebra. On paper, the students perform this operation in two rows and the result is a solution like the one on the left on Figure 4. However, the application of this rule in T-algebra is written in three rows (Fig. 1). The mentioned student was not able to solve problems of the type *Multiply both sides* in the pre-test. In the post-test he solved all five problems of this type successfully, but he always wrote the solution in three rows as illustrated on the right side on Figure 4.

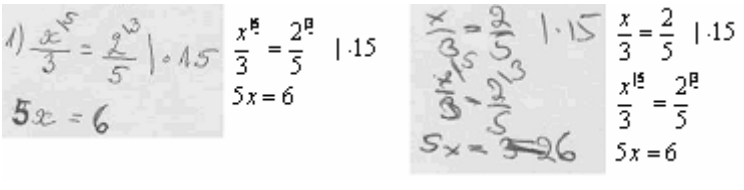


Fig. 4. Multiplication of both sides of equation in two rows (in textbooks, on paper) (left) and in three rows (emulating T-algebra) (right)

This picture was very unusual on paper, so we drew the conclusion that even two hours with T-algebra could affect the students' writing style. Naturally, this assumption still needs to be confirmed or refuted through future experiments.

Now we could look in more detail at the mistakes made in pre- and post-tests in the experimental and control groups. Table 3 shows the percentage of students in each group who made a specific mistake in the pre-test or in the post-test while the last columns show the percentage of the students who repeated the mistake they had made in the pre-test in the post-test as well. Many different kinds of mistakes were made, but Table 3 presents only the mistakes that were made in the pre-test by more than ten percent of the students in both groups (experimental and control). This restriction was

introduced to enable comparison of mistakes in pre- and post-tests (it would be very hard to distinguish the influence of T-algebra in case of mistakes made only by a few students in one or another group). Table 3 does not reflect whether the student made this mistake more than once in one test.

Table 3. Mistakes made in pre- and post-tests in experimental and control groups

No	Nature of mistake	Example of mistake	Pre-test		Post-test		Recurrence in post-test	
			experimental	control	experimental	control	experimental	control
1	Minus sign before fraction is taken into account only at first term	$\frac{4y}{3} - \frac{2y+3}{5} = \frac{4}{15} \quad \cdot 15$ $20y - 6y + 9 = 4$	55	56	29	46	52	82
2	Arithmetic mistake in combining and in evaluating	$7s - 9s = 2 - 12$ $-2s = -24$	46	50	21	33	45	67
3	In the problem *Check if number is a solution* the equation is solved		40	30	19	10	48	33
4	In the problem *Reverse sides* all variable terms are moved to the left side and all constant terms to the right side		32	23	8	10	24	43
5	Minus sign before parentheses is taken into account only at first term	$9 - 2(3y - 1) = 3 - 2y$ $9 - 6y - 2 = 3 - 2y$	22	46	10	40	47	85
6	Mistake in sign in dividing	$-4y = -8 \quad : (-4)$ $y = -2$	21	20	7	7	31	33
7	Arithmetic mistake in dividing	$-0.3y = -1.2 \quad : (-0.3)$ $y = 0.4$	17	14	9	7	53	50
8	Sign is not changed when moving to other side	$8u - 5u + 15 = 4u + 6$ $8u - 5u - 4u = 15 + 6$	16	20	8	10	50	50
9	Whole number is not multiplied	$\frac{x}{5} + 3 = \frac{7}{10} \quad \cdot 10$ $2x + 3 = 7$	15	23	3	10	18	43

As we assumed, T-algebra affects some error types. We can see that the students from the experimental group made fewer mistakes in the sign of second term (mistakes number 1 and 5) in the post-test. The same was observed in the earlier experiments [10] and we believe that showing the error message immediately and directing attention to the mistake and its location (box) is the cause of that. However, T-algebra

does not affect mistakes in sign, which are not connected to the second term, such as mistakes in sign in dividing and moving to other side (mistakes 6 and 8).

A decrease in the number of students from the experimental group who made the mistake number 9 was also predictable. The same was noticed in the earlier experiment [10]. The students working on paper often forget to multiply a number, which is not fraction, but T-algebra does not allow proceeding with such solution and notifies that all terms should be multiplied. This causes the reduction of this mistake in the post-test.

T-algebra can also affect learning of algorithms. The students from the experimental group made the mistake number 4 (in the problem *Reverse sides*) less frequently than the students from the control group. Experimental students made the mistake number 3 (in the problem *Check if number is a solution*) in the post-test more often than the control students, because teachers request checking the solution of linear equations when solving on paper (sixth type of problems – solve an equation). Therefore, the students solving equation on paper also practice checking the solution. This checking stage is omitted in T-algebra, because the program does not permit incorrect solutions. Consequently, the percentage of the students from the control group who made this mistake decreased, because they had more practice with this type than the experimental students.

It is hard to say whether T-algebra affects arithmetic mistakes or not. The students from the experimental group made the mistake number 2 less frequently than the students from the control group, but the frequency of the mistake number 7 was equal (and even slightly higher in the experimental group). We assumed that T-algebra does not affect arithmetic mistakes and we hope future experiments will explain the decrease in the frequency of the mistake number 2.

5 Conclusions

We have combined two known approaches for step-by-step solving of algebra problems and have designed a three-stage dialogue in T-algebra intelligent learning environment. We have succeeded in creating such rule dialogue in T-algebra that gives the student the possibility to learn both the solution algorithms and their steps, to make the same mistakes as on paper, and enables the program to check the knowledge and skills of the student, understand the student's mistakes, offer feedback and give advice. We have conducted the experiment to answer the question: How do T-algebra environment and its novel design affect the students' learning results? The experiment comprised pre- and post-tests on paper and practice with or without T-algebra.

As we saw in the post-test, the students from the experimental group did better than the students from the control group (the average pre-test score was almost equal in experimental and control groups). This shows that even a brief use (2 lessons) of T-algebra affects the results of learning. We also saw a progress of the students from the experimental group to a higher score in the post-test.

The experiment showed that even two hours with T-algebra could affect the students' writing style on paper. However, this observation needs further experiments for confirmation or refutation.

Finally, we saw that T-algebra could affect some error types, i.e., the students from experimental group made fewer mistakes of certain types (like mistakes in the sign of second term). However, this short period of use of T-algebra gave strange results for arithmetic mistakes; we hope to find an explanation for the change in these mistakes from a long-term experiment.

Obviously our decisions and ideas need some years of practical classroom trials before they can be finally confirmed. Starting from the school year 2006-2007, tens of teachers in Estonian schools use T-algebra for practice. The results of the school trials and teacher experiences will contribute to and support further development of T-algebra.

The conducted experiment examined the influence of the T-algebra environment on students and did not compare T-algebra with other environments. It would be very interesting to organize an experiment to compare whether the novel design of T-algebra produces better results than, for example, Aplusix [13] or MathXpert [3] environment.

Acknowledgements

T-algebra learning environment was developed as a project financed by the 'Tiger Leap' computerization programme for Estonian schools. The authors of T-algebra program were also supported by the Estonian Science Foundation under grant No. 7180 and by the targeted financing project No. SF0182712s06 "The methods, environments, and applications for solving large and complex computational problems"

We would like to thank the mathematics teachers Sirje Pihlap, Urve Olesk, Eha Kuld and Janika Kaljula and their students for participating in the experiment.

References

1. Alpert, S.R., Singley, M.K., Fairweather, P.G.: Deploying Intelligent Tutors on the Web: An Architecture and an Example. International Journal of Artificial Intelligence in Education 10(2), 183–197 (1999)
2. Anderson, J.R., Boyle, C.F., Corbett, A., Lewis, M.W.: Cognitive modelling and intelligent tutoring. Artificial Intelligence 42, 7–49 (1990)
3. Beeson, M.: MathXpert: un logiciel pour aider les élèves à apprendre les mathématiques par l'action. Sciences et Techniques Educatives 9(1–2) (2002)
4. Brown, J.S.: Process versus Product: A perspective on tools for communal and informal electronic learning. Journal of Educational Computing Research 1, 179–201 (1985)
5. Brown, J.S., Burton, R.R.: Diagnostic models for procedural bugs in basic mathematical skills. Cognitive Science 2, 155–192 (1978)
6. Burton, R.R.: Diagnosing bugs in a simple procedural skill. In: Intelligent Tutoring Systems, pp. 157–183. Academic Press (1982)
7. Cognitive Tutor by Carnegie Learning, Inc., `http://www.carnegielearning.com/`
8. Gall, M.D., Borg, W.R., Gall, J.P.: Educational research: an introduction, 6th edn. Longman Publishers, USA (1996)

9. Hall, R.: An Analysis of Errors Made in the Solution of Simple Linear Equations. Philosophy of Mathematics Education Journal 15 (2002)
10. Issakova, M.: Comparison of student errors made during linear equation solving on paper and in interactive learning environment. In: Dresden International Symposium on Technology and its Integration into Mathematics Education 2006, Dresden, Germany, 20p (2006)
11. Issakova, M.: Possible Mistakes during Linear Equation Solving on Paper and In T-algebra Environment. In: Proceedings of the 7th International Conference on Technology in Mathematics Teaching, Bristol, vol. 1, pp. 250–258 (2005)
12. Issakova, M., Lepp, D., Prank, R.: T-algebra: Adding Input Stage To Rule-Based Interface For Expression Manipulation. International Journal for Technology in Mathematics Education 13(2), 89–96 (2006)
13. Nicaud, J., Bouhineau, D., Chaachoua, H.: Mixing microworld and cas features in building computer systems that help students learn algebra. International Journal of Computers for Mathematical Learning 5(2), 169–211 (2004)
14. Oliver, J., Zukerman, I.: DISSOLVE: A system for the generation of human oriented solutions to algebraic equations. In: AI 1988. LNCS, vol. 406, pp. 91–107. Springer (1988)
15. Prank, R., Issakova, M., Lepp, D., Tõnisson, E., Vaiksaar, V.: Integrating rule-based and input-based approaches for better error diagnosis in expression manipulation tasks. In: Li, S., Wang, D., Zhang, J. (eds.) Symbolic Computation and Education, pp. 174–191. World Scientific Publishing Co., Singapore (2007)
16. Quigley, M.: A Simple Algebra Tutor. Journal of Artificial Intelligence in Education 1(1), 41–52 (1989)
17. Razzaq, L.M., Heffernan, N.T.: Tutorial Dialog in an Equation Solving Intelligent Tutoring System. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 851–853. Springer, Heidelberg (2004)
18. Sleeman, D.: An Attempt to Understand Students' Understanding of Basic Algebra. Cognitive Science 8, 413–437 (1984)
19. Sleeman, D., Smith, M.J.: Modelling student's problem solving. Artificial Intelligence 16(2), 171–187 (1981)
20. Strickland, P., Al-Jumeily, D.: A Computer Algebra System for improving student's manipulation skills in Algebra. The International Journal of Computer Algebra in Mathematics Education 6(1), 17–24 (1999)
21. Thompson, P., Thompson, A.: Computer presentations of structure in algebra. In: Proceedings of the Eleventh Annual Meeting of the International Group for the Psychology of Mathematics Education, vol. 1, pp. 248–254 (1987)

Learning Linked Lists: Experiments with the iList System

Davide Fossati[1], Barbara Di Eugenio[1], Christopher Brown[2], and Stellan Ohlsson[3]

[1] Department of Computer Science, University of Illinois, Chicago, IL, USA
[2] Department of Computer Science, U.S. Naval Academy, Annapolis, MD, USA
[3] Department of Psychology, University of Illinois, Chicago, IL, USA
dfossa1@uic.edu, bdieugen@cs.uic.edu, wcbrown@usna.edu, stellan@uic.edu

Abstract. This paper presents the first experiments with an Intelligent Tutoring System in the domain of linked lists, a fundamental topic in Computer Science. The system has been deployed in an introductory college-level Computer Science class, and engendered significant learning gains. A constraint-based approach has been adopted in the design and implementation of the system. We describe the system architecture, its current functionalities, and the future directions of its development.

1 Introduction

In this paper, we present the first version of iList, an Intelligent Tutoring System (ITS) in the domain of basic data structures and algorithms in Computer Science (CS), and its evaluation. Among the innovative features of our work are: the domain itself, and specifically, our focus on linked lists, due to pedagogical tenets for CS; the choice of constraint-based modeling as the basis for our ITS; and the structure of our graphical interface, itself partly due to the pedagogy of CS. This work is situated within our larger research program, whose main goal is, similarly to others [1,2,3,4], to better understand why human tutoring is effective, and to discover computational models of effective tutoring that can be implemented in ITSs. We are developing a second version of the ITS we present here, based on our data collection and analysis in this CS domain.

Computer Science as a Domain. In recent years, interest in CS among college students in the US has dropped dramatically. However, CS and Information Technology are of enormous strategic interest, and are projected to foster vast job growth in the next few years [5]. We believe that by supporting CS education in its core we can have the largest impact on reversing the trend of students' disinterest, and on attracting women and minorities. Our belief is grounded in the observation that the rate of attrition is highest at the earliest phases of undergraduate CS curricula. This is due in part to students' difficulty with mastering basic concepts [6], which require a deep understanding of static structures and the dynamic procedures used to manipulate them [7]. These concepts

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 80–89, 2008.

require a high level of abstraction, and the ability to move seamlessly among multiple representations, such as text, pictures, pseudo-code, and real code in a specific programming language. Thus, we believe that the availability of an ITS for basic CS would be of great benefit to both teachers and students. Such an ITS does not exist yet. This is surprising, since CS education is an area of active research, and ITSs are obviously software systems. Although ITSs on CS topics do exist, to our knowledge, only two of them tutor on the foundations. ADIS [8] tutors on basic data structures, but its emphasis is on visualization, and it appears to have been more of a proof of concept than a working system. ProPL [9] helps novices design their programs, by stressing problem solving and design skills. The other ITSs for CS focus on a diverse range of topics, from basic literacy as in AutoTutor [10], to teaching programming languages such as Lisp [11], C++ [12], and Java [13], to topics such as search algorithms used in Artificial Intelligence [14]. Of particular interest to us is the database suite of tutors composed by SQL-Tutor, NORMIT, KERMIT, and EER-Tutor [15]. These ITSs are built via constraint-based modeling, the same paradigm we chose for the development of our system.

Constraint-Based Modeling. Our system is based on a design paradigm known as *constraint-based* modeling. Originally developed from a cognitive theory of how people might learn from performance errors [16,17], constraint-based modeling has grown into a methodology used to build full-fledged ITSs, and an alternative to the model tracing approach adopted by many ITSs. In a constraint-based system, domain knowledge is modeled with a set of *constraints*, logic units composed of a *relevance condition* and a *satisfaction condition*. A constraint is irrelevant when the relevance condition is not satisfied; it is satisfied when both relevance and satisfaction conditions are satisfied; it is violated when the relevance condition is satisfied but the satisfaction condition is not.

In the context of tutoring, constraints are matched against student solutions. Satisfied constraints correspond to knowledge that students have acquired, whereas violated constraints correspond to gaps or incorrect knowledge. An important feature is that there is no need for an explicit model of students' mistakes, as opposed to buggy rules in model tracing. Errors are implicitly specified as the possible ways in which constraints can be violated. This property simplifies the difficult and time consuming task of knowledge modeling in an ITS.

There is currently a heated debate on whether constraint-based modeling is more or less appropriate than model tracing for building ITSs [18,19,20]. The application of the constraint-based paradigm to a new domain can contribute to a better understanding of this issue.

Empirical Grounding. Our goal is not just to develop an ITS for CS, but to endow it with a dialogue interface that can provide more sophisticated feedback, that can help improve students' learning [21,22]. To accomplish this goal, we are conducting an extensive tutoring dialogue collection in the data structures domain. We already collected 54 tutoring sessions, transcribed the video-recordings, and started annotating them. More details on our data collection and preliminary analysis can be found in [23,24]. The findings of future data analysis will

guide further development of our system. The ITS we describe in this paper is in fact a baseline to which we will compare the more sophisticated and empirically grounded versions that will follow. We now describe the specific sub-domain of our research, the basic ITS we have developed so far, and its first evaluation.

2 Linked Lists

A *linked list* is a data structure used to store information sequentially. It is composed of a set of *nodes*. Each node contains two pieces of information: a *value*, representing the data we are interested in storing, and a *link* to the following node of the list. Links between nodes are realized using *pointers*, that are explicit references to the memory locations where the nodes are stored. A graphical representation of a linked list can be seen in Figure 1.

Among numerous different data structures, linked lists play a very important role in the pedagogy of basic Computer Science, making them a particularly good topic for our research. Linked lists are usually presented early in Computer Science curricula; as such, more students see this topic. According to our experience on teaching data structures in classroom, students struggle with linked lists more than with other —sometimes more complex— data structures, such as stacks and binary search trees. The fundamental concepts of linked structures, pointer manipulations, object allocation, and traversals, which students learn in the context of linked lists, are all necessary for more complicated data structures, such as trees. Linked lists are important because students can learn these concepts in a relatively simple context, and they should not cause additional cognitive overhead when students are trying to understand more complicated structures. Part of what students learn while they struggle with linked lists is to think about an abstract visual model of their data, and to think of steps in a program/algorithm as making changes to that model. Mastering that way of thinking is a huge step for students, and one that they need to make to continue successfully in Computer Science.

In the linked list domain, there are several structural properties that a solution should have in order to be correct. For example, a list should contain the correct values, as specified in the description of each problem; lists should be free of cycles; lists should not terminate with undefined or incorrect pointers; no nodes should be made unreachable from any of the variables, i.e., lost in the heap space; nodes should be correctly deleted when necessary (this applies specifically to non-garbage collected languages, like C++). Having these properties in form of constraints allows our system to catch many common mistakes students make.

3 The iList System

The iList system works by providing a student with a simulated environment where linked lists can be seen and manipulated. Lists are represented graphically, and can be manipulated with programming language commands. Students are then asked by the system to solve problems in this environment, such as

insert new nodes in a given linked list, remove nodes, or perform other more complicated operations. As a student is working towards a solution, the system can provide feedback to help the student make progress.

A key difficulty about linked lists, as well as with other more sophisticated data structures, is that to really understand and use them effectively, students must think in pictures but act in code. The issue of multiple representations is subject of active research in science education [25,26,27]. In traditional data structures books, linked lists are illustrated with pictures, and it is sometimes difficult to connect that static representation with the dynamic procedures necessary to manipulate the structure itself. That is in fact what iList addresses. It makes the pictorial representation concrete. In a certain sense, the system reifies that conceptual image and makes it more accessible to the students. The central idea is that iList's interface is not just a box for entering input, but a dynamic visual environment that connects code actions to their effects on machine state.

Problem Types. The iList system supports two types of problems. The first kind of problems can be solved interactively, step-by-step. Students can enter a command into the system, and the system simulates the effect of that command, showing the effect of the action immediately on the simulated scenario. The second type of problems require writing a complete snippet of code, possibly involving structured conditional constructs like loops. Problems of this type usually introduce more than one initial scenario, and ask the student to write code that should work correctly in all the given scenarios. This setting forces the student to abstract away the specific details of a scenario, and think about more general algorithms for solving the problem on a wider range of situations.

The curriculum included in iList is currently composed of 7 problems, 5 of them of the first type, 2 of them of the second type. These problems have been carefully crafted based on some of the authors' experience as computer science educators, and on published CS curricula, such as ACM [7]. The goal is to challenge the students with the most common difficulties in manipulating linked lists. The problems are defined in the system using a human-readable XML format, making it easy to add new problems as needed.

Architecture. The architecture of iList is currently composed of four important modules: problem model, constraint evaluator, feedback manager, and graphical user interface. A student model and a pedagogical module, important components of a complete ITS [28], have not been implemented yet. Thus, the current version of iList is better defined as an *interactive learning environment*, rather than an ITS.

The *problem model* includes the representation of the problems presented to the student. A problem is given to the student in the form of a textual description and an initial scenario, which includes a configuration of variables and nodes (state space). The student is asked to progressively modify the state space by interactively providing a sequence of operations, until the desired configuration of the data structure has been reached.

When the student believes he/she is done with the current problem, the current state space is submitted to the *constraint evaluator*, that checks the given

solution. According to the constraint-based modeling paradigm, a solution is correct if it does not violate any constraint. Computationally, the evaluation of constraints is fairly simple. Each constraint is implemented as a computational unit with three fundamental functions: a boolean function checking the *relevance* of the constraint with respect to the solution, a boolean function checking the *satisfaction* of the constraint, and a *feedback* function responsible to return relevant information used to generate feedback for the student. A constraint is violated if the logic implication $isRelevant \Rightarrow isSatisfied$ is false for that particular state space. Constraints have access to two sources of information: the current student solution, and a correct solution provided with the problem definition. The specification of the correct solution needs only to include the minimum information necessary to evaluate a student solution, like the expected values of final lists. This is indeed one of the advantages of the constraint-based approach: the whole path towards a correct solution needs not to be specified in advance. This simplifies problem authoring, and most importantly, it allows alternative correct student solutions to be accepted by the system.

The *feedback manager* collects information from the individual constraints and builds a message directed to the student. Currently, this module simply relays messages provided by violated constraints, with minimum processing.

The *graphical user interface* is responsible for the main interaction with the student (Figure 1). The interface allows the student to interactively manipulate a data structure using C++ or Java commands. The command interpreter is quite flexible, allowing the student to focus more on the semantics of statements rather than language-dependent syntax details.

The system has been entirely implemented using the Java programming language. An early version of the system was interfaced to the WETAS system [29] for constraint evaluation. In subsequent versions, the constraint evaluator was re-implemented internally. To the user, the system appears as an applet integrated into a web page.

4 System Evaluation

A first version of the system has been deployed in a Computer Science class of a partner institution. 33 students took a pre-test before using the system, and a post-test immediately afterwards. After the post-test, the students also filled in a questionnaire about their subjective impressions on the system. The interaction of the students with the system was logged.

T-test on test scores revealed that *students did learn* during the interaction with iList (Table 1). We compared students' learning gain, defined as the difference between post-test score and pre-test score, with that of two other comparable groups of students. A group of 54 students interacted with a human tutor between the pre and post tests. The other group (control group, 53 students) attended a lecture about a totally unrelated topic between the two tests. The tutored group achieved statistically significant learning, whereas the control group did not (Table 1).

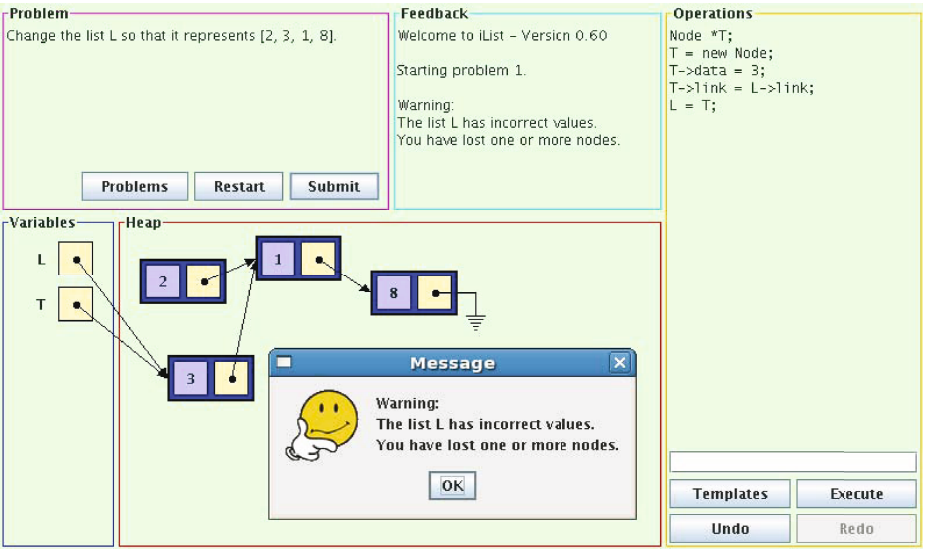


Fig. 1. A screenshot of iList

Table 1. Test scores. Range: 0 to 1.

Tutor	Pre-test score		Post-test score		Gain		T-test		
	Mean	Std dev	Mean	St dev	Mean	St dev	t	df	P
None	.34	.22	.35	.23	.01	.15	-0.56	52	ns
iList	.39	.23	.48	.27	.09	.17	-3.04	32	$< .01$
Human	.40	.26	.54	.26	.14	.25	-4.24	53	$< .01$

The learning gain of the iList group is somewhere in between the one observed in the control condition and the one of the tutored condition. ANOVA revealed overall differences between the three groups ($F(2, 137) = 5.96$, $P < 0.05$). Post hoc Tukey test indicated no significant difference between the control group and the iList group, nor between the iList group and the tutored group, whereas the difference between control and tutored groups is significant ($P < 0.01$).

The percentage of students who successfully solved each problem decreases with the problem number, as can be seen in Table 2. Problems were of increasing difficulty. Linear regression of individual problem success on learning gain showed a positive correlation between the number of problems successfully solved and learning. Also, we found significant positive correlation between solving the most difficult problems (number 5, 6, and 7) and learning (Table 3).

The first part of the questionnaire (Table 4) revealed that students felt that iList helped them learn linked lists to a moderate degree, and working with iList was interesting to them. The students found the feedback provided by the system somewhat repetitive, which is not surprising given the simple template-based

Table 2. Attempt and success rates on individual problems

Problem	1	2	3	4	5	6	7
Attempt rate	100%	100%	94%	91%	77%	74%	80%
Success rate	91%	80%	74%	66%	57%	46%	31%

Table 3. Linear regression. Each line represents an independent model.

Predictor	Dependent variable	R^2	β	df	F	t	P
Number of problems solved	Learning gain	.17	.41	1, 32	6.31	2.51	$< .05$
Problem 5 solved (yes/no)	Learning gain	.12	.35	1, 32	4.25	2.06	$< .05$
Problem 6 solved (yes/no)	Learning gain	.16	.40	1, 32	5.80	2.41	$< .05$
Problem 7 solved (yes/no)	Learning gain	.13	.36	1, 32	4.63	2.15	$< .05$
Questionnaire question 1	Learning gain	.22	.47	1, 31	8.33	2.89	$< .01$
Questionnaire question 4	Learning gain	.16	-.40	1, 31	5.83	-2.42	$< .05$
Questionnaire question 5	Learning gain	.37	.61	1, 31	17.72	4.21	$< .01$
Questionnaire question 6	Learning gain	.12	.36	1, 31	4.32	2.08	$< .05$
Questionnaire question 7	Learning gain	.35	-.59	1, 31	16.09	-4.01	$< .01$
Learning gain	Final class grade	.12	.36	1, 31	4.35	2.09	$< .05$

Table 4. Questionnaire: scaled questions

Question (Scaled response: 1=No to 5=Yes)	Mean	Std dev
1. Do you feel that iList helped you learn about linked lists?	2.9	1.2
2. Do you feel that working with iList was interesting?	4.0	1.3
3. Did you read the verbal feedback the system provided?	4.3	1.0
4. Did you have any difficulty understanding the feedback?	3.0	1.5
5. Did you find the feedback useful?	2.3	1.2
6. Did you ever find the feedback misleading?	2.2	1.2
7. Did you find the feedback repetitive?	3.9	1.2

generation mechanism. Also, the feedback was considered not very useful, but at least not too misleading.

Linear regression of questionnaire answers on learning gain revealed some significant correlations between students' feelings about the system and their learning (Table 3). The students who felt that iList helped them the most or found the feedback useful did indeed learn the most (questions 1 and 5). Those who had trouble understanding the feedback or found the feedback repetitive learned less (questions 4 and 7). Strangely, the students who found the feedback misleading learned more (question 6). A possible explanation may be that those students were more careful and exercised more critical thinking, thus getting more out of their interaction with the system.

Interestingly, students declared that they read the feedback provided by the system, but our evidence points to the opposite conclusion. From the log of the system, we estimated that students read feedback messages for 3.56 seconds on average (stdev = 2.66 seconds), resulting in a reading rate of 532 words/minute

(stdev = 224 words/minute). According to Carver's taxonomy [30], such speed corresponds to the process of quickly skimming a text. According to the same taxonomy, the activity of reading to learn would require a much lower rate, in the order of 200 words/minute. Possibly, the repetitiveness of feedback messages could have made the students ignore them [31].

The last item in the questionnaire was an open response question, asking the students for any comments on the program. The detailed comments provided by the students and the instructor of the class will be helpful in guiding further improvements of the system.

Finally, linear regression revealed a positive correlation of the learning gain obtained with iList with the students' final grade in the data structure class in which they used iList (Table 3). There is then a chance that the little bit of knowledge that students acquired interacting with iList carried over to their final exam, and hopefully will help them in their future career in Computer Science.

5 Future Work

We plan to significantly extend the functionalities of iList. We will design and implement a student model, to keep track of students' history and estimate their state of knowledge exploiting the modeling power of the constraint-based knowledge representation. Pedagogical strategies will be implemented, following the results of our data analysis and those already published in the literature.

One of the research issues we are mostly interested in is the delivery of effective feedback to students. We plan to build a more sophisticated feedback module, grounding its behavior in the outcome of the analysis of our tutorial data, as well as in our past experience with the development of natural language interfaces for ITSs [21,22]. A preliminary analysis of our human tutorial dataset suggested that *positive feedback*, i.e., reaction to correct student actions, may play an important role in tutoring [23]. We are planning on investigating the conditions and the modalities in which positive feedback is delivered by human tutors, and build a computational model of positive feedback that will be implemented and evaluated in iList. Providing meaningful positive feedback in ITSs, in particular in constraint-based ITSs, is still an open problem, and a system like iList will be a useful testbed for researching that problem.

Acknowledgments. This work is supported by award N00014-07-1-0040 from the Office of Naval Research, and additionally by awards ALT-0536968 and IIS-0133123 from the National Science Foundation.

References

1. Evens, M., Michael, J.: One-on-one Tutoring by Humans and Machines. Lawrence Erlbaum Associates, Mahwah (2006)
2. Graesser, A.C., Person, N.K., Magliano, J.P.: Collaborative dialogue patterns in naturalistic one-to-one tutoring. Applied Cognitive Psychology 9, 1–28 (1995)

3. Litman, D.J., Rosé, C.P., Forbes-Riley, K., Van Lehn, K., Bhembe, D., Silliman, S.: Spoken versus typed human and computer dialogue tutoring. International Journal of Artificial Intelligence in Education 16, 145–170 (2006)
4. VanLehn, K., Siler, S., Murray, C.: Why do only some events cause learning during human tutoring? Cognition and Instruction 21(3), 209–249 (2003)
5. AA.VV.: US bureau of labor statistics, http://www.bls.gov/oco/oco20016.htm
6. Katz, S., Allbritton, D., Aronis, J.M., Wilson, C., Soffa, M.L.: Gender and race in predicting achievement in computer science. IEEE Technology and Society, Special Issue on Women and Minorities in Information Technology 22(3), 20–27 (2003)
7. AA.VV.: Computing Curricula 2001 – Computer Science (2001) Report of the Joint Task Force (IEEE Computer Society, Association for Computing Machinery), http://www.sigcse.org/cc2001/
8. Warendorf, K., Tan, C.: ADIS - an animated data structure intelligent tutoring system or putting an interactive tutor on the WWW. In: Intelligent Educational Systems on the World Wide Web (Workshop Proceedings). Eighth World Conference of the AIED Society (1997)
9. Lane, H.C., VanLehn, K.: Coached program planning: Dialogue-based support for novice program design. In: Proceedings of the Thirty-Fourth Technical Symposium on Computer Science Education (SIGCSE 2003), pp. 148–152. ACM Press (2003)
10. Graesser, A.C., Person, N., Lu, Z., Jeon, M.G., McDaniel, B.: Learning while holding a conversation with a computer. In: PytlikZillig, L., Bodvarsson, M., Brunin, R. (eds.) Technology-based education: Bringing researchers and practitioners together, Information Age Publishing (2005)
11. Corbett, A.T., Anderson, J.R.: The effect of feedback control on learning to program with the Lisp tutor. In: Proceedings of the Twelfth Annual Conference of the Cognitive Science Society, Cambridge, pp. 796–803 (1990)
12. Kumar, A.N.: Model-based reasoning for domain modeling, explanation generation and animation in an ITS to help students learn C++. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363. Springer, Heidelberg (2002)
13. Sykes, E., Franek, F.: An intelligent tutoring system for learning to program in Java. In: IEEE International Conference on Advanced Learning Technologies (2003)
14. Kalayar, M., Imekatsu, H., Hirashima, T., Takeuchi, A.: An intelligent tutoring system for search algorithms. In: Proceedings of ICCE 2001, pp. 1369–1376 (2001)
15. Mitrović, A., Suraweera, P., Martin, B., Weerasinghe, A.: DB-suite: Experiences with three intelligent, web-based database tutors. Journal of Interactive Learning Research 15(4), 409–432 (2004)
16. Ohlsson, S.: Constraint-based student modelling. Journal of Artificial Intelligence in Education 3(4), 429–447 (1992)
17. Ohlsson, S.: Learning from performance errors. Psychological Review 103, 241–262 (1996)
18. Kodaganallur, V., Weitz, R.R., Rosenthal, D.: A comparison of model-tracing and constraint-based intelligent tutoring paradigms. International Journal of Artificial Intelligence in Education 15, 117–144 (2005)
19. Mitrović, A., Ohlsson, S.: A critique of Kodaganallur, Weitz and Rosenthal, A comparison of model-tracing and constraint-based intelligent tutoring paradigms. International Journal of Artificial Intelligence in Education 16, 277–289 (2006)
20. Kodaganallur, V., Weitz, R.R., Rosenthal, D.: An assessment of constraint-based tutors: A response to Mitrovic and Ohlsson's critique of A comparison of model-tracing and constraint-based intelligent tutoring paradigms. International Journal of Artificial Intelligence in Education 16, 291–321 (2006)

21. Di Eugenio, B., Fossati, D., Yu, D., Haller, S., Glass, M.: Aggregation improves learning: Experiments in natural language generation for intelligent tutoring systems. In: ACL 2005, Proceedings of the 42nd Meeting of the Association for Computational Linguistics, Ann Arbor (2005)
22. Di Eugenio, B., Fossati, D., Yu, D., Haller, S., Glass, M.: Natural language generation for intelligent tutoring systems: A case study. In: AIED 2005, 12th International Conference on Artificial Intelligence in Education, Amsterdam, The Netherlands (2005)
23. Fossati, D.: The role of positive feedback in intelligent tutoring systems. In: ACL 2008, The 46th Annual Meeting of the Association for Computational Linguistics, Student Research Workshop, Columbus (2008)
24. Ohlsson, S., Di Eugenio, B., Chow, B., Fossati, D., Lu, X., Kershaw, T.C.: Beyond the code-and-count analysis of tutoring dialogues. In: AIED 2007, 13th International Conference on Artificial Intelligence in Education (2007)
25. Goldman, S.R.: Learning in complex domains: When and why do multiple representations help (commentary). Learning and Instruction 13, 239–244 (2003)
26. Meltzer, D.E.: Relation between students' problem-solving performance and representational format. American Journal of Physics 73(5), 463–478 (2005)
27. Hundhausen, C.D., Douglas, S.A., Starko, J.T.: A meta-study of algorithm visualization effectiveness. Journal of Visual Languages and Computing 13(3), 259–290 (2002)
28. Beck, J., Stern, M., Haugsjaa, E.: Applications of AI in education. ACM crossroads (1996), http://www.acm.org/crossroads/xrds3-1/aied.html
29. Martin, B., Mitrović, A.: WETAS: A web-based authoring system for constraint-based ITS. In: Bra, P.D., Brusilovsky, P.L., Conejo, R. (eds.) Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Malaga, pp. 543–546. Springer (2002)
30. Carver, R.P.: Reading Rate: A Review of Research and Theory. Academic Press, San Diego (1990)
31. Heift, T.: Error-specific and individualized feedback in a web-based language tutoring system: Do they read it? ReCALL Journal 13(2), 129–142 (2001)

Re-evaluating LARGO in the Classroom: Are Diagrams Better Than Text for Teaching Argumentation Skills?

Niels Pinkwart[1], Collin Lynch[2], Kevin Ashley[3], and Vincent Aleven[4]

[1]Clausthal University of Technology, Department of Informatics, Germany
[2]University of Pittsburgh, Intelligent Systems Program, Pittsburgh, PA, USA
[3]University of Pittsburgh, Learning Research and Development Center, Pittsburgh, PA, USA
[4]Carnegie Mellon University, HCI Institute, Pittsburgh, PA, USA

Abstract. Diagrams appear to be a convenient vehicle for teaching argumentation skills in ill-defined domains, but can an ITS provide useful feedback on students' argument diagrams without assuming a well-defined procedure for objectively evaluating argument? LARGO is an ITS for legal argumentation that supports students as they diagram transcripts of US Supreme Court oral argument. It provides on-demand advice by identifying small, interesting or incomplete patterns within students' graphs. We conducted a study in which LARGO was used as mandatory part of a first-year law school class. In contrast to prior findings in lab studies with voluntary participants, the use of LARGO did not lead to superior learning as compared to a text-based note-taking tool. These results can be partially attributed to low use of the graphical tools and advice by the students as well as (and possibly due to) a different motivational focus. Some evidence was found that higher engagement with the system led to better learning, leaving open the tantalizing possibility of helping especially lower-aptitude students through use of LARGO.

Keywords: Ill-defined Domains, Legal Argumentation, Diagram Representations, ITS Evaluation.

1 Introduction

In a variety of domains, a central goal of education is training students to produce *robust* arguments that not only address the current problem but survive the test of other examples and cases that have been encountered in the past or that may arise in the future. When a student proposes a rule for defining a class of mathematical objects, a theory for explaining scientific data, or a rule justifying a legal decision, one expects other students or the teacher to respond, "But what if…." That is, they test the proposal by posing hypothetical examples or cases that may occur and that highlight potential problems with the proposed rule or theory.

Law students are taught to make arguments through Socratic classroom dialogue, participation in moot court sessions and the analysis of examples, notably important precedents. These activities imitate court room arguments. Advocates before the court make their arguments by proposing *tests* or legal rules which, if adopted and used to

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 90–100, 2008.

decide the case at hand, would achieve their goals. To challenge these proposed decision rules, an opponent or judge may pose *hypothetical cases* that may occur, are relevant to the issues of the argument, and illustrate situations that the rule should cover but does not or decides wrongly given the underlying principles and policies of the law. The advocates can then respond by modifying their tests as needed to cover or avoid the hypothetical case, or by distinguishing the hypothetical situation from the facts of the case [12].

Interestingly, in legal education, teachers instruct students by engaging them in practice making and responding to such arguments, but seldom make explicit the process itself. If a student's argument has a flaw, the teacher does not explain the flaw; instead, the teacher typically will respond to the argument with a counterargument that exploits the flaw, thus leaving to the student the responsibility of later reflecting on why his argument was weak. It is not always clear why this approach is taken. It raises the possibility that students might learn better if their self-reflections about the process were explicitly guided. ITS systems such as CATO [1] and ArguMed [16] could help as tools both to give students practice in making arguments and to make explicit the process of argumentation.

Graphical representations of argument and argument diagramming have gained currency in recent years [4,13]. Proponents of argument diagrams argue that they can make the essential logical relations explicit while retaining formal validity. Work by Carr [5] in the legal domain indicated that the production of argument diagrams can improve students' ability to produce high-quality arguments, and Schank [14] showed that the production of diagrams can improve students' argument coherence. Recent work by Harrell [7] and Easterday et al. [6] has substantiated that argument diagrams can be useful learning tools. In summary, the current state of research suggests that diagrams are a useful educational tool, but controlled empirical studies are still rare.

The LARGO Intelligent Tutoring System [3,10,11] for legal argumentation supports students in the process of analyzing oral argument transcripts (taken from the U.S. Supreme Court). These are complex, real-world examples of the kind of Socratic arguing with tests and hypotheticals in which professors seek to engage students in class. However, they are written rather than purely oral as in the classroom, and thus may be good examples to use in reflecting upon the process of argument. Since these transcripts tend to be more complicated than classroom arguments, students probably need support in order to understand and reflect on them. LARGO provides that support by capitalizing on the pedagogical value of argument diagrams. While using the system, students read through the transcript and produce a graphical markup of it, identifying the key tests, hypotheticals, responses, and facts as well as the relationships between them. LARGO helps students by giving feedback in the form of self-explanation prompts.

In the fall of 2006, we conducted a study of LARGO with paid volunteers from the first year Legal Process course at the University of Pittsburgh's School of Law. The subjects analyzed a pair of cases using either LARGO or a text-based note-taking tool. We found no overriding differences between the two conditions. However, lower aptitude students, as measured by their Law School Admission Test (LSAT) score (a frequently-used predictor for success at law schools), showed higher learning gains using LARGO than using the note-taking tool. Also, the use of LARGO's on-demand

help features was strongly correlated with learning [11]. Further analysis indicated that familiarity with the system led students to engage in better note taking [9].

Since participation in the study was voluntary, the students were self-selected (from among those enrolled in the course) for their interest in the curriculum, the ITS, and the pay. Many expressed an interest in the system, making it apparent that they were among the more inquisitive members of their class. We therefore concluded that a second study was necessary to further examine and substantiate the findings with non-voluntary participants. We sought out an opportunity where LARGO would be required in a course setting, so that we would have a sample of students that is more directly representative of the LARGO target population, and that (compared to our earlier study) may include a larger proportion of lower-LSAT students, for whom LARGO was most effective in that earlier study.

Based on our prior results, the two hypotheses for the new study are: a) Lower-aptitude students will derive more benefits from LARGO than their higher-aptitude peers, and b) additional experience with the system will improve students' use and benefit of it (i.e., we hypothesized stronger effects than in our previous study, if we include more study sessions). The following sections of this paper describe the type of argumentation LARGO teaches, and the design and results of the study.

2 Arguing with Tests and Hypotheticals

An example taken from the case *Asahi Metal Industry Co. v. Superior Court,* (480 U.S. 102 (1987)), illustrates both the process taught, legal reasoning with test and hypotheticals, and the way in which LARGO's argument diagrams support learning. Law students encounter the *Asahi* case in their first semester "Legal Process" course. It deals with an important legal concept: *personal jurisdiction*, a court's power to require that a person appear in court and defend against a lawsuit.

Cases like *Asahi* involve a court in one state attempting to assert power over a non-resident of that state. In such cases, the principle that a state's courts may redress in-state harm conflicts with the U.S. Constitutional guarantee of "Due Process" requiring safeguards against the arbitrary exercise of government power. In *Asahi*, a motorcycle accident injured the driver and killed his wife. The driver filed a product liability claim against Cheng Shin, the Taiwanese maker of the tire in a California state court, alleging that a defect caused the accident. Shin in turn filed a claim against Asahi, the Japanese manufacturer of the tire's valve assembly, alleging that a defective valve caused the accident. Asahi moved to dismiss for lack of personal jurisdiction. The case made its way to the U.S. Supreme Court.

A typical Legal Process course book would include the Supreme Court's opinion in *Asahi* along with the facts of the case and its reasons. A law professor likely would engage the class in a Socratic discussion of the meaning and limitations of the Court's rule and alternate rules it might have adopted. If a student argued that Asahi should be subject to jurisdiction in California as its valves ended up there (i.e., proposed a test), the professor might ask: "How far up the stream of supply does it go? Does California have jurisdiction over the steel maker whose steel is in the valve?" (i.e., poses a hypothetical). Students learn to respond to such questions by analogizing the hypothetical to or distinguishing it from the case facts and defending the proposed rule, modifying

the rule to accommodate it, or abandoning the rule in favor of another. In this way, the professor introduces students to the legal rules of personal jurisdiction, and to the *nature* of legal rules, the fact that they are defeasible, have an open texture, and may be applied differently in different circumstances.

At the U.S. Supreme Court, advocates often propose tests that decide the case at hand in their favor. The Justices often evaluate the tests by posing hypothetical cases like the one above to probe the test's meaning, its limits and consistency with precedents, principles, and policies. Thus, oral arguments at the U.S. Supreme Court provide complex examples of the kind of reasoning employed in the classroom, and therefore have a pedagogical value. Traditionally, however, oral arguments have not been employed in law school classes due to their complexity and lack of availability.

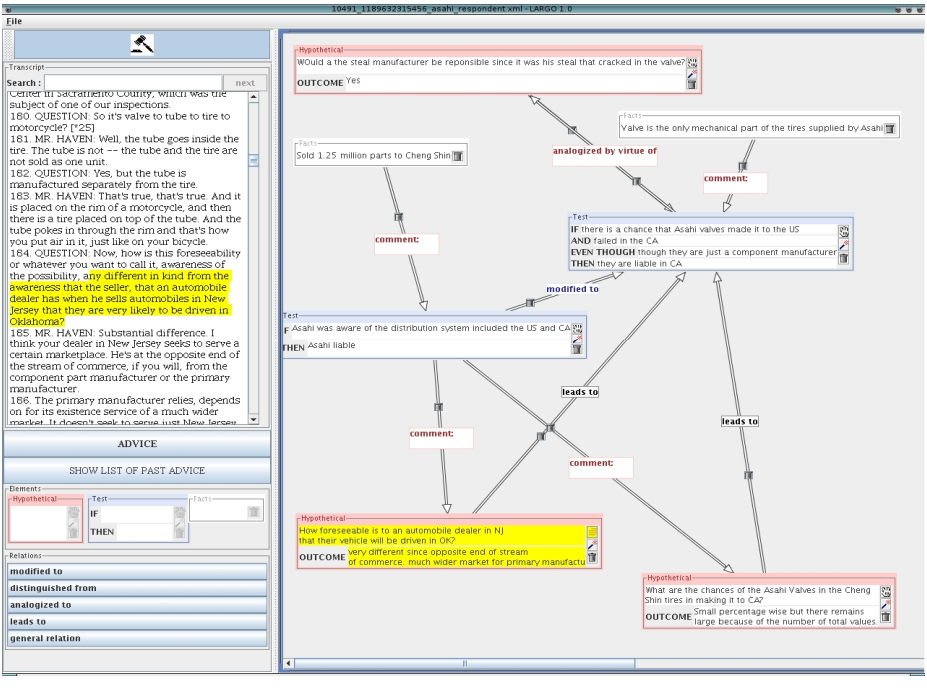


Fig. 1. A student diagram for the Asahi argument

Figure 1 shows a student's actual LARGO diagram for a portion of the *Asahi* argument. The argument transcript is shown on the left, together with two buttons for advice and a palette from which the student can select the main graph elements. The student has identified two tests in the argument transcript, one a modification of the other, three hypotheticals posed, and a number of relations among them that (in this student's diagram somewhat imperfectly) reflect the role the hypotheticals play in evaluating the tests. LARGO helps students to find, diagram and relate the important elements of the text by providing hints based on small specific argument patterns [10].

3 Study Description

We carried out a study to evaluate LARGO within one section of the 2007 first-year Legal Process course at the University of Pittsburgh School of Law. All 85 subjects in the section were required to complete the activities in the study. The students were not paid but were given coffee gift cards as a token of appreciation. Since students are assigned randomly to one of three course sections, we have every reason to believe that the section that participated in the experiment is representative of their peers. The LARGO curriculum (which consisted of three personal jurisdiction cases) was integrated into the class as preparation for a graded writing assignment on personal jurisdiction, counting for 10% of their grade.

The students were assigned to two study conditions, balanced by LSAT scores, but otherwise assignment was random. The experimental group used a graphical version of LARGO that supported diagram creation and gave advice [11], as described above. The control group made use of a text version that offered no feedback. The curriculum consisted of six weekly two-hour sessions. In the first week, the students took a multiple-choice pre-test. During the second week they read background material on *Asahi* and annotated the transcript in LARGO or the text tool. They then answered two written questions about it without their diagrams or notes. Over the next two weeks they completed two more cases in the same way. During week five they took a post-test consisting of multiple-choice and free answer questions. Finally, we offered a debriefing session to show students in each condition the version that had been used by the other condition, in order to compensate for any differences in learning between conditions prior to the course exam.

We classified the test items by type. Both the pre-test and post-test contained multiple-choice questions about: a) everyday reasoning with hypotheticals; b) generic aspects of tests and hypotheticals in legal argument; c) the domain of personal jurisdiction; and d) generic argument questions drawn from the LSAT. The post-test also contained: e) factual recall questions related to the specific transcripts studied during training; f) interpretation questions regarding these transcripts; and g) analysis and free-text questions regarding a novel case. We also grouped the items with respect to the aspect of the argument model to which they were most related (hypothetical, test, legal issues, legal policies, relation between test and hypothetical, response to hypothetical). The design of the study and the materials used were the same as in the 2006 study [11], except that one extra training session was added.

4 Results

All 85 students completed the study. While they had a maximum of two hours time to work on each of the training cases, their average time per case was 55.8 minutes (sd=13.3). There was no significant training time difference between the conditions.

We excluded a total of 15 students from the analysis. Four candidly told us that they were not working and deliberately entered off-topic responses in the post-test. Two others completed the post-test in less than 30 minutes, less time than is needed merely to read the materials (approx. 50 minutes). The remaining 9 spent less than 30 minutes on one or more of the training cases, less time than it takes an expert to work

through the material (approx 45 minutes). It is therefore highly unlikely that they put considerable effort into their task. The analyses below are based upon the remaining 70 students (36 Control, 34 LARGO).

Table 1 contains the mean scores and standard deviations of the case-specific post-test questions (i.e., the post-test only items). Table 2 shows the pre-post gains for counterbalanced items shared between the tests. All scores are given on a [0,1] scale. Both tables show the results for all 70 students as well as the sub-results for the 27 low-LSAT students whose LSAT scores are below the median of 159. For this group, our previous study showed a positive effect of LARGO as compared to the text tool.

Table 1. Study results for post-test only items

mean (sd) of post-test score	All students (N=70)		Low-LSAT students (N=27)	
	Control	LARGO	Control	LARGO
All items	.63 (.09)	.64 (.09)	.64 (.08)	.61 (.11)
Case Interpretation	.46 (.11)	.48 (.10)	.45 (.10)	.49 (.11)
Case Recall	.71 (.10)	.73 (.12)	.73 (.09)	.67 (.14)
Hypotheticals	.71 (.12)	.71 (.14)	.72 (.13)	.64 (.14)
Legal issues	.39 (.49)	.35 (.48)	.50 (.52)	.38 (.51)
Legal policies	.36 (.49)	.29 (.46)	.50 (.52)	.23 (.44)
Relations tests/hypotheticals	.48 (.11)	.50 (.11)	.49 (.11)	.48 (.16)
Responses to hypotheticals	.44 (.23)	.45 (.24)	.40 (.32)	.49 (.28)
Tests	.75 (.18)	.79 (.15)	.75 (.17)	.76 (.21)

Table 2. Study results for counterbalanced between tests

mean (sd) of gain score	All students (N=70)		Low-LSAT students (N=27)	
	Control	LARGO	Control	LARGO
All items	-0.01 (.16)	-0.04 (.18)	-0.01 (.13)	-0.08 (.19)
Everyday argumentation *	0.01 (.34)	-0.05 (.36)	0.09 (32)	-0.19 (.38)
Generic items	-0.01 (.31)	-0.01 (.27)	-0.02 (.28)	-0.03 (.25)
LSAT questions	-0.03 (.23)	-0.02 (.24)	-0.02 (.20)	-0.06 (.25)
Personal jurisdiction *	0.07 (.40)	-0.13 (.42)	0.00 (.35)	-0.21 (.32)
Hypotheticals	0.08 (.53)	0.00 (.49)	0.21 (.42)	0.00 (.41)
Relations tests/hypotheticals	-0.01 (.22)	0.01 (.30)	-0.03 (.24)	-0.07 (.40)
Responses to hypotheticals	0.06 (.39)	0.01 (.34)	0.14 (.36)	-0.12 (.36)
Tests	-0.17 (.65)	-0.18 (.52)	-0.36 (.63)	-0.15 (.55)

There were no significant differences between the two conditions with respect to post-test only test items – neither overall nor for the lower LSAT subjects.

For the question types that were shared between pre-test and post-test (in a counterbalanced manner), the Control group gained significantly more than the LARGO group on the personal jurisdiction items ($F(1,68) = 4.250$; $p<.05$). For the low LSAT students, the Control group gained significantly more than the LARGO group on the "everyday hypothetical argumentation with hypotheticals" questions ($F(1,25) = 4.313$; $p<.05$). No other significant differences were found. A repeated measures analysis reveals that the only significant difference between pre-test and post-test scores is a drop for the low-LSAT LARGO students ($F(1,10)= 5.333$; $p<.05$) on the "personal jurisdiction" domain questions.

These results seemingly contradict our 2006 results where the low-LSAT LARGO students outperformed their Control peers on several important question types. When we analyzed the log files from the study sessions and the LARGO diagrams, we found that the 2006 students made far greater use of LARGO's advice functions than the students in the current study (see Table 3). Moreover, in the current study, the advice usage dropped over time unlike in 2006: during the last session, on average only 0.6 advice requests were made per case (1.6 during the first case). The diagrams created in the current study contained fewer elements and relations than those from the 2006 study, and students in the current study did not link their diagram elements to the transcript as often (31% vs. 87%).

Table 3. Advice usage and diagram complexity

mean (sd)	2006 study (N=15)	2007 study (N=34)
Clicks on Advice button (shows 3 hints) per case	10.1 (10.8)	1.8 (3.9)
Selection of one of the 3 shown hints per case	7.6 (8.2)	1.2 (2.2)
Advice usage by case over time	increasing from 7.1 to 8.1	decreasing: 1.6, then 1.3, then 0.6
Number of elements in student graphs	9.6 (2.7)	7.5 (2.3)
Number of relations in student graphs	7.9 (2.3)	5.2 (2.9)
Rate of elements that are linked to the transcript	.87 (.23)	.31 (.31)

Table 4. Correlations between advice requests in LARGO and test scores

Pearson correlations	All students (N=34)			Low-LSAT students (N=13)		
	Pre-test	Post-test	Gain	Pre-test	Post-test	Gain
Case Interpretation	-	.03	-	-	.15	-
Case Recall	-	-.05	-	-	.02	-
Everyday argumentation	-.06	.34 *	.33	.07	.46	.29
Generic items	-.06	-.19	-.18	.06	-.14	-.21
LSAT questions	-.07	.02	.06	-.11	.30	.24
Personal jurisdiction	-.09	.21	.16	.04	.46	.30
Hypotheticals	-.02	-.19	-.04	.24	-.18	-.28
Relations tests / hypotheticals	-.09	-.20	-.17	.28	-.29	-.37
Responses to hypotheticals	-.15	.29	.33	-.16	.54	.61 *
Tests	.05	.06	-.03	-.07	.11	.16

*: significant correlations ($p<.05$).

Together, these results seem to indicate that LARGO's advice was a key factor in the positive effects that we observed in 2006, and that the graphical representation alone is not sufficient. We therefore analyzed if, within the current study, a higher number of advice requests correlates with higher post-test or gain scores.

The pre-test scores are not correlated with advice usage: students with higher pre-test scores did not use help more or less often than students with lower pre-test scores. However, advice usage is positively correlated with the post-test score for everyday argumentation items for all subjects. For low-LSAT students, the advice usage is also highly positively correlated to pre/post gains on items about responses to hypotheticals. The advice given by the system, apparently, helped these students to better understand how one can respond to a hypothetical during (legal or everyday) argument.

These strategies are mentioned in the feedback messages LARGO provides, which supports this hypothesis. Due to the relatively small number of low-LSAT LARGO students (N=13), additional correlations (e.g., for everyday argumentation or for personal jurisdiction) did not reach the level of statistical significance at the .05 level. However, the general trend is that advice seems to have a positive effect on the performance of the lower LSAT students.

5 Discussion

The study results did not confirm our initial hypotheses: the use of LARGO as a mandatory (though non-graded) part of a legal process course did not lead to learning gains when compared to a simple note-taking tool. Further, students in both conditions did not improve from pre-test to post-test in any of the tested categories even though they studied the materials for approximately 6 hours. These results are not in line with our 2006 findings with paid volunteers, even though the experiment was similar in all respects. We see three possible ways of accounting for these differences: student motivation, engagement with the system, and post-test design.

5.1 Motivational Issues

The extent to which users engage with a system depends on their specific goals. In 2006 the users were volunteers paid for their participation. As such they appear to have been more motivated to explore the system, to exercise key features such as graphical relations, links between diagram and transcript, and on-demand advice, and to take their time. Our present population comprised unpaid "conscripts" who had to use the system as a part of their course. They were inclined to use the system in the most convenient manner possible and tended to underutilize its key features. In many ways they used the system as a note-taking tool with movable text boxes.

Yet, the success or failure of an ITS, and particularly of one that offers its important features on demand as LARGO does, depends on the extent to which users actually use these features. In our prior study, the low-LSAT students chose to make use of LARGO's key features and showed performance gains. In the present study, neither the high- nor the low-LSAT students did so consistently. Thus, the LARGO group derived fewer benefits from the system and performed no better than the Control group. To get students to engage with the beneficial features outside of the lab it seems necessary to better integrate the tool into the classroom. In the current study, use of LARGO was aligned with the course goals but not a core part of the course. Students were required to participate in the LARGO sessions, but were not graded on these activities. The payoff for the students lay in the preparation that the activities gave them for their future work. If we want the students to use the on-demand system functions, future studies of LARGO (and probably this result is valid also for other ITSs) should pay more nuanced attention to the specific motivation of the students, especially in real classroom situations. This can probably be done by assigning grades to the graphs that students create with LARGO and through in-class support (e.g., discussion of the benefits of LARGO for the learning goals).

5.2 Engagement with the System

Our analysis of the study data suggests that low use of the LARGO advice functions at least partially accounts for the lack of difference between the study conditions: the LARGO students who used the advice more frequently did better at some centrally important post-test questions. The low usage of important system features may be connected to motivational issues (cf. 5.1). Consequently, we may need to modify LARGO in order to increase the student's engagement with the system even if their motivation to do so may be low. The current version of LARGO leaves many things to the users –the way they create the diagrams, how and if they link elements in the diagram to specific passages in the transcript being studied, and how often (if at all) they receive comments and feedback on their work. As previous research shows, this strategy may be problematic not only due to motivational aspects, but also because students often do not ask for help even though they could benefit from it [2]. The diagrams the students created in the current study support this position. A large number of students' graphs had errors of a type that would be noted and commented on by LARGO if the student requested its advice. But since students did not do so very frequently, they were often not informed of their misconceptions.

How could LARGO be redesigned to avoid this problem? Presenting corrective feedback immediately after they make a mistake (as done by many successful ITS systems) would be problematic in the ill-defined domain of legal argumentation. As described in [10], LARGO's on-demand feedback avoids false error messages that are likely to occur in this domain, where it is often not clear whether a diagram correctly reflects an argument or not. False or inappropriate feedback would be very problematic also because the feedback LARGO gives is cognitively demanding (self-explanation prompts).

A reasonable alternative and a compromise between the two extremes, to be tested in further studies, could be to highlight diagram regions on which LARGO could give feedback (similar to the feedback in Andes [15]). Thus, students would be aware that feedback is available, but would not be forced to attend to it immediately (or at all). Another design option would be to structure the interaction with LARGO so that the students have "diagram creation" phases and also phases where they are explicitly asked to reflect on their diagrams, assisted by advice from LARGO.

Perhaps students could also be made to engage more with LARGO by requiring a clear and tangible "result" of their analysis (e.g., a "final test") that could be checked against what actually happened in court. Also, it may be interesting to give feedback to students indicating whether they did better or worse than the attorney in the actual case, or than peer students, and how their result relates to the final opinion of the court. Also, a future version of LARGO could present additional material not contained in the transcript, and engage students more in actually *making* arguments in addition to *analyzing* them.

5.3 Post-Test Design

Many researchers argue that even without feedback, diagrams are better than texts for learning argument skills. In that light, one would have expected a benefit of LARGO in this study even though the advice usage was low. However, this was not the case.

Could it be that the post-test somehow did not fully measure what was taught? At the content level, that notion can be rejected. The post-test items were well aligned with the tasks students had to solve in the training session. Yet, there was a subtle (and necessary) difference between what we tested and what was taught with LARGO. During training, the LARGO students created graphs, whereas the post-test employed a textual notation only, since this is the standard format in which legal argument and legal reasoning tasks are presented to students. However, the effectiveness of graphical tools generally strongly depends on the amount and type of usage of these tools [8], and our chosen format may have favored the students in the text condition The graphs created by the students could have been used for some of the questions on the posttests (and a few students asked for them for exactly that purpose), and would surely have helped, but we did not provide them. Thus, we tested whether training with graphs *transfers* to textual questions better than training with texts, not whether students were able to use the representations they created effectively in a post-test. As mentioned, we deemed a textual post-test to have higher ecological validity.

6 Summary and Conclusion

In this study we tested the LARGO ITS as a mandatory part of a first-semester law school course. Prior research on graphical argument representations has suggested that the graphical format of LARGO and the on-demand help it provides would be beneficial. However, our results showed no evidence that the LARGO condition was better than the Control condition. The post-test was well-aligned with the instruction and we had sufficient statistical power. Our hypothesis that graphs are better then text for learning complex argumentation skills was not confirmed. The students who used graphs were also no worse than the text users - since many ITSs for argumentation rely on the graph structure as a central component to enable the system feedback, this is still an important result for ITS designers. Yet, it contradicts our prior positive results with LARGO in lab studies [10].

Although we did not find a difference between the two conditions, the study provides some evidence that those students who engaged more with the graphs as evidenced by more frequent use of LARGO's advice function, especially the low-LSAT students, did better than the text condition. This finding is consistent with our 2006 study [10] in which the paid volunteers used more of the LARGO features and benefited from them.

One tentative conclusion to take away from this study is that graphs may still be better than text, but that engagement is essential. One way to support engagement could be to change the feedback mechanism. The current on-demand feedback is well suited for ill-defined domains since it avoids false error messages, but it remains to be explored whether prompting the student with messages (at the risk of giving inappropriate or suboptimal advice) or at least highlighting "weak regions" in diagrams will engage the students and not confuse them. Another take-home message of the study is that the subject's motivation is a decisive factor, especially when "leaving the lab" and entering the classroom with ITS technology. Apparently, and somewhat to our surprise, it can make a difference whether participation is voluntary or mandatory – and if it is mandatory, whether the students are motivated to participate in a manner

so that the key ITS features are used, especially if their usage is on-demand. Future studies with LARGO – on its way toward regular classroom usage – will have to take these factors into account.

Acknowledgment. NSF Grant IIS-0412830 supported this work.

References

1. Aleven, V.: An intelligent learning environment for case-based argumentation. Technology, Instruction, Cognition, and Learning 4(2), 191–241 (2006)
2. Aleven, V., Stahl, E., Schworm, S., Fischer, F., Wallace, R.M.: Help Seeking in Interactive Learning Environments. Review of Educational Research 73(2), 277–320 (2003)
3. Ashley, K., Pinkwart, N., Lynch, C., Aleven, V.: Learning by Diagramming Supreme Court Oral Arguments. In: Proceedings of the 11th International Conference on Artificial Intelligence and Law, pp. 271–275. ACM Press, New York (2007)
4. van den Braak, S.W., van Oostendorp, H., Prakken, H., Vreeswijk, G.: A Critical Review of Argument Visualization Tools: do users become better reasoners? In: Workshop notes of the ECAI 2006; Workshop on Computational Models of Natural Argument, Italy (2006)
5. Carr, C.S.: Using Computer Supported Argument Visualization to Teach Legal Argumentation. In: Visualizing Argumentation, pp. 75–96. Springer, London (2003)
6. Easterday, M., Aleven, V., Scheines, R.: Tis better to construct than to receive? The effects of diagramming tools on causal reasoning. In: Proceedings of AIED, pp. 93–100. IOS Press, Amsterdam (2007)
7. Harrell, M.: The improvement of critical thinking skills in What Philosophy Is (Tech. Rep. CMU-PHIL-158). Carnegie Mellon University, Department of Philosophy (2004)
8. Hundhausen, C., Douglas, S., Stasko, J.: A Meta-Study of Algorithm Visualization Effectiveness. Journal of Visual Languages and Computing 13(3), 259–290 (2002)
9. Lynch, C., Ashley, K., Pinkwart, N., Aleven, V.: Argument diagramming as focusing device: does it scaffold reading? In: Proceedings of the AIED Workshop on Applications for Ill-Defined Domains, pp. 51–60 (2007)
10. Pinkwart, N., Aleven, V., Ashley, K., Lynch, C.: Toward Legal Argument Instruction with Graph Grammars and Collaborative Filtering Techniques. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 227–236. Springer, Heidelberg (2006)
11. Pinkwart, N., Aleven, V., Ashley, K., Lynch, C.: Evaluating Legal Argument Instruction with Graphical Representations Using LARGO. In: Proceedings of AIED, pp. 101–108. IOS Press, Amsterdam (2007)
12. Prettyman, E.B.: The Supreme Court's Use of Hypothetical Questions at Oral Argument. Catholic University Law Review 33, 555–591 (1984)
13. Reed, C., Walton, D., Macagno, F.: Argument Diagramming in Logic, Law and Artificial Intelligence. The Knowledge Engineering Review 22, 87–109 (2007)
14. Schank, P., Ranney, M.: Improved reasoning with Convince Me. Human Factors in Computing Systems. In: CHI 1995 Conference Companion, pp. 276–277. ACM, New York (1995)
15. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The Andes Physics Tutoring System: Lessons Learned. International Journal of Artificial Intelligence and Education 15(3) (2005)
16. Verheij, B.: Artificial Argument Assistants for Defeasible Argumentation. Artificial Intelligence 150, 291–324 (2003)

Automatic Multi-criteria Assessment of Open-Ended Questions: A Case Study in School Algebra

Élisabeth Delozanne[1], Dominique Prévit[2], Brigitte Grugeon[3], and Françoise Chenevotot[3]

[1]L'UTES - Université Paris VI - 4 pl. Jussieu - 75005 PARIS – France,
elisabeth.delozanne@upmc.fr
[2]LIUM-Avenue Laennec 72085 Le Mans Cedex 9– France,
dominique.previt@bretagne.iufm.fr
[3]DIDIREM - Université Paris VII - 2 place Jussieu - 75251 PARIS Cedex 5 – France,
brigitte.grugeon@amiens.iufm.fr,
francoise.chenevotot@lille.iufm.fr

Abstract. This paper deals with authoring assessments of complex competence involving open-ended questions. We present, PépiGen, a multi-criteria automatic assessor for school algebra, via a walkthrough of an example. PépiGen is based on our previous work on Pépite, an automatic cognitive diagnosis tool that capitalizes on educational research results. From that prototype, we derived patterns of diagnosis tasks. A pattern models (i) a class of exercises, (ii) the different students' points of view on the solutions reported in the literature or observed in a corpus, (iii) and a multidimensional assessment for each solution approach. To adapt an assessment to a specific classroom context (e.g. level of difficulty, time, learning objectives) an interface allows an IT non expert (e.g. a teacher) to generate new instances of exercises by filling the pattern parameters. The originality of our research lies in the fact that our system generates the automatic analysis of students' simple or complex answers, such as algebraic reasoning. This is an ongoing work but preliminary evaluation shows that PépiGen is already successful in generating and analyzing most answers on several classes of problems.

1 Introduction

The work reported here is part of an ongoing project, the Lingot project. Its objective is to design an intelligent aid that supports math teachers when they have to monitor learning in a classroom context, taking into account their students' cognitive diversity. This paper focuses on diagnosing students' cognitive profiles in algebra. It presents PépiGen, a system that generates Automatic Multi-criteria Assessments of students' competence in school algebra.

We first present the background, the objectives and the methodology we adopted to elicit patterns from the first Pépite assessment system used as a prototype. Then, we illustrate the modelling language we defined by describing an example of pattern of diagnosis tasks involving open-ended questions. The next section describes PépiGen, the system that allows a user to generate diagnostic tasks that instantiate patterns. We

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 101–110, 2008.

end with a discussion of our work in comparison with related works and with a summary of contribution and plans for future research.

2 Background

The key point of our assessment approach is that students' answers to problems are not simply interpreted as errors or as lack of skills but as indicators of incomplete, naive and often inaccurate conceptions that the students themselves have built. A fine analysis of the students' work is required to understand the coherence of the personal conceptions, to develop or to strengthen right conceptions, and to question wrong or unsuitable ones that interfere with, and sometimes prevent learning [1]. Detecting these conceptions is a very complex task that requires special training and a lot of time. ITSs can be a very helpful aid for teachers to reveal implicit conceptions which are very difficult to access without automatic reasoning on students' performance. Designing such systems is not trivial; especially when the student's input is not very constrained.

We developed such a cognitive diagnosing tool, derived from Educational Research [6], called Pépite, and we tested it in real settings [3]. This previous work aimed to prove that it was possible to automatically build a rich student cognitive profile from data collected after the student solved a set of tasks especially designed for that purpose. These tasks involved preformatted answers and open-ended answers. Like in other systems [5], in Pépite, the diagnosis is a three stage process. First, a local diagnosis provides, for each student's answer, a set of codes referring to the different criteria involved in the question. A code gives an interpretation of the student's answer according to a set of 36 *criteria* on six *assessment dimensions* (see section 4 for an example). Second, Pépite builds a detailed report of the student's answers by collecting the same criteria across the different exercises to have a higher-level view on the student's activity. At this stage, the diagnosis is expressed by success rates on three *components of the algebraic competence* (usage of algebra, translation from one representation to another, algebraic calculation) and by the student's *strong points* and *weak points* on these three dimensions. This level is called *personal features* of the student's cognitive profile. Third, Pépite evaluates a level of competence in each component with the objective to group of students with "equivalent" cognitive profiles. This level is called the *stereotype* part of students' profiles. Stereotypes were introduced to support the personalization in the context of whole class management and to facilitate the creation of working groups [4].

3 The PépiGen Project

In the present stage of the project, the aim is to offer an authoring tool, called PépiGen, to generate different Pépite-like diagnosis tools adapted to different school contexts and teachers' objectives. We had a lot of feedback from teachers who used the previous Pépite tools [3]. One of their points was that Pépite was interesting for a given school level. But teachers would need a database of diagnosis exercises to use Pépite-like tools at other school levels. Most teachers asked for off-the-shelf diagnosis

material, arguing that their job was to monitor learning, not to author materials. Some asked for assessments that can be tuned to specific contexts. Very few asked to define their own exercises but they asked to do so with no programming at all. These observations are confirmed by [16] in a state of art review of ITS authoring tools.

Thus, the work reported here describes how to build *banks of exercises supporting the diagnosis*. We focussed on the following design scenario: a teacher chooses a prototypic exercise in the bank and, if need be, asks for another equivalent one retrieved from the bank, or adapts the statement of the exercises by filling in forms (Cf. 6.1). In order to achieve this objective, in this paper, we investigate two research questions:

1. How to derive patterns of diagnostic tasks from the first Pépite prototype?
2. How to generate the procedure to analyze open-ended questions when (most) current technology restricts to preformatted answers?

From a computational point of view, the most difficult problem to be solved was to design and implement a system that assesses open-ended answers, both generic enough to apply to many classes of algebraic problems, and specific enough to detect students' personal conceptions. With open-ended questions, it is impossible to predict every student's answers. Thus the main points in our design are (i) to anticipate most current students' solution approach to one type of question by detailed and accurate epistemological and empirical studies, and (ii) to generate a set of answers representing each solution approach.

Our research approach is a bottom-up approach informed by educational theory and field studies. In previous work, we started from a paper and pencil diagnosis tool grounded in mathematical educational research and empirical studies [1, 6]. Then we automated it in a prototype called Pépite and tested it with dozens of teachers and hundreds of students in different school settings [3]. In the present research, we generalize this first design to create a framework for authoring similar diagnosis tools offering configurable parameters and options.

4 An Example of Diagnosis Task Pattern

Let us take a prototypic exercise from the original Pépite involving an open-ended question (Fig. 1). The objective of this exercise is to have deep insight in the student's algebraic thinking and to assess her/his skills and conceptions in the six dimensions of algebraic competence: (i) Validity, (ii) Meaning of Letters, (iii) Algebraic Writing, (iv) Translation (ability to switch between various representations: graphical, geometrical, algebraic, natural language), (v) Type of Justifications ("proof" by example, proof by algebra, proof by explanation, "proof" by incorrect rule), (vi) Numerical Writing.

Table 1 shows four examples of students' answers and their coding in Pépite. In those examples we can notice that no students' solutions are fully correct, but we can suspect very different levels of development in their algebraic thinking. Of course, building a cognitive profile from one answer is not reliable, but we can hypothesize that these students will benefit from different learning activities [4].

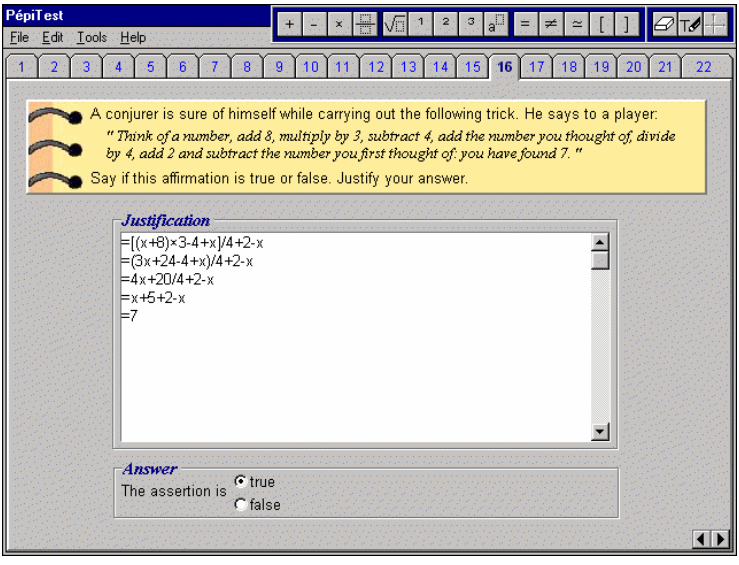


Fig. 1. A Pépite prototypic exercise of the "Proof and calculation process" pattern

To clone this exercise for a lower school level, we considered the following design scenario. An author is presented with the prototypical exercise and changes the italic sentence in the statement by the following one (statement 2): *Think of a number. Add 6 to this number multiply the result by 3, subtract three times your number to the result. You find 18.* This statement is a parameter of the pattern.

The system generates the algebraic expression, here (x+6)*3-3*x. The difficulty is to generate the anticipated solutions and their coding. In this type of task, [6] distinguished mainly four approaches for students to justify their answer:

1. An algebraic approach involving several processing types
   a. A correct translation in algebra by a global expression with correct/ incorrect use of parenthesis and an optimal-correct/non optimal correct/incorrect reduction to a number (7 or 18 in the examples);
   b. A partially correct translation to algebra using a step-by-step translation with correct/incorrect reduction to a number;
   c. An incorrect translation where the equal sign is not an equivalence sign between numbers.
2. A numerical approach where the student takes one or several examples involving the same types of processing as in the algebraic one;
3. A combination of both approaches where the student tries an algebraic proof but does not succeed and falls back on numerical examples to justify;
4. A justification in natural language.

In Table 1, Laurent's and Karine's solutions are examples of the first approach, while Khemarac's and Nicolas's are examples of the second one. For each solution approach and processing type, PépiGen, generates a corresponding set of algebraic

Table 1. Types of students' answers and their multidimensional coding in Pépite (age 15 or 16)

Khemarak	Nicolas	Karine	Laurent
Soit 5 un nombre ((5+8)×3-4+5)/4+2-5=7 ? ((13)×3-4+5)/4+2-5=7 ? (39-4+5)/4+2-5=7? 10+2-5=7 ? 10-3=7 ? 7=7 ? Oui donc cela marche (Yes thus it works)	3 + 8 = 11 11 × 3 = 33 33 - 4 = 29 29 + 3 = 32 32/4 = 8 8 + 2 = 10 10 - 3 = 7	x + 8 = 8x 8x 3 × 8x = 24+3x= 27x 27x-4 = 23x 23x+x=24x 24x/4=6x 6x+2=8x 8x-x=7	=[(x+8)×3-4+x]/4+2-x =(3x+24-4+x)/4+2-x =4x+20/4 + 2-x =x+5 +2-x =7
Justification by example (J2)	Justification by example (J2)	Justification by school authority (J4)	Justification by algebra (J1)
Valid translation in algebra (T1). Global expression with parenthesis, expressions are seen as a whole	Partially valid translation(T2). Step- by-step translation, expressions are seen as a process	Algebra is use to abbreviate (T4). The = sign announces a result, not an equivalence	Valid translation in algebra (T1). Global expression with parenthesis, expressions are seen as a whole
Correct numeric writing rules (NWR1)	Correct numeric writing rules (NWR1)	Incorrect identification of operation (AWR4); incorrect algebraic rules : $x + a \rightarrow x\ a$ $a\ x \pm b \rightarrow (a \pm b)$ $a\ x - x \rightarrow a - 1$	Incorrect use of parenthesis with memory of the meaning (AWR31)
No use of letters (L5)	No use of letters (L5)	Use of letters to calculate with incorrect rules (L3)	Correct use of letters (L1)
Invalid answer (V3)	Invalid answer(V3)	Invalid answer (V3)	Invalid answer (V3)

expressions. It associates a set of codes that characterizes the algebraic processing type from a diagnosis point of view.

One pattern describes the original exercise and the exercise generated by statement 2. The pattern *name* is: "Proof and calculation process". The two exercises are "similar" because the *interface*, the set of words to express the statement (see the "palette" Fig. 2), the diagnosis *objective*, the *anticipated solving approaches*, and the *set of possible codes* involved are all the same.

The differences between a clone and the prototypic exercise are the statement, the algebraic expression that translates the statement in algebra, and the complexity of this algebraic expression (level of parenthesis, number of operators, and number of division). The statement and the algebraic expression are *parameters* of the patterns and the three indicators for the complexity are *parameter characteristics*. These characteristics will be used to query the database and to tune a test to a school level. The parameters may be constrained. In the example, there is one *constraint*: the algebraic expression is reduced in a constant or a linear function; otherwise the diagnosis task

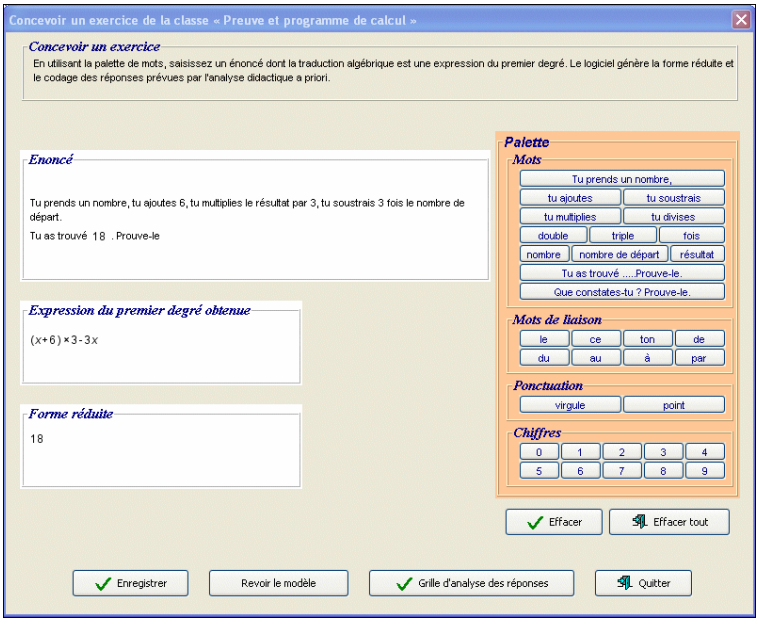


Fig. 2. Parameters setting for the "Proof and calculation process" patterns of diagnostic task

would change. The differences in the diagnosis part of the exercises are expressions representing *optimal correct solutions*, *non optimal correct solutions*, *partially correct solutions*, *incorrect solutions*. Each solution is characterized by a *comment*, a *code*, one or several *expressions* and correct or incorrect *rules*.

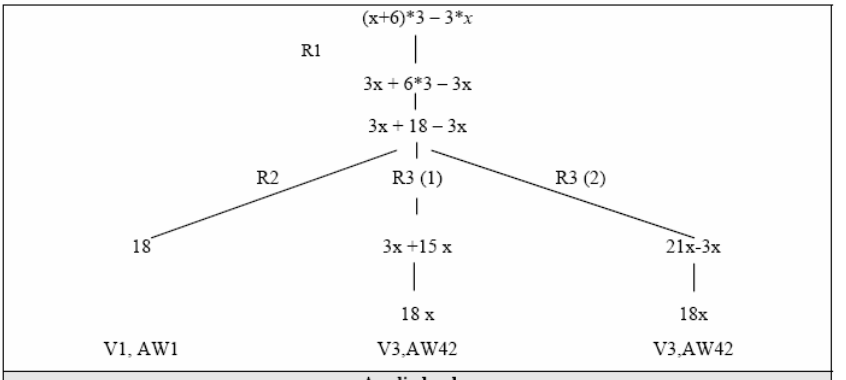
5 How to Generate a Diagnostic Task from a Pattern?

PépiGen is implemented in Java. It creates, initializes and saves, in an XML database, instances of the different classes representing the dynamic part of a pattern of diagnostic tasks. The static part is described by an XML schema. A *diagnostic task* consists of an exercise (problem statement and questions), a set of correct or incorrect anticipated solutions, and a set of codes that characterizes each solution from a cognitive diagnosis point of view. It is generated by PépiGen once the parameters of a pattern are set. Thus generating a diagnostic task is a two stage process: setting the parameters and generating the solutions tree and the coding for each branch. Data generated are stored in XML files and retrieved at run time to generate the student interface and to assess the student's answer.

When very constrained, the parameters are automatically generated by PépiGen (e.g. a formula to be instantiated with integer values between 1 and 20). This mode is called *automatic parameter setting*. But, for more complex patterns, the parameters are set by a human author (a teacher, a teacher trainer or a researcher). This mode is called *aided parameter setting* (e.g. Fig. 2).

When human authoring is required to set the parameters, PépiGen provides a Graphical Interface to enter the parameters. The author enters one parameter, the statement in natural language using the palette on the right side of the screen, and PépiGen generates the other parameters (the corresponding global algebraic expression and its reduced form), and displays them on the left of the screen. A software component based on a grammar and a finite state machine is used to interpret users' input in a constrained natural language and to translate it into algebra. This component is also used for analysis of students' input in other diagnosis tasks.

When parameters are set, a procedure specific to the pattern is called by PépiGen, to automatically generate all the information necessary to diagnose the students' answers to the exercise. This procedure is simple when answers are preformatted. In case of open-ended questions involving the dimensions "Algebraic calculation" or "Numerical calculation" in the pattern description, a software component, called Pépinière, builds a tree representing all anticipated solutions to the exercise and codes each solution on several dimensions.



Applied rules		
Correct rules	R1 : (A+B)C → AC+BC R2 : AC+BC→ (A+B)C	
Incorrect rules	R3 : A+BC→ (A+B)*C Applied to : (1) 18 – 3x (2) 3x +18	
Codes		
Validity	V1 : correct	V3 : incorrect
Algebraic Writing	AW1 : Correct use of rewriting rules	
	AW42 : The student gathers the terms in an algebraic sum.	

Fig. 3. Anticipated algebraic solutions for the clone example

Pépinière is a specific Computer Algebra System (CAS) dedicated to interpreting and generating students' algebraic input according to an epistemological and didactical analysis. It is independent of the different patterns. It relies only on mathematical foundations (mainly parsing of mathematical expressions, unification theory, algebraic rewriting rules), and on the multidimensional model of algebraic competence that grounded the Pépite project (i.e. on a set of multidimensional criteria represented

by a code, an extensible structured set of rewriting rules and a set of heuristics to prevent infinite loops). [14] presents a detailed description of Pépinière. In the present section, we just describe our general approach to automatic generation of the diagnosis by illustrating it with one example of pattern instantiation. In this example, the procedure to generate the coding instructions file is a two step process.

First, Pépinière builds a tree with every anticipated solution. It applies correct and incorrect reduction and developing rules. Heuristics are used to tackle the difficult problems of combinatorial explosion and infinite loops [14]. Fig. 3 shows the tree generated from the algebraic expression parameter that characterizes the clone (x+6)*3-3*x.

Second, the tree is walked in a way specified by the type of approach (algebraic/numeric). Each node (expression and rule applied) is saved along with the coding. For instance, correct solutions are generated by saving the nodes in walking through the tree considering only the correct rules. Incorrect solutions with an algebraic approach and a correct translation to algebra are generated by saving the nodes with incorrect rules. For incorrect solutions with a step-by-step translation Pépinière is called recursively with expressions generated by the preceding step. Incorrect solutions with a numerical approach are generated in the same way.

After students passed the test, the diagnosis system asks Pépinière to compare one expression in the student's answer to the expressions in the coding prescription file. To this end, Pépinière builds trees representing the expressions and tests the equivalence of the expressions regarding the commutability and associability of the operators.

6 Tests

Since PépiGen is still in the development phase it is difficult to have usability tests in real settings with teachers. Thus, we describe here a primary evaluation round. First we tested PépiDiag on a corpus of answers collected with the prototypic exercise (N=353) and its clone (N=39) presented in section 5. The system coding was validated by two educational researchers (the third and fourth authors). They agreed 100%. This means that PépiGen implementation is conform to the educational research model PépiGen is based on. Then, we asked three mathematics teachers to generate clones with PépiGen. They understood the potential of the system and found it easy to create exercises. They were satisfied with the solutions generated. We also tested Pépinière to generate solutions for other patterns involving simpler algebraic reasoning [17].

7 Related Work

Assessment and student modeling is a hot research topic in ITS and the e-learning community. We are especially interested in assessment modeling approaches and particularly in assessment of mathematical skills involving open-ended questions.

The leading specification for assessment is QTI, developed by IMS Global Learning Consortium [8]. The primary goal of this specification is interoperability between Learning Management Systems but it is limited to multiple-choice items and their

variations. [9] provide a broader conceptual model for assessment allowing the use of several assessment instruments (e.g. portfolio assessment or peer-assessment) and several types of assessment (e.g. multi-dimensional assessment). It is a first step to integrate QTI and IMS-LD specification. A perspective of our work could be to test their model by translating to their Item Construction Model, our conceptual model of diagnostic task patterns exemplified in section 5. But, so far it is unclear for us, if their model can represent both correct and incorrect conceptions. Moreover, as far as we know, it is a descriptive model and there is no implementation. In section 6, we presented through a worked example, a domain specific implementation corresponding to the "response rating part" of their model.

Many ITS or e-learning systems focus on math education and implement student's modeling or assessment authoring tools. Some of them analyse open answers when they are numerical or reduced to a single algebraic expression (Algebra Tutor [10], Assistment [2], LeActiveMath [11]). Very few analyse a whole reasoning. From this point of view, closely related to our work are Diane [7], Andes [15], and Aplusix [12].

Diane is a diagnosis system to detect adequate or inadequate problem solving strategies for some arithmetic classes of problem at elementary school level. Like Pépite, it is based on a very precise cognitive analysis. For each isomorphic class of problems, Diane analyses open-ended numerical calculation according to several criteria. It is very efficient compared to human assessment by experts. However, for more complex domains such as Physics or Algebra, researchers had to use a standard CAS or to develop one, specific to the type of students' inputs and to the type of diagnosis needed in the project.

For instance, Aplusix is a micro-world devoted to algebra learning in secondary schools, widely used in actual classrooms in France and in other countries. A teacher generates problems from different patterns of algebraic expressions for several tasks (e.g. factorisation, equation). Aplusix provides a very fined grained analysis of students' use of algebraic rewriting rules. PépiGen diagnosis is not so deep in the algebraic writing dimension but assesses a broader panel of skills on five other dimensions because the objective is to link formal processing with other students' conceptions like meaning of letters or meaning of algebra. Thus, in the Lingot project, there are very different diagnosis tasks involving algebraic expressions but also geometric figures and calculation programs.

8 Conclusion

In this paper we presented an approach to design and implement Automatic Multi-criteria Assessment of open-ended questions in early algebra. Our approach balances between very specific and rigid off-the-shelf tools and heavy generic authoring tools [16]. We benefited from empirical and theoretical educational studies to model patterns of diagnostic tasks. We designed and partially implemented the PépiGen system that automatically generates the diagnosis tasks after the parameters have been set. A specific CAS, Pépinière, generates all the students' reasoning usually observed in math class and assesses them with multi-dimensional criteria. PépiGen is a significant step toward an interactive assessment authoring tool in Algebra to support teachers in addressing their students' difficulties more effectively. Although the first PépiGen

testings are encouraging, there is still much work to be done. We are currently completing the system development by implementing automatic diagnosis on reasoning on other classes of algebraic problems (e.g. equation solving). We are also investigating with educational researchers how learners themselves can benefit from the Pépite diagnosis.

The software component we implemented to analyze answers to open-ended questions is inevitably domain dependant, but we propose a model to describe pattern of diagnosis tasks derived from educational research that could apply to many problem solving assessments using explicit criteria on several dimensions of evaluation.

References

1. Artigue, M., Assude, T., Grugeon, B., Lenfant, A.: Teaching and Learning Algebra: approaching complexity through complementary perspectives. In: ICMI Study Conference, Melbourne, pp. 21–32 (2001)
2. http://www.assistment.org/
3. Delozanne, É., Prévit, D., Grugeon, B., Jacoboni, P.: Supporting teachers when diagnosing their students in algebra. In: AIED supplementary proceedings, pp. 461–470 (2003)
4. Delozanne, É., Vincent, C., Grugeon, B., Gélis, J.-M., Rogalski, J., Coulange, L.: From errors to stereotypes: Different levels of cognitive models in school algebra. E-learn, 262–269 (2005)
5. Delozanne, É., Le Calvez, F., Merceron, A., Labat, J.-M.: A Structured set of Design Patterns for Learners' Assessment. JILR 18(2), 309–333 (2007)
6. Grugeon, B.: Design and development of a multidimensional grid of analysis in algebra. RDM 17(2), 167–210 (1997)
7. Hakem, K., Sander, E., Labat, J.-M.: DIANE, a diagnosis system for arithmetical problem solving. AIED, 258–265 (2005)
8. IMS Question & Test Interoperability, http://www.imsglobal.org/question/index.cfm
9. Joosten-ten Brinke, D., van Bruggen, J., Hermans, H., Burgers, J., Giesbers, B., Koper, R., Latour, I.: Modeling assessment for re-use of traditional and new types of assessment. Computers in Human Behavior 23(6), 2721–2741 (2007)
10. Koedinger, K.R., Anderson, J.R.: Intelligent Tutoring Goes to School in the Big City. IJAIED (8), 30–43 (1997)
11. LeActiveMath project, `http://www.leactivemath.org/`
12. Nicaud, J.F., Chaachoua, H., Bittar, M.: Automatic calculation of students' conceptions in elementary algebra from Aplusix log files. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 433–442. Springer, Heidelberg (2006)
13. Murray, T.: An Overview of Intelligent Tutoring System Authoring Tools: Updated analysis of the state of the art. In: Authoring Tools for Advanced Technology Learning Environments, ch.17, Kluwer Academic (2003)
14. Prévit, D.: PépiGen, un générateur de batteries d'exercices pour un diagnostic cognitif en algèbre élémentaire, PhD dissertation (to appear)
15. Shapiro, J.: An Algebra Subsystem for Diagnosing Students' Input in a Physics Tutoring System. IJAIED 15, 205–228 (2005)

Why Tutored Problem Solving May be Better Than Example Study: Theoretical Implications from a Simulated-Student Study*

Noboru Matsuda[1], William W. Cohen[2], Jonathan Sewall[1], Gustavo Lacerda[2], and Kenneth R. Koedinger[1]

[1]Human-Computer Interaction Institute
[2]Machine Learning Department
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh PA 15213 USA
{mazda,wcohen,sewall,gusl,koedinger}@cs.cmu.edu

Abstract. Is learning by solving problems better than learning from worked-out examples? Using a machine-learning program that learns cognitive skills from examples, we have conducted a study to compare three learning strategies: learning by solving problems with feedback and hints from a tutor, learning by generalizing worked-out examples exhaustively, and learning by generalizing worked-out examples only for the skills that need to be generalized. The results showed that learning by tutored problem solving outperformed other learning strategies. The advantage of tutored problem solving was mostly due to the error detection and correction that was available only when skills were applied incorrectly. The current study also suggested that learning certain kinds of conditions to apply rules only for appropriate situations is quite difficult.

Keywords: Intelligent Authoring System, Simulated Student, Programming by Demonstration, Machine Learning, Cognitive Tutor.

1 Introduction

SimStudent is a machine-learning agent that learns cognitive skills by generalizing solutions demonstrated [1] and also by being tutored as we describe in this paper. Our original motivation to develop SimStudent was to automate cognitive modeling to author a Cognitive Tutor that deploys *model tracing* to provide individualized feedback and contextualized help [2]. To perform model tracing, the Cognitive Tutor needs a *cognitive model* that represents domain principles. However, cognitive modeling is a labor-intensive task that requires significant knowledge and experience in cognitive task analysis and AI-programming. Embedded into Cognitive Tutor Authoring Tools (CTAT [3]), SimStudent acts as an intelligent building block that allows authors to perform *authoring by demonstration*, where authors merely demonstrate

* The research presented in this paper is supported by the National Science Foundation Award No. REC-0537198. This work was also supported in part by the Pittsburgh Science of Learning Center, which is funded by the National Science Foundation Award No. SBE-0354420.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 111–121, 2008.

how to solve problems (correctly and incorrectly) instead of writing a cognitive model by hand. SimStudent generalizes demonstrations and create a set of production rules that reproduce the problem-solving steps demonstrated.

A critical research question addressed in this paper is about the efficiency of SimStudent: How can SimStudent be taught most effectively?

Originally, SimStudent was a "*passive*" learner in the sense that SimStudent attempted to generalize every problem-solving step demonstrated, but did not attempt to perform problem-solving steps on its own. SimStudent could reduce the learning load by *selectively choosing* certain steps to generalize; for instance, generalizing a step only when SimStudent does not have a production rule that reproduces the step demonstrated. Assuming that applying an existing skill is easier than learning a new skill, this learning strategy might require a relatively shorter learning time to achieve the same quality of cognitive model. A third possibility is that SimStudent could actively solve problems, rather than explaining demonstrations, and get feedback. Since the author will see SimStudent performing actions, which provides a chance to explicitly correct errors, this tutoring strategy might outperform passive or the selective learning strategies.

In this paper, we compare three learning strategies to answer the following research question: *Which learning strategy is better in terms of efficiency of training and quality of resulting cognitive models?* Answering this question is not only important for authoring purposes, but it may also provide us theoretical insights into understanding human learning by inspecting SimStudent's learning processes and learning outcomes, which are not easily attainable in human subjects.

2 SimStudent: A Machine-Learning Agent You Can Teach

An actual image of the Cognitive Tutor used in the current study is shown in **Fig. 1**. Suppose that an author is trying to build a Cognitive Tutor for Algebra equation solving. The author has just built the Tutor interface shown in **Fig. 1** by using CTAT. Now, the author launched SimStudent to create a cognitive model for equation solving by using the Tutor interface and solves a few problems.

2.1 An Example Cognitive Tutor: Algebra Equation Tutor

In this tutor, equations are represented with a mathematical operation to transform a given equation to another form. To transform an equation, an operation must be specified first, followed by the left-hand and right-hand sides of the resultant equation being entered in the adjacent row. **Fig. 1** shows that the author has decided to "add -1" to both sides, and the left-hand side has just been entered. In sum, a single equation-solving step (e.g., transforming "`3x+1=x+4`" into "`3x=x+3`") is modeled as *three steps* – (1) selecting an operation for transformation, (2) entering an expression for the left-hand side, and (3) for the right-hand side. The first step is called *transformation step*, and the last two steps are called *type-in steps*. In this paper, the word "step" means one of these three steps. An operation for transformation must be specified prior to entering any expressions. The order of entering sides can be arbitrary, but both sides must be entered before selecting the next operation. The skills to select an appropriate operation are called *transformation skills*, and the skills to enter left- and right-hand sides are called *type-in skills*.

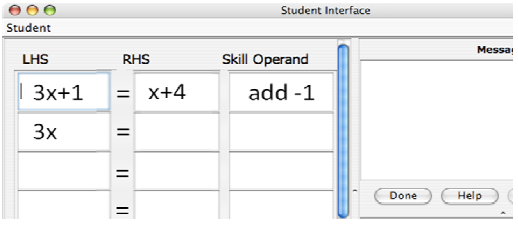


Fig. 1. The Tutor Interface for the Algebra I CTAT Tutor. Students are supposed to enter an operation for transformation first in the column labeled as "Skill Operand." Then corresponding expressions for the left- and right-hand sides must be entered.

2.2 Learning Production Rules by Demonstration

Each of the production rules represents an individual skill to perform a particular step. Performing a step is modeled as generating a tuple that consists of an *action* taken (e.g., "entering some text"), a place that was *selected* to take the action (e.g., "the second cell in the first column"), and the value that was *input* as a result of taking the action (e.g., the string "*3x*"). Those are called *action*, *selection*, and *input*. A tuple of <selection, action, input> is called an *SAI tuple*.

A production rule models a particular skill in terms of *what*, *when*, and *how* to generate a particular SAI tuple. In other words, a production rule shows that "To perform a step, first look at X and see if a condition Y holds. If so then do Z." The part of the production rule representing X (*what*) is called the *focus of attention* that specifies particular elements with certain constraints like "the cell in the table" shown in the Tutor interface. The part of the production rule representing Y (*when*) is called the *feature tests*. The feature tests represent a set of conditions that must hold about the focus of attention – e.g., the two cells must be in the same row, the expression in the cell must be polynomial, etc. Together, the focus of attention and the feature tests compose the left-hand side (i.e., the condition part) of a production rule. The right-hand side (i.e., the action part) of a production rule contains a sequence of *operations* that generates the value of the input in the SAI tuple.

Prior to learning, SimStudent is given a hierarchical structure of the elements in the Tutor interface with which to express the constraints among the focus of attention, a set of *feature predicates* with which to express feature tests, and a set of *operators* with which to compose a sequence of operations. SimStudent has a library of feature predicates and operators that are general for arithmetic and algebra, but the authors might need to write domain-specific background knowledge to use SimStudent for other domains.

When demonstrating a step, the author first needs to specify the focus of attention by double-clicking the elements on the Tutor interface. Then he/she performs a step, namely, takes an "action" upon a "selection" with an appropriate "input" value. Finally, the author needs to label the demonstrated step. This label is called the *skill name*.

When a step is demonstrated for a particular skill K with a focus of attention F and an SAI tuple T, the pair <F, T> becomes a *positive example* of the skill K. The pair <F, T> also becomes a *negative example* for all other skills. This indicates to "apply

skill K to carry out the SAI tuple T when you see the focus of attention F, but do not apply any skills other than K when you see F." We call this kind of negative examples the *implicit negative examples* as opposed to the explicit negative examples used for tutored problem solving, which is described in the next section. Once a positive example is acquired, it stays as positive throughout a learning session. On the other hand, an implicit negative example for a skill would later become a positive example if the same focus of attention is eventually used to demonstrate that skill.

When a new positive or negative example is added for a particular skill, SimStudent *learns* the skill by generalizing and/or specializing the production rule for the skill so that it applies to all positive examples and does not apply to any negative examples. The focus of attention is generalized so that they are consistent with all instances of the focus of attention appearing in the positive examples. An example generalization is to shift from "first column" to "any column." Feature tests are generalized and/or specialized so that they cover all positive examples and no negative examples that is done by Inductive Logic Programming [4] in the form of Foil [5]. The operator sequence is generalized so that it generates "input" values from the focus of attention for all SAI tuples in the positive examples.

2.3 Learning Strategies

The original version of SimStudent always learns skills whenever a step is demonstrated by generalizing existing skills or introducing a new skill. This can be seen as a model of human students diligently learning skills from worked-out examples, regardless of what they already can do (although it sounds too idealistic).

As an interesting twist (and a step towards a more realistic model), the author can also have SimStudent try to "*explain*" the step demonstrated, by identifying a previously learned skill that replicates the step demonstrated, and having SimStudent learn skills only when it fails to explain the step. This is analogous to human students learning from worked-out examples while self-explaining the solutions.

Furthermore, the author can instead *tutor* SimStudent on how to solve problems. The author provides problems to SimStudent, lets SimStudent solve them, and provides feedback on each of the attempts made. When SimStudent makes an error, the author can provide negative feedback, which will motivate SimStudent to accumulate an *explicit negative example*– i.e, it will learn when *not* to apply a skill because it produces an incorrect output. When SimStudent has no rules indicating how to perform a step, the author provides a "hint" on what to do next; this hint is just a demonstration of how to perform the step. This is a model of learning by tutored problem solving.

In summary, we implemented these three learning strategies for SimStudent:

Diligent Learning – provides demonstrations on every step and SimStudent learns skills each time a step is demonstrated.

Example Study – provides demonstrations on every step and SimStudent attempts to identify a production rule that reproduces the step demonstrated. Only when the attempt fails, does SimStudent learn skills.

Tutored Problem Solving – provides SimStudent with problems to solve. For each step, SimStudent is asked to show *all rule applications* that can be done. For each of the rule applications, SimStudent gets flagged feedback from an *oracle*,

which merely tells the correctness of the rule application. Correct rule applications become positive examples and incorrect ones become negative examples. When there is no correct rule application for a step, SimStudent asks a what-to-do-next hint to the oracle. The oracle then demonstrates to SimStudent how to perform the step.

The oracle for the Tutored Problem Solving can be either a human or another computer program. In the current study, we used the commercially available Cognitive Tutor, Carnegie Learning Algebra I Tutor, as the oracle. The details follow.

3 Learning Strategy Study

This section describes a study conducted to evaluate the efficiency of each of the three learning strategies described in section 2.3.

3.1 Method

Three versions of SimStudent were implemented – one for each of the three learning strategies. Each SimStudent was trained with 20 problems and tested with ten problems. Since hundreds of steps must be demonstrated and tested to complete the study, it was not realistic to ask human authors to be involved in the study. Instead, we used pre-recorded and machine-generated demonstrations as described below.

The pre-recorded demonstrations were collected from a previous classroom study conducted in the PSLC LearnLab.[1] In the LearnLab study, the Carnegie Learning Algebra I Tutor was used in an urban high-school algebra class. The high-school students were asked to use the Algebra I Tutor individually. The students' activities were logged and stored into a large database, called DataShop.[2] We then extracted problems and human students' *correct* steps from DataShop for the current study. An entire (correct) solution for a particular problem made by a particular student became a single training problem for the Example Study condition and the Diligent Learning condition. The problems were randomly selected from the DataShop data.

For the Tutored Problem Solving condition, SimStudent was tutored by the Carnegie Learning Algebra I tutor. That is, when SimStudent got stuck, SimStudent asked a what-to-do-next hint to the Carnegie Learning Algebra I tutor, and the Carnegie Learning Tutor provided a precise instruction for what to do in the form of the *bottom-out hint*, which provides the same information as the SAI tuple. Whenever SimStudent performed a step, each of the rule applications was sent to the Carnegie Learning Algebra I Tutor to get a flagged feedback.

There were five disjoint sets of training problems (i.e., the total of 100 training problems). Thus, there was a total of 15 experimental sessions (five training sets for each of the three learning-strategy conditions).

Each time SimStudent was trained on a new training problem, the production rules learned were tested with the ten test problems. The same set of test problems was used for all of the 15 experimental sessions. The test problems were also randomly collected from the LearnLab study. For each of the steps in a test problem, we asked

[1] www.learnlab.org
[2] www.learnlab.org/technologies/datashop

SimStudent which production rules can be fired. Since we wanted to know how *poorly* SimStudent solves problems in addition to how well, we recorded all possible rule applications for each step. More precisely speaking, for each step, we enumerated all production rules whose left-hand conditions hold. The correctness of a rule application was evaluated by the Carnegie Learning Algebra I Tutor. The steps performed by SimStudent were coded as *correct* if there was at least one correct rule application attempted. Otherwise, the steps were coded as *missed*.

3.2 Evaluation Metrics

We define a dependent variable, called the *Step score*, that represents how well the production rules learned were applied on individual steps in the test problems. A step is scored as zero if it was missed (i.e., no correct rule application was made – see the definition above). Otherwise, a step was scored as a ratio of the number of correct rule applications to the total number of rule applications applicable to that particular step. For example, if there were 2 correct and 6 incorrect rule applications for the step, then the Step score for that step is 0.25. The step score ranges from 0 (no correct rules applicable) to 1 (no incorrect rules applicable, and at least one correct rule applies). We define the *Problem score* as the average Step score for all steps in a test problem.

In general, there are several correct and incorrect rule applications available for each step. Since SimStudent does not have any strategy to select a single rule among these conflicting rule applications, the Step score can be seen as a probability that the step is performed correctly at the first attempt.

4 Results

4.1 Overall Learning Performance

Fig. 2 shows average Problem Score for each learning-strategy condition. The X-axis shows the number of training problems learned. The Problem score was aggregated across the ten test problems and the five training sets (i.e., average of the 50 Problem scores for each condition). All three conditions showed an overall improvement on the Problem score when more training problems were learned.

The three learning conditions improved equally on the first 8 problems. After that, the Tutored Problem Solving condition outperformed other conditions. There was a point, for all three conditions, where the improvement of the performance on the test problems diminished to almost nothing. After training on all 20 problems, the average Problem score was 0.78 for the Tutored Problem Solving, 0.72 for the Diligent Learning, and 0.66 for the Example Study. ANOVA revealed a main effect of the learning strategy; F=7.68, p<0.001. The paired t-tests showed that all three learning-strategy conditions are significantly different from each other. *The Tutored Problems-Solving condition outperformed the other two conditions on the Problem score. The Example Study was the least efficient learning strategy in terms of the Problem score.*

To further investigate why the Tutored Problem Solving condition led to better learning, we broke down the Step score (the basis of the Problem score) into two scores: (1) the *Precision* score showing the ratio of the number of correct to incorrect rule applications for a step, and (2) the *Recall* score showing the ratio of the number of steps that were performed correctly to the total number of steps in a test problem.

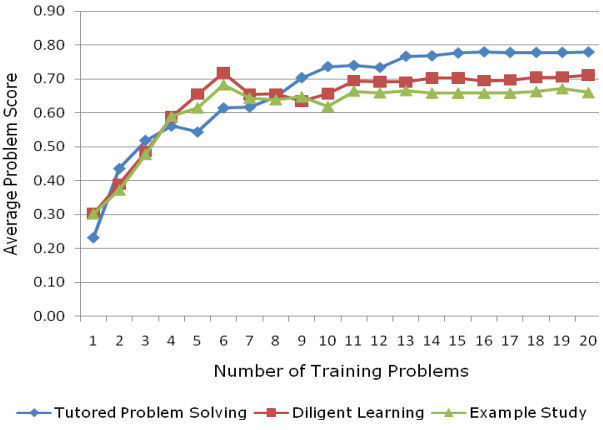


Fig. 2. Overall improvement of the Problem scores. The X-axis shows the number of training problems. The Y-axis shows the average Problem scores on the ten test problems, aggregated across five training sets.

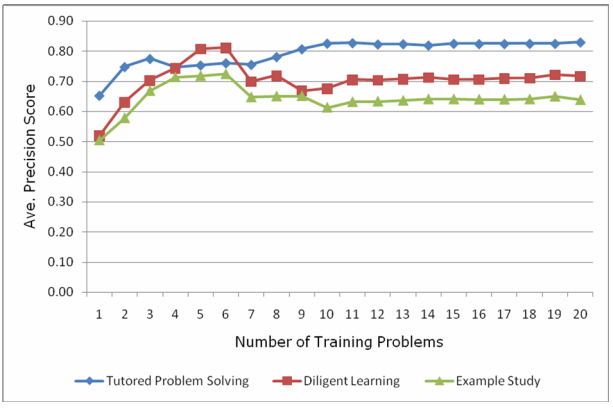


Fig. 3. Average Precision scores. The X-axis shows the number of training problems learned by the time the Precision score was measured.

Fig. 3 shows the average Precision score for the ten test problems aggregated across the training sets. On the 20th training problem, there was a main effect of the learning strategy; F=24.49, p<0.001. The paired t-tests confirmed that all three conditions are significantly different from each other. The Tutored Problem Solving condition outperformed other conditions on the Precision score. This means that *the production rules learned by Tutored Problem Solving were more likely to produce correct rule applications than the rules learned by other learning strategies.*

Fig. 4 shows the average Recall score. ANOVA showed a main effect of the learning condition; F=7.68, p<0.001. The paired t-tests showed that the Tutored Problem Solving was significantly inferior to the other two conditions (both t=2.01, p<0.001), but the difference between Example Study and Diligent Learning was not significant. The Tutored Problem Solving condition was significantly inferior to other two

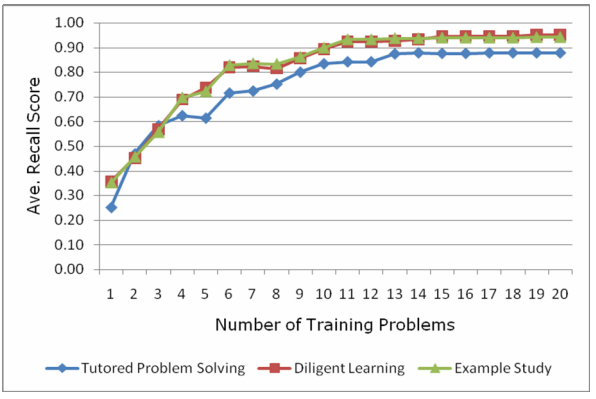


Fig. 4. Average Recall score. The X-axis shows the number of training problems.

conditions, meaning *the Tutored Problem Solving condition did not learn as many production rules necessary to solve test problems as other conditions did*. On average, the Tutored Problem Solving condition learned the fewest production rules (11.6), and the Diligent Learning condition learned the most (21.0). The Example Study condition learned 16.0 production rules on average.

4.2 Types of Errors

To see if there were any differences in the kinds of errors made by each learning condition, we categorized the errors appeared on the test problems. Regardless of the learning strategy, once the learning was saturated (i.e., after learning ten problems for Diligent Learning and Example Study, and 13 problems for Tutored Problem Solving), there were only two types of errors: (1) *Step-Skipping* error – attempting to apply a transformation skill without completing previous type-in steps, (2) *No-Progress* error – applying a transformation skill that does not make the transformed equation any closer to a solution (see section 0 for the definition of steps and skills).

An example of a Step-Skipping error is to apply another transformation skill to the situation shown in **Fig. 1**, and enter, say, "divide 3" into the rightmost cell on the second row when the middle cell (right-hand side of the equation) is left blank.

An example of a No-Progress error is to "subtract 2x" from `2x+3=5`. This is a mathematically valid step, but it does not make the resultant equation any closer to a solution.

Both Step-Skipping and No-Progress steps are considered as a wrong step by the Carnegie Learning Algebra I Tutor. Thus, SimStudent received negative feedback on both of these erroneous steps during tutored problem-solving.

No-Progress errors appeared in all three conditions. Quite interestingly, there were no Step-Skipping errors observed for the Tutored Problem Solving condition. Why? We hypothesized that only Tutored Problem Solving had a chance to revise incorrect skills during training, by making a Step-Skipping error and receiving negative feedback, which allowed SimStudent to accumulate negative examples to correctly learn LHS conditions. Namely, *making an explicit error and getting a flagged feedback on it (which, by definition, merely tells the correctness of the step) should have positively*

contributed to learning. To test this hypothesis, we controlled the creation of negative examples for the Tutored Problem Solving condition, which is described in the next section.

4.3 Control Experiment with No Explicit Negative Feedback

We have modified the Tutored Problem Solving condition, so that it does not generate negative examples for incorrect rule applications. SimStudent still received negative *feedback* for incorrect rule applications, thus another attempt was made to perform a step. This means that the modified version of Tutored Problem Solving still had the same amount of positive examples during training as the original version.

With this modification, the Tutored Problem Solving condition made the same Step-Skipping errors as the other conditions. Thus, *it was the* explicit *negative examples obtained by incorrect rule applications that caused the high Precision score for the Tutored Problem Solving condition*.

This modification did not affect the appearance of the No-Progress errors – having more negative examples did not prevent skills from being incorrectly generalized and making No-Progress errors.

5 Discussion

5.1 The Impact of Negative Feedback on Learning

The most important finding in the current study is that *the most effective way to train SimStudent is Tutored Problem Solving*. It is crucial for successful learning to allow SimStudent to commit itself to apply its own skills to solve problems – this is a natural way to give SimStudent *negative feedback explicitly* for the incorrect skill applications so that incorrect skills are appropriately generalized.

It is interesting to see that Example Study and Diligent Learning are superior to Tutored Problem Solving at some point on the fifth and sixth training problems (Fig. 2). This was mostly due to the high Recall scores – Example Study and Diligent Learning tend to learn more rules that correctly perform steps. However, at the same time, they also have a tendency to learn incorrect rules as well. Those incorrect rules can only be eliminated through explicit negative feedback.

Despite the importance of the negative examples, programming by demonstration in most cases only produces positive examples. Kosbie and Myers [6] emphasized the issue of *program execution* in the shared common structure of programming by demonstration. We further emphasize the importance of feedback about incorrect program execution in providing *explicit* negative examples. Interactive Machine Learning [7] is a good example of successful application of programming by demonstration where the learning agent can acquire negative examples explicitly through program execution.

5.2 Difficulty in Rule Induction

Another important lesson learned is *the difficulty of inductive learning*. It turned out that learning appropriately generalized rules that do not generate No-Progress errors is

challenging in this particular domain. Despite having explicit negative feedback on the No-Progress errors during training, the Tutored Problem Solving condition still made the No-Progress errors on the test problems.

Since No-Progress errors always generate mathematically valid steps (meaning, the RHS operator sequence is correct), the challenge is in learning LHS conditions – *learning* when *to apply a particular rule is more difficult than learning* how *to perform a step*. Since it is beyond the scope of the current paper, we do not further discuss this issue, but now *we have narrowed down the difficulty of inductive learning to learning conditions for when to apply rules*. This must be addressed further in future studies.

6 Conclusion

The empirical study showed that tutored problem-solving results in learning production rules more accurately than learning from examples for SimStudent's learning. Thus, for authoring purposes, *tutoring* SimStudent instead of *demonstrating* solutions may be a better form of using SimStudent as an aid to author Cognitive Tutors, assuming that providing feedback does not cost too much for the authors. In the 20 training problems, each skill was *demonstrated* 13.5 times on average for Diligent Learning and Example Learning, and 2.8 times for Tutored Problem Solving. For the Tutored Problem Solving, the tutor provided positive feedback 14.1 times and negative feedback 3.5 times on average throughout the 20 training problems. Future studies on the authoring cost analysis are necessary.

That tutored problem solving is significantly inferior to other learning conditions on the Recall score must be studied further. What about starting from the example study first and shifting to tutored problem solving later? This is a well-known learning strategy that is effective for human students [8]. All three conditions tied on the Step score for the first few training problems, and still the example study conditions were better on the Recall score on those steps. Thus, starting from an example study would allow SimStudent to acquire production rules more quickly, and switching to tutored problem-solving would provide good opportunities to correct these rules.

The current study also provides insight into future studies on inductive learning. Although, SimStudent has characteristics that are essentially different from human learning, finding out why some features are more difficult to learn than others would open the door for future studies on human and machine learning.

References

1. Matsuda, N., Cohen, W.W., Koedinger, K.R.: Applying Programming by Demonstration in an Intelligent Authoring Tool for Cognitive Tutors. In: AAAI Workshop on Human Comprehensible Machine Learning (Technical Report WS-05-04), pp. 1–8. AAAI association, Menlo Park (2005)
2. Koedinger, K.R., Corbett, A.: Cognitive Tutors: Technology Bringing Learning Sciences to the Classroom. In: Sawyer, R.K. (ed.) The Cambridge Handbook of the Learning Sciences, pp. 61–78. Cambridge University Press, New York (2006)

3. Koedinger, K.R., Aleven, V.A.W.M.M., Heffernan, N.: Toward a Rapid Development Environment for Cognitive Tutors. In: Hoppe, U., Verdejo, F., Kay, J. (eds.) Proceedings of the International Conference on Artificial Intelligence in Education, pp. 455–457. IOS Press, Amsterdam (2003)
4. Muggleton, S., de Raedt, L.: Inductive Logic Programming: Theory and methods. Journal of Logic Programming 19-20(Supplement 1), 629–679 (1994)
5. Quinlan, J.R.: Learning Logical Definitions from Relations. Machine Learning 5(3), 239–266 (1990)
6. Kosbie, D.S., Myers, B.A.: Watch what I do: programming by demonstration. In: Cypher, A. (ed.) Watch what I do: programming by demonstration, pp. 423–431. MIT Press, Cambridge (1993)
7. Fails, J.A., Olsen Jr., D.R.: Interactive machine learning. In: Proceedings of IUI 2003, pp. 39–45. ACM, Miami Beach (2003)
8. Renkl, A., et al.: From example study to problem solving: Smooth transitions help learning. Journal of Experimental Education 70(4), 293–315 (2002)

A Case Study Empirical Comparison of Three Methods to Evaluate Tutorial Behaviors

Xiaonan Zhang[1], Jack Mostow[1], and Joseph E. Beck[2]

[1] Project LISTEN, School of Computer Science, Carnegie Mellon University
[2] Computer Science Department, Worcester Polytechnic Institute

Abstract. Researchers have used various methods to evaluate the fine-grained interactions of intelligent tutors with their students. We present a case study comparing three such methods on the same data set, logged by Project LISTEN's Reading Tutor from usage by 174 children in grades 2-4 (typically 7-10 years) over the course of the 2005-2006 school year. The Reading Tutor chooses randomly between two different types of reading practice. In assisted oral reading, the child reads aloud and the tutor helps. In "Word Swap," the tutor reads aloud and the child identifies misread words. One method we use here to evaluate reading practice is conventional analysis of randomized controlled trials (RCTs), where the outcome is performance on the same words when encountered again later. The second method is learning decomposition, which estimates the impact of each practice type as a parameter in an exponential learning curve. The third method is knowledge tracing, which estimates the impact of practice as a probability in a dynamic Bayes net. The comparison shows qualitative agreement among the three methods, which is evidence for their validity.

Keywords: educational data mining, randomized controlled trials, learning decomposition, knowledge tracing, evaluating tutor strategies.

1 Introduction

The behavior of an intelligent tutor affects its efficacy, so it is important to evaluate. One reason is to improve the tutor as part of data-driven iterative refinement. Another reason is to draw lessons for what behaviors to embrace or avoid in designing other tutors. The obvious way to evaluate alternative tutorial behaviors is to perform a controlled between-subjects comparison of different versions of the tutor, with each version employing a different behavior. However, such experiments may require many students and considerable time to achieve statistically reliable results. Is there a better way?

Fortunately, intelligent tutors can log detailed, longitudinal interactions, and experimentally vary the behaviors that affect those interactions. Analyzing the resulting data lets us evaluate tutorial behaviors. Such evaluation can test whether a behavior works, gauge how well it works, and compare alternatives.

Previous research has employed various methods to perform such analyses, but we are not aware of any studies whose express purpose was to compare alternative methods for

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 122–131, 2008.

evaluating tutor behavior. To help fill this gap, we present a case study that applies three analysis methods to the same data set, described in Section 2. Sections 3, 4, and 5 respectively describe each method as applied to the data. Finally, Section 6 summarizes results, conclusions, and contributions.

2 Case Study: Evaluate Two Modes of Practice in a Reading Tutor

We carried out our case study on data from Project LISTEN's Reading Tutor, which helps children learn how to read [1]. The Reading Tutor and the student take turns to pick a story, which is then displayed line by line on a computer screen. The Reading Tutor listens to the student read the story aloud, and uses automatic speech recognition (ASR) to track the student's position in the text, detect (some) mistakes, and measure the time to read each word. The Reading Tutor also provides various forms of assistance when the student gets stuck, or clicks for help. It logs its interactions and speech recognizer output into a database.

The analysis problem in our case study is to compare two modes of practice for children who are still learning the letter-sound mappings of English. The Reading Tutor uses an instructional activity adapted from published interventions [2-5] to teach these mappings in the context of isolated words. To exercise taught mappings in the context of connected reading, the Reading Tutor then presents practice text in one of two modes – choosing randomly between them each time, but using the same text either way.

One mode of practice is assisted oral reading. In this mode, the Reading Tutor displays each successive story sentence, e.g., *Sam sat on the mat*, and listens to the child read it aloud, giving help as necessary.

In the other mode, called Word Swap, the Reading Tutor reads aloud, and the child provides feedback. Word Swap is based on an activity used by a human expert to teach children to attend to the correspondence between print and sound. First the Reading Tutor explains the task:

> *Good, careful readers make sure that what they say matches what they see. Let's play a game called Word Swap. The Reading Tutor will read the story to you, but it might read some words wrong. Click on the words that do not match what you hear!*

In Word Swap, the Reading Tutor picks a word at random from each sentence, e.g., *sat*, and replaces it with some other random word from the story, e.g., *am*. It displays the modified sentence, e.g. *Sam am on the mat*, but plays the narration of the original sentence, so as to deliberately "misread" the replaced word. (The Reading Tutor uses recorded human speech, so it is easier to modify the displayed text of the sentence than its spoken narration.) The student's task is to click on the "misread" word. When the student clicks on the "misread" word *am*, the Reading Tutor replaces it with the correct word *sat* and says *Right! This says am, not sat.* If the student clicks on a correctly read word, the Reading Tutor says, *no, the Tutor read that word right!*

Which is more effective – assisted reading or Word Swap? To study this question, we define "effective" in terms of how well students do on the words in the story when

they read them again later. We measure performance in reading an individual word (in context) based on how long the student takes to read the word, whether the student clicks on the word for help, and whether the speech recognizer accepts the word as read correctly. We compute this information from the Reading Tutor's log data. Ideally we would also measure how well the child attends to spelling-sound correspondence when reading the word – the goal of Word Swap. However, we have not defined or automated such a measure, in part because the very signs that may indicate such attention (slow reading and frequent self-corrections) may merely indicate poor reading.

The 2005-2006 Reading Tutor logged 2669 encounters of letter-sound practice passages by 174 students in grades 2-4. The 1311 encounters in assisted reading mode comprised 76,326 words. The 1358 instances of Word Swap totaled 83,421 words. To avoid ceiling effects, we exclude the most common 200 English words from the dataset, leaving 31,216 word encounters under assisted reading conditions, and 37,028 under Word Swap conditions, respectively. We now discuss the three methods we used to evaluate the effects of these encounters.

3 RCT Analysis

Randomized controlled trials (RCTs) manipulate experimental variables to test their effects on outcomes. Randomizing assignment to treatment ensures that statistically reliable effects are truly causal. Intelligent tutors can randomize tutorial decisions such as what type of practice, assistance, or feedback to provide, and log large numbers of randomized trials, as illustrated by experiments in the Reading Tutor [1] as well as other tutors [6]. Each trial has a context in which it occurred, the decision made, and its outcome [7]. Aggregating over many trials by many students lets us analyze how the decision affects the outcome.

The context of the RCTs analyzed in this paper is the point at which the Reading Tutor has just taught some letter sounds and the student encounters a word in a practice text. The decision is which mode of practice to give – namely, assisted reading or Word Swap. The Reading Tutor randomizes this decision within-subject and within-text. That is, each time the Reading Tutor finishes a letter-sound lesson, it makes this decision anew. Randomizing within-subject – that is, giving each student both types of practice – controls for individual differences among students. Likewise, randomizing within-text – that is, using the same set of texts for both modes of practice – controls for differences among texts. However, the Reading Tutor chooses the mode of practice for an entire text at a time, rather than for each individual word. We can treat the practiced words as separate trials, but they are not independent.

How to define outcome? To analyze which mode of practice results in better word learning, we define the outcome of each trial as the student's performance on a later encounter of the same word. (Practice on a word affects performance on that word much more than on other words [8].) If this encounter occurs in a story the student has read before, the student's performance may reflect remembering the story rather than reading the word. If the encounter occurs too soon, the student's performance may just reflect how recently the student or tutor has read the word. On the other hand, as time elapses, the trial's effect diminishes relative to other influences, such as classroom instruction.

Therefore we define its outcome as performance on the student's first encounter of the word 1-3 days after the randomized trial, provided it occurs in a new context.

As Section 1 explained, we measure performance on a word based on how long the student takes to read it, whether the student clicks on it for help, and whether the speech recognizer accepts it as read correctly. Table 1 defines the measures we use for RCT analysis. We represent undefined outcomes as null values.

Table 1. Outcome measures used in RCT analysis

Measure	Definition
Accepted	The speech recognizer (ASR) recognized the word as read correctly
Asked help	The student clicked on the word for help in reading it
Credited	True if the ASR accepted the word without the student receiving help; false if the ASR rejected the word or the student requested help; undefined (and excluded from RCT analysis) if the ASR accepted the word after tutor-initiated help that masked whether the student knew the word
Latency [9]	The delay from the end of reading the previous word until starting to say the current word
Reading time	Latency plus the time to say the word, with this sum capped at 3 seconds to deal with outliers
Adjusted time [10]	Reading time for credited word; 3 seconds for uncredited word; undefined if credit is undefined

Sources of variance in word reading performance include student, word, story, and practice mode. Since words differ more than students (C. Perfetti, personal communication), we compare practice modes paired by story and word. That is, for a story word encountered in both assisted reading and Word Swap (generally by different students), we compare performance on each word after one mode of practice versus after the other, averaged across students. We compute the difference in a performance measure M as M(Word Swap) – M(assisted reading). We use a t-test, paired by story and word, to test whether performance differs significantly by practice mode, so the degrees of freedom (253) are one fewer than the number of such words.

As Table 2 shows, this difference is significantly greater than 0 for latency, reading time, and adjusted time. The positive difference means that students read words significantly slower after Word Swap than after assisted reading practice. Whether this is good news or bad news for Word Swap depends on *why* they read slower: are they paying better attention? or did they just learn the words less well? We can't tell.

Because this comparison does not control for student identity, one possible confounding factor is the difference between students who get one type of practice and students who get the other. However, since treatment assignment is randomized anew for each passage for each student, and for each word is based on different randomized subsets of students across stories, we assume we can ignore differences between these subsets. To test this assumption, we verified that reading proficiency (measured by a paper pretest) and reading level (estimated by the Reading Tutor) did not differ significantly between treatment conditions.

Besides pairing by word, we also tried pairing by student and averaging performance after each mode across the words the student practiced in that mode, but none of the differences were statistically reliable. This approach is more conservative because it controls for individual differences among students, and because each student's performance is independent of other students' performance. However, it is less powerful statistically because there were fewer students than words, and because it does not control for differences between the words practiced in different modes.

Table 2. Differences in word reading performance after assisted reading versus after Word Swap, paired by story and word and averaged over the students who practiced that story word in that condition

Outcome measure	**Outcome differences (Word Swap – assisted reading)**				**Paired t-test**
	Mean	Std. Dev.	95% Confidence Interval of the Difference		Sig. (2-tailed)
			Lower	Upper	
% accepted	0.000	0.175	-0.021	0.022	0.965
% asked help	0.015	0.171	-0.006	0.036	0.168
% credited	-0.007	0.203	-0.032	0.018	0.592
Latency	0.039	0.244	0.009	0.069	**0.011**
Reading time	0.060	0.347	0.017	0.103	**0.006**
Adjusted time	0.074	0.537	0.008	0.150	**0.028**

4 Learning Decomposition

Learning decomposition generalizes classic exponential learning curve analysis to estimate the relative benefit of different types of practice [10], and has now been used in several such analyses. In brief, it models each student's item performance data (in this case word reading times) as an exponential function of previous practice on the item. The model disaggregates practice into the number of encounters of each practice type (e.g., Word Swap or assisted reading), each weighted by a free parameter coefficient. Fitting the model to the data (e.g. in SPSS) yields parameter estimates that represent the relative value of each type of practice for that student. Averaging and bootstrapping the parameter estimates across students gives confidence intervals on the means and tells which differences between practice types are reliable.

We follow earlier work [10] in three respects. First, we measure performance using the adjusted time measure defined in Table 1 of Section 3, and exclude encounters where its value is undefined. Second, to exclude recency and story memorization effects as mentioned in Section 3, we measure performance only on a student's first encounter of a word each day, and only in a story that he or she has not read before. Third, we adopt the same general model form, including an additive term to represent the effect of word length. However, we use different practice types, namely assisted reading and Word Swap. Equation 1 shows the resulting model:

$$adjusted\ \ reading\ \ time = L * word\_length + A * e^{-b*(\#Reading + \beta * \#Swap)} \qquad (1)$$

Our four model parameters mean roughly this:

- L: the increase in predicted word reading time for each additional letter in the word
- A: the predicted time to read a word with no prior practice in either condition
- b: learning rate
- β: the impact of a Word Swap encounter compared to an assisted reading encounter, whose impact we define to be 1

The input variables #*Reading* and #*Swap* count the number of prior encounters of the same word in assisted reading and Word Swap, respectively. These practice variables include *all* encounters of the word, not just the first encounter on each day or in each story.

Using Equation 1, we build a model for each individual student. After excluding models for which the fitting procedure fails due to sparse data, we take medians of the remaining 140 models as the overall parameter estimates. We use medians instead of means in order to deemphasize outliers in the noisy individual estimates. We also derive the 95% confidence interval for each parameter using non-parametric bootstrapping [11]. Table 3 shows the result.

Table 3. Overall parameter estimates (± 95% confidence interval)

Parameter	L	A	b	β
Estimate	0.0615 ± 0.022	0.7035 ± 0.0775	-0.0515 ± 0.015	0.125 ± 0.1147

The confidence interval for β shows that it is significantly less than 1, which implies that Word Swap has significantly less impact than assisted reading in reducing (adjusted) word reading time. However, β is also reliably (though just barely) greater than 0, implying that Word Swap also reduces word reading time.

5 Knowledge Tracing

Knowledge tracing [12] infers a student's knowledge of a skill from observations of the student's performance on that skill. Knowledge tracing incrementally updates the probability K_n that the student knows a given skill at time step n, according to a dynamic Bayes net model with the following parameters:

- *knew*: Probability K_0 that the student already knew the skill prior to instruction
- *learn*: Probability of acquiring the skill from a single practice
- *forget*: Probability of losing a known skill
- *guess*: Probability of answering correctly without knowing the skill
- *slip*: Probability of answering incorrectly despite knowing the skill.

To investigate how different modes of practice influence student knowledge, we introduce another node *Practice Mode (PM)* into the basic knowledge tracing model, as Figure 1 shows.

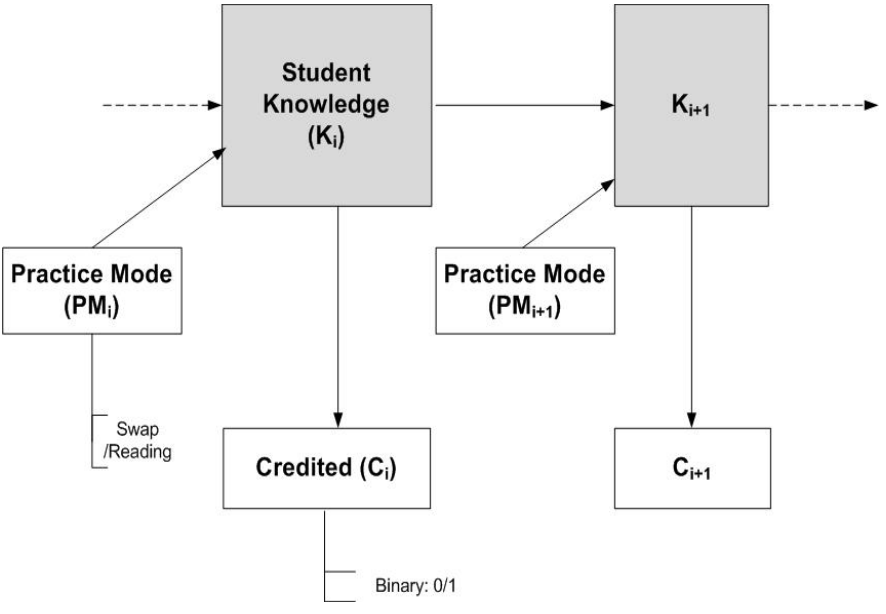


Fig. 1. Knowledge tracing model extended with a binary-valued "Practice Mode" node

This model assumes that the probability of a student's learning a skill depends on practice mode. To measure student performance in assisted word reading, we use *credited* (see section 3 for definition). For Word Swap steps, however, since we do not have observations of a student's reading the word, *credited* is unobservable. The extended model has more parameters: *is_reading* is the probability that the practice mode is assisted reading; K_0*_swap* and K_0*_reading* are the probability that the student already knew the word prior to any practice, conditioned on whether the first practice was Word Swap or assisted reading; *learn_swap* and *learn_reading* are the respective probabilities of acquiring the skill from a Word Swap or assisted reading practice opportunity; and finally, *forget_swap* and *forget_reading* are the respective probabilities of losing a skill after a Word Swap or assisted reading practice opportunity. The parameters *guess* and *slip* remain the same as in the basic knowledge tracing model.

One problem with fitting the data to a knowledge tracing model, however, is that the observed student performance can correspond to an infinite family of possible model parameter estimates [13]. Following [14], we address this problem by specifiying a plausible initial value for each parameter, and encoding domain knowledge as Dirichlet priors on the parameters to bias the model fitting procedure. We specify an order-2 Dirichlet distribution as two positive numbers α1 and α2, which correspond roughly to the number of positive and negative examples seen. For example, we use α1=9 and α2 = 6 for K_0*_swap*. These values mean roughly that the Dirichlet prior for K_0*_swap* is generated from 9 cases of the student already knowing a skill, and 6 cases of the student not knowing it, when the first practice is Word Swap. The expected value of K_0*_swap* is 9/(9+6) = 0.6.

We set the initial parameters, as well as α1 and α2, by examining histograms from previous knowledge tracing experiments, getting similar values to those in [14]. The

first three columns in Table 4 show the name and initial value of each parameter, as well as the α1 and α2 values of the Dirichlet priors. Notice that they avoid any bias toward either Word Swap or assisted reading. We refrain from specifying Dirichlet priors for the *learn* and *forget* parameters, so as not to prejudice the search through the model space.

Table 4. Initial values, Dirichlet priors, and aggregated estimates of the parameters in the knowledge tracing model

Parameter	Initial Value	Dirichlet (α1, α2)	Mean	Std. Dev.
is_reading	0.5	N/A	0.683	0.237
K_0*_swap*	0.66	(9,6)	0.599	0.019
K_0*_reading*	0.66	(9,6)	0.655	0.061
Guess	0.64	(17,9)	0.670	0.041
Slip	0.07	(1,15)	0.028	0.020
learn_swap	0.14	N/A	0.258	0.187
learn_reading	0.14	N/A	0.566	0.360
forget_swap	0.0014	N/A	0.014	0.086
forget_reading	0.0014	N/A	0.011	0.087

To investigate which practice mode helps more to learn a word, we treat the ability to read each word as a distinct skill. Then we build a model for each word using observations of many students' encounters of that word, using Bayes Net Toolkit for Student Models (BNT-SM) [15]. After excluding the cases where model construction fails due to sparse data (e.g. the word was encountered very few times, or in only one treatment condition), we get 259 word-specific models, across which we average the parameter estimates. The last two columns of Table 4 show the mean and standard deviation for each parameter.

A t-test, paired by word, shows no significant difference between *forget_swap* and *forget_reading*. In contrast, *learn_reading* is significantly larger than *learn_swap* ($p<0.01$). That is, students are likelier to acquire a word from assisted reading practice than from Word Swap practice.

6 Conclusion

This paper explored three methods to evaluate tutorial behaviors: RCT analysis, learning decomposition, and knowledge tracing. It reports a case study in the context of Project LISTEN's Reading Tutor, to test whether assisted reading and Word Swap practice differ in how well they help students learn words.

One result of this endeavor is to confirm that knowledge tracing can usefully be adapted to evaluate the impact of different tutor behaviors. Previous work [16, 17] used this approach to evaluate the same mode of practice with versus without tutor help. Here we evaluate two different modes of practice, each with a different task for the student, and consequently different types of performance to observe.

In comparing evaluation methods, we have two basic questions. First, did their results agree? Yes, all three methods indicate that assisted reading beat Word Swap on one or more of our measures. Though the three methods differ in input, output, and model form, the qualitative consistency of their results provides some empirical evidence for the validity of the results and the methods.

Second, were some methods more sensitive than others? If methods A and B agree, and A is more sensitive than B, we expect A to achieve statistical significance on more comparisons than B does. We see no such pattern. The methods agree qualitatively, but not on which measures show statistically significant differences between the two modes of practice. Clarifying the empirical behavior and relative utility of these methods will require comparing them on additional data sets from diverse domains.

The results imply that assisted reading is more effective than Word Swap at helping students learn to read words quickly, accurately, and independently. They do not necessarily imply that Word Swap is inferior for its intended purpose of teaching children to attend to the correspondence between print and speech. Indeed, conceivably children read words more slowly after Word Swap than after assisted reading because it actually succeeded. One challenging direction for future work is to develop an automated measure of such attention.

Acknowledgments. The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305B070458 to Carnegie Mellon University, by the National Science Foundation under ITR/IERI Grant No. REC-0326153, and by the Heinz Endowments. The opinions expressed are those of the authors and do not necessarily represent the views of the Institute, the U.S. Department of Education, the National Science Foundation, or the Heinz Endowments. We also thank the educators, students, and LISTENers who helped generate and analyze our data.

References

1. Mostow, J., Aist, G.: Evaluating tutors that listen: An overview of Project LISTEN. In: Forbus, K., Feltovich, P. (eds.) Smart Machines in Education, pp. 169–234. MIT/AAAI Press, Menlo Park (2001)
2. Beck, I.: Reading Today and Tomorrow (Teachers' Editions for Grades 1 and 2). Holt and Co., Austin (1989)
3. Beck, I., Hamilton, R.: Beginning reading module. American Federation of Teachers, Washington (2000) (Original work published 1996)
4. Cunningham, P.M., Cunningham, J.W.: Making Words: Enhancing the invented spelling-decoding connection. The Reading Teacher 46(2), 106–114 (1992)
5. McCandliss, B., Beck, I.L., Sandak, R., Perfetti, C.: Focusing attention on decoding for children with poor reading skills: Design and preliminary tests of the word building intervention. Scientific Studies of Reading 7(1), 75–104 (2003)
6. Razzaq, L., Heffernan, N.T., Lindeman, R.W.: What level of tutor feedback is best? In: Proceedings of the 13th Conference on Artificial Intelligence in Education, IOS Press (2007)

7. Mostow, J., Beck, J.: Some useful tactics to modify, map, and mine data from intelligent tutors. Natural Language Engineering (Special Issue on Educational Applications) 12(2), 195–208 (2006)
8. Zhang, X., Mostow, J., Beck, J.E.: All in the (word) family: Using learning decomposition to estimate transfer between skills in a Reading Tutor that listens. In: AIED 2007 Educational Data Mining Workshop, Marina del Rey (2007)
9. Mostow, J., Aist, G.: The sounds of silence: Towards automated evaluation of student learning in a Reading Tutor that listens. In: Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI 1997), pp. 355–361. American Association for Artificial Intelligence, Providence (1997)
10. Beck, J.: Using learning decomposition to analyze student fluency development. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053. Springer, Heidelberg (2006)
11. Cohen, P.R.: Empirical Methods for Artificial Intelligence, p. 405. MIT Press, Cambridge (1995)
12. Corbett, A., Anderson, J.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User modeling and user-adapted interaction 4, 253–278 (1995)
13. Beck, J.E., Chang, K.: Identifiability: A Fundamental Problem of Student Modeling. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511. Springer, Heidelberg (2007)
14. Beck, J.E.: Difficulties in inferring student knowledge from observations (and why you should care). In: Proceedings of the AIED 2007 Workshop on Educational Data Mining, Marina del Rey, pp. 21–30 (2007)
15. Chang, K.-m., Beck, J., Mostow, J., Corbett, A.: A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems. In: Proceedings of the 8th International Conference on Intelligent Tutoring Systems, Jhongli, Taiwan (2006)
16. Jonsson, A., Johns, J., Mehranian, H., Arroyo, I., Woolf, B., Barto, A., Fisher, D., Mahadevan, S.: Evaluating the Feasibility of Learning Student Models from Data. In: Educational Data Mining: Papers from the AAAI Workshop, pp. 1–6. AAAI Press, Pittsburgh (2005)
17. Chang, K.-m., Beck, J.E., Mostow, J., Corbett, A.: Does Help Help? A Bayes Net Approach to Modeling Tutor Interventions. In: AAAI 2006 Workshop on Educational Data Mining, Boston (2006)

Children's Interactions with Inspectable and Negotiated Learner Models

Alice Kerly and Susan Bull

Electronic, Electrical and Computer Engineering, University of Birmingham,
Edgbaston, Birmingham, B15 2TT, UK
{alk584,s.bull}@bham.ac.uk

Abstract. The Learner Model of an Intelligent Tutoring System (ITS) may be made visible (opened) to its users. An Open Learner Model (OLM) may also become a learning resource in its own right, independently of an ITS. OLMs offer potential for learner reflection and support to metacognitive skills such as self-assessment, in addition to improving learner model accuracy. This paper describes an evaluation of an inspectable and a negotiated OLM (one that can be jointly maintained through student-system discussion) in terms of facilitating self-assessment accuracy and modification of model contents. Both inspectable and negotiated models offered significant support to users in increasing the accuracy of self-assessments, and reducing the number and magnitude of discrepancies between system and user beliefs about the user's knowledge. Negotiation of the model demonstrated further significant improvements.

1 Introduction

Intelligent Tutoring Systems (ITS) routinely employ a learner model in order to provide tutoring and interaction tailored to the needs of the individual student. Conventionally this model has only been for the use of the system, and hidden from the learner. Open Learner Modelling argues that making the contents of the model visible for inspection by the student may bring opportunities for developing skills in reflection, metacognition and deep learning, e.g. [1], [2], [3], [4], [5]. Open Learner Models (OLM) may also allow the student and system to engage in a process of negotiation about the contents of the model, potentially enhancing learner reflection and model accuracy. Such negotiated learner models (e.g. [1], [2]) involve a collaborative construction and maintenance of the learner model. By requiring learners to discuss their beliefs about their knowledge with the system, argue against the system's assessment where they disagree or provide evidence for their own beliefs, it is suggested that learner reflection may be increased [1], [2]. This negotiation may also improve the accuracy of the learner model, leading in turn to improved adaptation by the ITS. OLMs may also be used as learning resources independent of an ITS, to prompt learners to reflect on their knowledge (or lack of it), to facilitate planning future learning, and to encourage users to take more responsibility for their learning [6]. Other researchers have argued that it is necessary for educational systems to model the student's meta-knowledge in addition to their domain knowledge [7]. It is this approach of modelling the student's own beliefs about their knowledge that is discussed in this paper.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 132–141, 2008.

Educational theorists have long emphasised the importance of learner reflection [8], [9], [10]. This is now being supported in the school classroom by Assessment *for* Learning, a UK education strategy that highlights the importance of supporting the development of metacognitive skills. Promoting pupil self-assessment is regarded as an essential component of this [11]. However, it is recognised that while the most effective learners are self-regulating [12] the effectiveness of this self-regulation is reliant on accurate self-assessment of what is known [13]. It has been shown, perhaps unsurprisingly, that not all (adult) students are good at evaluating their knowledge [14], and it was suggested that allowing the student to visualize the learner model may improve self-evaluation [15]. We propose to investigate this potential for learner model visualization in improving self-evaluation in younger (primary school) learners.

This paper describes an evaluation using two versions of CALMsystem – an Open Learner Model with an integrated ***C****onversational* ***A****gent* for ***L****earner* ***M****odelling* – independent of an ITS. The inspectable version of the system offers a learner the opportunity to inspect their learner model, to view the beliefs they and the system hold about their knowledge, and to make changes to their own beliefs about their knowledge as appropriate. The negotiated version adds a conversational agent to allow learners to discuss the learner model using a natural language interface and to negotiate changes. We consider these inspectable and negotiated versions of CALMsystem in terms of facilitating self-assessment accuracy and modification of model contents.

2 CALMsystem

CALMsystem opens the learner model to students, allowing them to see the representations of their current knowledge level as assessed by the system, and their self-assessment for each of the topics in the subject domain. The negotiated version also offers learners an opportunity to discuss and develop their learner model. Both inspectable and negotiated versions have potential to promote metacognitive skills and improve the model's accuracy.

The CALMsystem environment is browser based, operating independently of an ITS, and allows easy access to users from a variety of platforms. It allows users to view pages that show only their own confidence in their knowledge, only the system's assessments of their knowledge, or compare these in parallel. It also allows them to answer further questions on a topic of their choice, or one selected by the system, thereby allowing both user and system to initiate further interaction to update the learner model in the usual manner. Fig. 1 shows the browser interface (common to both versions of the system) and the conversational agent used to provide negotiation.

The system tracks the student's confidence and the system's assessment of the student's knowledge in each topic using two numerical scores. These two belief sets (learner's and system's) which form the learner model are stored independently, as is necessary for comparison and negotiation of the different beliefs (as in [1]). The user's confidence in each topic is maintained by the system as a continuous value between 0 and 1. For the purpose of display to the user, this value is converted into "low", "moderate", "good" or "high" levels, based on the ranges 0 - 0.25, 0.25 - 0.5, 0.5 - 0.75 and 0.75 - 1 respectively. These four levels offer an age-appropriate model for the 10-11 year old users in this study, who are familiar with self-evaluation scales of this granularity.

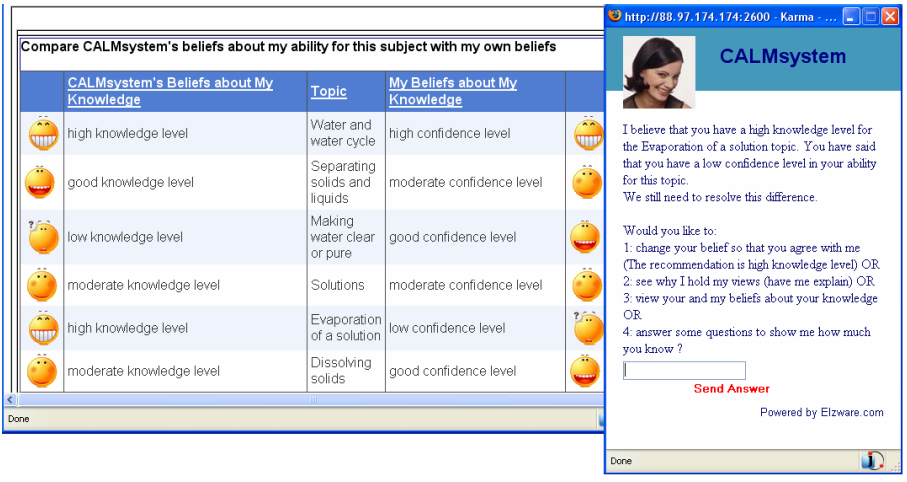


Fig. 1. System and learner assessments on six topics, and the conversational agent[1]

When a student first uses CALMsystem, they are required to assess their confidence in each topic by selecting the appropriate level ("low", "moderate", "good" or "high" confidence) and the initial numerical value is set as appropriate. Each time a student answers one of the multiple choice questions in CALMsystem (using the Answer Questions menu link) they are required to state the level that best matches their confidence in the topic. The system does not immediately change the numerical confidence value to match the user's new assessment, but uses an exponential filter[2] that weights most recent user assessments more strongly (so older results have a progressively lesser effect), allowing users to keep their model current.

The system's assessment of the student's knowledge is also maintained as a continuous value between 0 and 1, and uses an identical exponential filter, ensuring that the assessment represents the current knowledge level. This score for each topic is also recalculated every time the user answers a question (once past a threshold of 'sufficient evidence'). The score is increased each time a student answers a question correctly, and is reduced when a wrong answer is given. A student consistently answering questions correctly will attain a score approaching 1, and if most questions are answered incorrectly, the score will approach 0. For display, this knowledge value is also converted to four levels ("low", "moderate", "good" or "high") using the same

[1] Text reads "I believe that you have a high knowledge level for the Evaporation of a Solution topic. You have said that you have a low confidence level in your ability for this topic. We still need to resolve this difference. Would you like to: 1: change your belief so that you agree with me (The recommendation is high knowledge level) OR 2: see why I hold my views (have me explain) OR 3: view your and my beliefs about your knowledge OR 4: answer some questions to show me how much you know?"

[2] $y_t = (1-\alpha)\cdot y_{t-1} + \alpha \cdot x_t$ where y_t is the output of the filter (new score) at time moment t; y_{t-1} is the output of the filter after previous question (user's old score; t-1); x_t is the input of the filter (1 or 0 indicating correct or incorrect answer); $0 \leq \alpha \leq 1.0$ is the weighting parameter. The output y_t is the weighted sum of previous output and current input values. The smaller the parameter α, the longer the 'memory' of the filter and the greater the degree of smoothing.

numerical ranges as in user confidence. Both system and user beliefs are also illustrated with smiley faces (see Fig. 1) to allow easy comparison by the target users (aged 10-11) in this investigation.

2.1 Negotiation of the Learner Model

In the negotiated version of CALMsystem, inspection of the model is as described above, with negotiation of the learner model contents provided by a chatbot. It allows learners to use natural language to (i) query the model contents, (ii) ask for explanation or justification of the system's beliefs, (iii) offer justification of their own beliefs, (iv) change their beliefs as they refine their self-assessments, (v) modify their belief to match that of the system where they have been convinced by the model evidence, (vi) try to compromise with the system, or (vii) receive further test questions. These strategies were developed in [1] and explored in natural language in [16]. Discussion may be initiated by either the chatbot or the user.

The aim of this process of negotiation or discussion is to allow the student and system to refine their beliefs about the user's knowledge, and that this will lead to a reduction in areas of discrepancy between the learner and system beliefs. We suggest that a model refined in this way may be considered more accurate for use in driving an ITS's adaptation and may prompt further user reflection and self-assessment.

The chatbot does not attempt to be a tutor. The largest part of the chatbot's knowledge base relates to discussion of the learner model. Users may give inputs such as "why do we think differently?", "what am I good at?", "what's your belief?", "I have difficulty with [topic]", "change my belief", "what should I do next?", "why do you think I have a low level for [topic]?" or "let's talk about [topic]". They may request that the chatbot tell them the beliefs that are held, or just give the name of a topic to focus on. The learner can also ask the chatbot to give them test questions on a stated or unspecified topic. To handle many of these inputs the chatbot must connect to the database holding the learner model data, and construct its responses appropriately (see [16] for architectural details). For delivering tests it can select unanswered questions, ask the question, check the user's answer for correctness, and tell the user about the system's belief modifications made as a result. In this way it maintains focus on self-assessment and reflection on the contents of the OLM, an important benefit of independent OLMs [6]. However, the chatbot could be integrated into an ITS and provide tutoring as an extension of the system.

A short extract from a user interaction is shown below, showing the user modifying their belief after seeing the evidence from further test questions.

Chatbot: I believe that you have a low knowledge level for the Evaporation topic. You have said that you have a moderate confidence level in your ability for this topic. Would you like to: … [see footnote 1 for options offered]

User: 2

Chatbot: You've answered 4 questions and got 3 wrong for this topic. Therefore I believe you have a low knowledge level.

User: ask me quistions [sic] [OLMBot asks 2 questions; user answers correctly]

Chatbot: From your responses, I now believe you have a good knowledge level. Your own belief is that you have a moderate confidence level for the Evaporation topic. We will have to discuss this difference some more, or you might consider changing your belief for this topic.

User: i want to change my belif [sic]
Chatbot: What would you like to change your confidence level for Evaporation to? (Remember, this is *your* belief about your ability – not CALMsystem's assessment). You may change your beliefs to: 1: low confidence, 2: moderate confidence, 3: good confidence, 4: high confidence.
User: good
Chatbot: OK, I've updated your beliefs to show that you have good confidence in your abilities for the Evaporation topic.

The chatbot also has strategies and content to support 'smalltalk' (inputs not directly related to discussion of the learner model, but which have been found to be valuable in building rapport with users [17]) and to deal with unexpected user inputs. The intention is always to lead the user back into discussion of their learning as quickly as possible.

3 Experimental Evaluation

The aim of this study is to compare the effects of an inspectable and a negotiated learner model on self-assessment accuracy. As a proxy for self-assessment accuracy we compared the discrepancy between the system's assessment of the user's knowledge, and the user's assessment of their own capability. Inaccuracies in the system's modelling due to the user's accidental errors in answering questions are minimised by the use of the four broad knowledge levels, and the smoothing function of the exponential filter. It was hypothesised that using the inspectable version of CALMsystem would reduce this discrepancy, and that the discrepancy would be reduced further for participants who negotiated the learner model with the chatbot.

3.1 Measures of Self-assessment Accuracy

Three measures of the discrepancy between the student's confidence and system's assessments (and hence self assessment accuracy) were calculated for each user:

- **Numerical Measure of Discrepancy:** This measure sums the difference between the maintained numerical values for user confidence and system-assessed knowledge across all topics.
- **Number of Topics:** Where there is disagreement: This measure represents the number of topics that are not in agreement for a particular student. Topics are considered to be in agreement when the confidence and knowledge beliefs relate to the same level ("low", "moderate", "good" or "high").
- **Level Discrepancy:** This measure is a refinement of the Number of Topics measure outlined above, but takes into account the fact that a "low" to "high" discrepancy is more significant than, say, a "low" to "moderate" discrepancy. Adjacent levels (e.g. "moderate" and "good") are allocated a discrepancy distance of 1, those two levels apart (e.g. "low" and "good") a distance of 2 and those three levels apart (i.e. "low" and "high") are allocated 3. These distances are summed across all topics to give a measure of level discrepancy for each user (a theoretical maximum of 18). This discrepancy measure is considered to be of particular relevance, as it mirrors the typical view of a learner as to how far their own assessment differs from that of the system.

3.2 Participants, Materials and Methods

The study involved 25 UK Primary school children aged 10-11. CALMsystem was populated with questions on six science topics from their current study unit.

A between-subjects design was used, with the participants divided into two matched mixed-ability groups based on the results of a diagnostic test on the topics. One group was allocated to an *inspectable* learner model (LM) condition, and the other to a *negotiated* LM condition. All participants were shown how to use the system, its purpose and how it might be useful to them. Participants used the system for two sessions, three weeks apart, totalling 120 minutes. All users interacted with the system to make initial self-assessments, answer multiple choice questions, view their confidence ratings and the system's assessments, and modify their confidence records where they desired. Those in the negotiated LM condition also interacted with the chatbot to discuss their model.

As both users' confidence ratings and the system's assessments are recalculated after every question that is answered, the current values are always known and displayed by CALMsystem. The data used in this analysis was extracted from the learner model logs. The initial ('before-use') values are the beliefs held at the point where the system first had sufficient data about the user's knowledge of a topic to model the user. The final state of the learner model after both sessions gives the 'after-use' state.

3.3 Results

3.3.1 Improvement in Self-assessment Accuracy (Numerical Measure)

Before using CALMsystem, the mean self-assessment error for all 25 participants across all six topics was 1.74 (median 1.56, range 0.69-4.31). After final use of the system this mean error was reduced to 0.82 (median 0.66, range 0.29-2.43) for all users in the inspectable or negotiated conditions. The improvement by inspectable LM users (mean reduction 0.45, median 0.55, range -0.99-1.64) was significant ($t=1.83$, $p<0.05$). Negotiated LM users made highly significant ($t=4.72$, $p<0.0005$) improvements, (mean reduction in error 1.35, median 0.93, range 0.16-3.99). Notably, this improvement was significantly greater ($t=2.38$, $p<0.025$) than that for inspectable LM users (see Fig. 2).

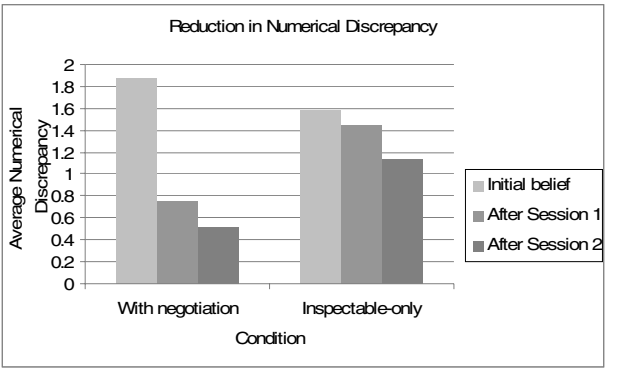


Fig. 2. Improvement in Self-Assessment (Reduction in Numerical Discrepancy)

3.3.2 Reduction in Number of Topics

The number of topics in which there was disagreement between the user and system as to the user's ability was counted. Before using the system, the mean number of topics with discrepancy was 3.88 (median 4, range 1-6). After final use of CALMsystem this average was reduced to 1.52 (median 1, range 0-6), an average reduction of 2.36. Inspection of the LM reduced the number of discrepancies significantly (mean reduction 1.5, median 2, range -3-5, ($t=1.95$, $p<0.05$)). The reducution was significantly greater ($t=2.08$, $p<0.025$) for participants in the negotiated LM condition (mean reduction 3.15, median 3, range 1-6, ($t=8.01$, $p<0.0005$)) than for those in the inspectable LM condition (see Fig. 3).

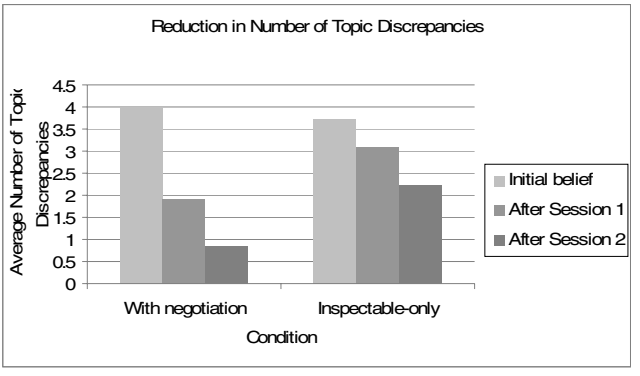


Fig. 3. Improvement in Self-Assessment (Reduction in Number of Topic Discrepancies)

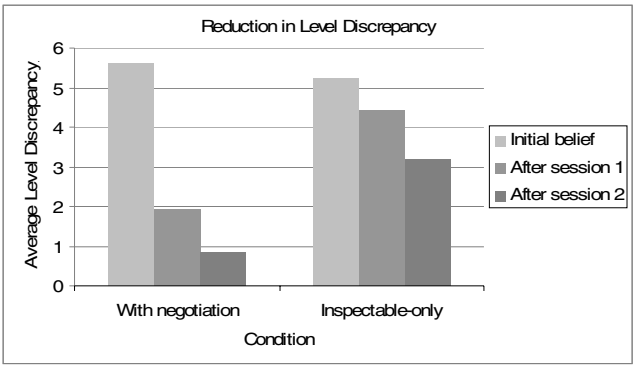


Fig. 4. Improvement in Self-Assessment (Reduction in Level Discrepancy)

3.3.3 Reduction in Level Discrepancy

The Level Discrepancy measure gives a value representing the disparity between levels ("low", "moderate", "good", "high") held by the student and system. Before using the system the mean level discrepancy was 5.44 (median 6, range 1-11). After final use of CALMsystem this average was reduced to 1.96 (median 1, range 0-9), an average

reduction of 3.48. Users in the inspectable LM condition reduced the Level Discrepancy significantly (mean reduction 2.08, median 3, range -4-7, ($t=1.84$, $p<0.05$)). Again it was found that the reduction in the Level Discrepancy was significantly greater ($t=2.31$, $p<0.025$) for participants in the negotiated LM condition (mean reduction 4.77, median 4, range 1-10, ($t=7.12$, $p<0.0005$)) (see Fig. 4).

3.3.4 Questions Answered

Users of the negotiated LM answered an average of 35.15 questions (median 35, range 22-61). Users in the inspectable condition answered an average of 51.08 questions (median 49, range 34-79), a highly significant difference ($t=3.19$, $p<0.005$).

4 Discussion

The results show that after using the CALMsystem open learner model all participants (in both conditions) significantly reduced the mean error in their self assessments. Users who engaged in negotiation with the chatbot demonstrated a significantly greater improvement in their self-assessment accuracy. These results suggest that inspection of the learner model can help prompt students to re-assess their knowledge, and that the chatbot negotiation element offers further benefit. Use of the system also reduced the number of discrepancies in learner/system beliefs. There was a substantial reduction in discrepancies for all participants; again this was significantly greater for negotiation users than for inspectable LM users. This reduction in the number of topics where user and system disagree results in a model where both parties hold more similar beliefs, allowing users to help direct potential ITS adaptations which they may consider of more value. The improvement in self-assessment accuracy should allow users to better target future learning and develop greater learner autonomy.

Interestingly, the discrepancy measures reduce rapidly across the trial for negotiated LM users, but markedly less so for inspectable users. This suggests that exposure to the OLM alone was lesser of an incentive for children to substantially change their self-assessments of confidence in a topic. The more proactive chatbot element, which persuades the users to challenge their belief where there are discrepancies appears to be more effective in making them consider their ability and make changes to their self-assessments. As shown in Figures 2, 3 and 4, the interaction continued to reduce discrepancies after a second session, suggesting that there was some lasting effect over the period between sessions (three weeks). Further study would be required to ascertain whether the extended use of a negotiated learner model would improve general self-assessment and metacognitive skills, and whether the improvements in self-assessment would be maintained over time.

Users in the inspectable LM condition answered far more questions in the interaction; this was the main activity available to them. This will have given them greater opportunity to view the representations of the beliefs held. However, despite this opportunity to consider the different beliefs more often, these users' beliefs did not change as significantly as those of the users with chatbot negotiation. Answering questions, re-stating confidence, and seeing the resultant model alone appears beneficial, but a lesser prompt to reflect on the learner model than offered by negotiation.

Users of both the inspectable and negotiated systems demonstrated significant improvements in self-assessment accuracy and in reducing the number and magnitude of discrepancies. The further improvements demonstrated by the negotiated LM suggest that where negotiation can be included this would provide additional benefits. The chatbot may persuade or help users to engage with their learning by exposing them to a proactive tool that they are willing to work with. This may be an effect of the novelty, naturalness or accessibility of a chatbot, or may be due to the content it offers.

Further work is necessary to explore whether the improvements in self-assessment transfer back to normal classroom scenarios (i.e. without computer), and whether belief changes persist beyond use of the system. It will also be interesting to explore if it is the chatbot's dialogue content that is effective, or whether the presence of the chatbot is a motivational factor which keeps young users engaged with the process.

5 Summary

We have presented an evaluation of two versions of an Open Learner Model. One version offers inspection of the learner model, while the other is supported by a chatbot to provide discussion and negotiation of the learner model contents. This negotiation allows the user and system to collaboratively construct and maintain the learner model, providing further opportunities for the learner to reflect on their knowledge and to refine their self-assessments than was seen in users of the inspectable-only model. Improvements were seen in both conditions. The study showed that users who engaged in negotiation reduced inaccuracies in their self-assessments significantly more than those users who used the system without negotiation support.

Acknowledgements

The first author is funded by the UK Engineering and Physical Sciences Research Council. We acknowledge Elzware Ltd's support and provision of software licences.

References

1. Bull, S., Pain, H.: Did I Say What I Think I Said, and Do You Agree With Me?: Inspecting and Questioning the Student Model. In: Greer, J. (ed.) World Conference on Artificial Intelligence in Education. AACE, Charlottesville, pp. 501–508 (1995)
2. Dimitrova, V.: STyLE-OLM: Interactive Open Learner Modelling. International Journal of Artificial Intelligence in Education 13, 35–78 (2003)
3. Kay, J.: Learner Know Thyself: Student Models to give Learner Control and Responsibility. In: Halim, Z., Ottomann, T., Razak, Z. (eds.) ICCE 1997 International Conference on Computers in Education, Kuching, Malaysia, pp. 17–24 (1997)
4. Morales, R.: Exploring Participative Learner Modelling and Its Effects on Learner Behaviour. PhD Thesis. University of Edinburgh, Edinburgh (2000)
5. Zapata-Rivera, J.D., Greer, J.: Externalising Learner Modelling Representations. In: Workshop on External Representations of AIED: Multiple Forms and Multiple Roles. International Conference on Artificial Intelligence in Education, San Antonio, Texas (2001)

6. Bull, S., Mabbott, A., Gardner, P., Jackson, T., Lancaster, M., Quigley, S., Childs, P.A.: Supporting Interaction Preferences and Recognition of Misconceptions with Independent Open Learner Models. In: Adaptive Hypermedia 2008. Springer, Berlin (2008)
7. Aleven, V., Koedinger, K.: Limitations of Student Control: Do Students Know When They Need Help?: Intelligent Tutoring Systems, pp. 292–303. Springer, Berlin (2000)
8. Dewey, J.: How We Think: A Restatement of the Relation of Reflective Thinking to the Educative Process. D. C. Heath and Company, Boston (1933)
9. Schön, D.: The Reflective Practitioner: How Professionals Think in Action. Arena. Ashgate Publishing Limited, Aldershot (1991)
10. Kolb, D.: The Process of Experiential Learning. In: Kolb, D. (ed.) The Experiential Learning: Experience as the Source of Learning and Development, Prentice-Hall, NJ (1984)
11. Black, P., Wiliam, D.: Inside the Black Box: Raising Standards Through Classroom Assessment. King's College London, School of Education, London (1998)
12. Butler, D.L., Winne, P.H.: Feedback and Self-Regulated Learning: A Theoretical Synthesis. Review of Educational Research 65, 245–281 (1995)
13. Schoenfeld, A.H.: What's All the Fuss About Metacognition? In: Schoenfeld, A.H. (ed.) Cognitive Science and Mathematics Education, pp. 189–215. Lawrence Erlbaum, Hillsdale (1987)
14. Mitrovic, A.: Self-assessment: how good are students at it? In: AIED 2001 Workshop on Assessment Methods in Web-Based Learning Environments & Adaptive Hypermedia, San Antonio, pp. 2–8 (2001)
15. Mitrovic, A., Martin, B.: Evaluating the Effect of Open Student Models on Self-Assessment. International Journal of Artificial Intelligence in Education 17, 121–144 (2007)
16. Kerly, A., Ellis, R., Bull, S.: CALMsystem: A Conversational Agent for Learner Modelling. Knowledge Based Systems 21, 238–246 (2008)
17. Kerly, A., Hall, P., Bull, S.: Bringing Chatbots into Education: Towards Natural Language Negotiation of Open Learner Models. Knowledge Based Systems 20, 177–185 (2006)

Using Similarity Metrics for Matching Lifelong Learners

Nicolas Van Labeke, Alexandra Poulovassilis, and George Magoulas

London Knowledge Lab,
Birkbeck, University of London,
23-29 Emerald Street
London WC1N 3QS - United Kingdom
{nicolas,ap,gmagoulas}@dcs.bbk.ac.uk
http://www.lkl.ac.uk/research/myplan/

Abstract. The L4All system provides an environment for the lifelong learner to access information about courses, personal development plans, recommendation of learning pathways, personalised support for planning of learning, and reflecting on learning. Designed as a web-based application, it offers lifelong learners the possibility to define and share their own timeline (a chronological record of their relevant life episodes) in order to foster collaborative elaboration of future goals and aspirations. A keystone for delivering such functionalities is the possibility for learner to search for 'people like me'. Addressing the fact that such a definition of 'people like me' is ambiguous and subjective, this paper explores the use of similarity metrics as a flexible mechanism for comparing and ranking lifelong learners' timelines.

1 Introduction

Supporting the demands of lifelong learners is increasingly considered at the core of the learning and teaching strategy of HE and FE institutions and poses new challenges, such as enabling better support for lifelong learners and facilities for accessing cross-institutional resources. To address these challenges, it is important to exploit further the advantages of Information and Communications Technology networks to enable better support for planning lifelong learning and ubiquitous access to lifelong learning facilities from home, the workplace and educational organisations. This new trend to educational services has led to research and development that involves the provision of new learner-centred models of organising and delivering educational resources (see for example the integrated framework proposed in [1,2]).

The L4All system [3,4] provides an environment for the lifelong learner to access information about courses, personal development plans, recommendation of learning pathways, personalised support for planning of learning, and reflecting on learning. The MyPlan project follows on from the initial L4All pilot project and aims to develop, deploy and evaluate personalised functionalities for the creation, searching and recommendation of learning pathways. This will enhance individual learners' engagement with the lifelong learning process by offering

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 142–151, 2008.

personalised levels of learner control over their learning pathways, personalised support in the reflection of where their learning activities may take them, and management of their personal record of progress and attainment. It will also support building communities of learners with similar interests, and information sharing with other members of the community, other users of the L4All system, and HE/FE institutions. Figure 1 shows the main page of the L4All system.

At the core of L4All is the specification of a User Model that addresses the specificities of lifelong learners and is based on the notion of learning 'trails' [5]. In the context of L4All, a 'trail' is a *timeline*-based representation of learners' work, learning and other life experiences that provides a holistic approach and continuity between their learning episodes and work experiences.

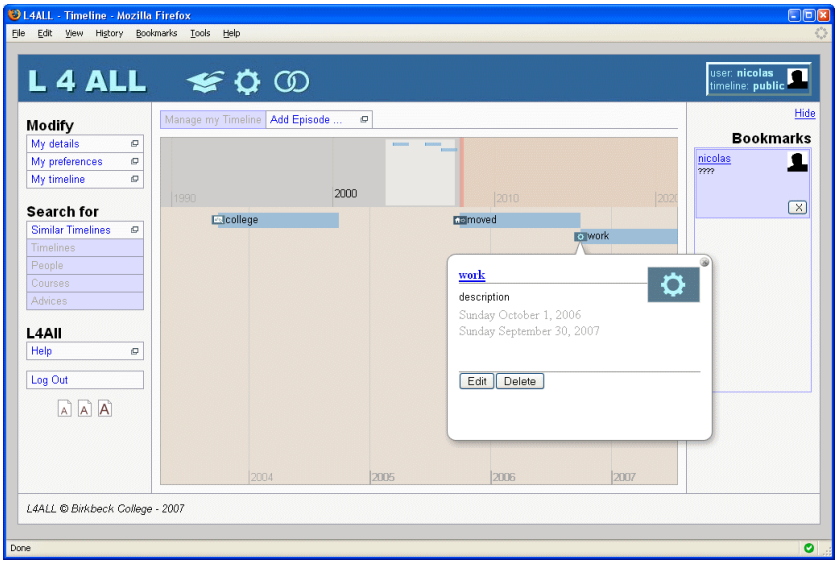


Fig. 1. The main page of L4All with the user's timeline (*centre*), access to the various functionalities (*left*) and a bookmarks section for networking (*right*)

One requirement for offering such personalised services is to provide learners with the possibility to search for 'people like me', i.e. to exploit the full structure and content of their profiles and timelines in order to find similar matches that will foster collaborative elaboration of future goals and aspirations. This poses some interesting challenges that this paper addresses. First, the structure of the timeline, as a sequence of temporal records, is potentially of such complexity that it does not immediately suggest a natural way of enabling comparisons between timelines. Second the notion of similarity of timelines is vague and subjective, and it is not clear which aspects of that complex structure should be considered and how they should be compared. Third, assuming that a personalised search and similarity-ranking of timelines can be designed and developed, supporting learners in exploiting such functionalities is an open problem.

This paper presents an investigation of these challenges and is organised as follows. First, we review the User Model underlying the L4All system, in particular the way timelines are represented. Second, a flexible mechanism for encoding the timeline in a form suitable for similarity matching is presented. Third, several algorithms for similarity measures of timelines are compared and analysed from the point of view of their behaviour in identifying key aspects of timeline comparisons. Fourth, we describe the user interface for the personalised construction of a new 'people like me' search, and for the visualisation and exploration of the timelines returned. The paper concludes by addressing some of the issues arising from our work and proposes some future developments.

2 User Model and Timelines

The L4All User Model [6] is comprised of three parts:

1. The *User Profile* contains personal information about learners such as their name, gender, year of birth, email, login name and password.
2. The *Learning Profile* contains information about the educational and professional background of learners (such as current occupation, highest qualification and skills) and information about their learning needs (such as willingness to travel, current learning goal, preferred mode of learning – part-time, full-time – and preferred learning methods – in groups, alone, online).
3. The *Timeline* is the novel part of the User Model, specifically addressing the particularities of lifelong learners. It represents the learning – and, more generally, life – pathway of the learners to date and contains a chronological record of those episodes of their life that they deem significant to their personal development.

Episodes in a timeline are identified by their category, selected from 20+ categories currently supported by the system. They include personal episodes, e.g. relocation, travel abroad, illness, marriage, death in family, etc., occupational episodes, e.g. started work, set up business, retired, did voluntary work, etc. and educational episodes, e.g. attended college, university or school, attended courses, etc. Each episode is specified by a start date and a duration (if applicable), title, description, keywords and an optional URL.

In order to extend the descriptive power of the timeline, some of the most significant episodes are also further elaborated by one or two further attributes, referred to as primary and secondary classifications: educational episodes by a subject and a qualification level; work episodes by an industry sector and a position; and business episodes by an industry sector. These additional attributes are populated by a specific tree-like taxonomy of values selected form relevant British standards[1]. The structure and identifiers of these taxonomies have been

[1] The Standard Industrial Classification (SIC), the Standard Occupational Classification (SOC), the National Qualification Framework (NQF) and the Labour Force Survey's Subject of Degree (SBJ). See the Labour Force Survey User Guide http://www.statistics.gov.uk/downloads/theme_labour/Vol5.pdf.

maintained but, for usability purposes, their depth is limited to four levels. Elements in each taxonomy can therefore be represented by four-digit identifier, each digit uniquely identifying its precise position in the tree.

3 Similarity Measures of Timelines

The initial prototype of the L4All system supported several search functionalities over users and their timelines. Two limitations of this approach were identified during the first piloting phase [4]. First, all the search functionalities were keyword-based, targeting the various fields of the User Profile, Learning Profile and Timelines, and therefore limited in their scope. In particular, searching on timelines returns matches based solely on the occurrence of the keywords present in one or several episodes but cannot exploit the overall structure of the timeline. Second, the results of any search were not personalised according to the particular user performing the search. An alternative approach was needed, that could take into account both these issues: in other words, some form of comparison or similarity measure between a user's timeline and the rest of the timelines in the L4All repository.

String metrics offer such a possibility. String metrics (also referred to as similarity metrics) have been widely used in information integration and in several fields of applied computer science [7,8]. In the context of Intelligent Tutoring Systems, similarity metrics have been used in the REDEEM system [9] to compare alternative sequences of instructional activities as produced by authors. In the context for the L4All timelines, the main requirement for using similarity metrics is to encode a time-based sequence of records into a token-based string. For this purpose, we have made four simplifying assumptions at the outset (the implications of these assumptions for users will be explored in our forthcoming evaluation activities):

The precise duration and dates of an episode have no particular significance. This may seem strange for a time-dependent data structure but the relevance and usage of such information for searching for 'people like me' is ambiguous. Should we consider two learners having done the same university degree but at different dates similar or not? Should we consider them more different if one of them has taken twice as long as the other (being part-time for example)? Or is it enough, at some level, to consider them similar since both of them have done this particular degree? In the absence of evidence supporting one point of view against the other, we decided, initially, to ignore this information. Only each episode's relative time-stamp (i.e. its position in time compared to the other episodes in the timeline) is used in order to 'linearise' the timeline by ordering the episodes in chronological order.

Gaps between episodes have no particular significance unless explicitly expressed as an episode. The problem posed by gaps in timelines is the lack of explicit explanation for their occurrence and therefore for their significance for the timeline. Again, in the absence of such information, they are ignored.

Some categories of episode may have no role to play in defining 'people like me'. The purpose of a timeline is for learners to record every episode of their background that may have an impact on their learning pathways. For example, personal episodes such as marriage, illness, relocation, etc. are important as they may have a clear influence on the decisions made for personal development (e.g. a course at a particular learning institution may have been followed because of a relocation to a particular city). However, this does not necessarily mean that such episodes are a prerequisite or a necessary condition for reaching a particular stage in someone else's development. Their importance while searching for role models, inspiration, or 'people like me' are therefore ambiguous and subjective. Therefore, whether to include or not particular categories of episode in the similarity matching should be left to the user to specify.

The exact classification of an episode may not be significant in defining 'people like me'. As described earlier, some of the most important episodes in the timeline (educational and work-related episodes) use specific attributes to precisely describe their instance, e.g. working as a researcher in computer science. However, taking such a fine-grained description of episode may not be useful in searching for 'people like me', as it may make more sense to consider that a researcher (without a precise field) is someone to consider 'like me'. Therefore the level of specialisation of episodes should also be left to the user to specify.

Using these assumptions, it is now relatively straightforward to generate a token-based string representing the timeline. Each episode of the timeline is encoded as a string token composed of a two-letter unique identifier of the category of the episode (e.g. `Cl` for a College episode, `Wk` for a Work episode) and two four-digit codes classifying the exact instance of this episode (as described in the previous section). Note that, in order to maintain a consistent pattern for the token's encoding, nonexistent or unspecified classifications are encoded as 0.0.0.0.

Combining the two first assumptions above means that no time information is used to encode episodes, only their relative position matters[2]. Filters are then applied to the string of tokens to remove the episodes that should not be considered in the current similarity search, as well as for limiting the depth of their classification. In the latter case, the use of the coding system for the classification facilitates the process: digits below the specified depth are replaced by 0, replacing the specific classification by a more general parent.

4 Comparison of Similarity Measures

The metrics used in this study are part of the `SimMetrics`[3] JAVA package, an open source extensible library of metrics that provides real number-based similarity measures between strings, allowing both normalised and un-normalised output. The `SimMetrics` package contains about 20 different metrics, some of them customisable by using user-defined cost functions and tokenisers. Not all

[2] With an arbitrary decision as to their ordering if multiple episodes coincide in time.

[3] `SimMetrics`, see `http://www.dcs.shef.ac.uk/~sam/stringmetrics.html`.

Table 1. List of encoded timelines used for the metrics comparison

Ref.	Description	Encoding
Source	Timeline used as the source for the similarity measure	Cl00 Un00 Mv00 Wk00
ID	Timeline identical to *Source*.	Cl00 Un00 Mv00 Wk00
RE	Timeline containing the same episodes as *Source* but in a totally different order.	Un00 Wk00 Cl00 Mv00
AD_w	New episode (similar to an existing one) added to *Source*.	Cl00 Un00 Mv00 Wk00 Wk00
AD_e	New episode (different from all existing) added to *Source*.	Cl00 Un00 Mv00 Wk00 Bs00
RM_w	Last episode removed from *Source*.	Cl00 Un00 Mv00
RM_u	One episode removed from *Source*.	Cl00 Mv00 Wk00
SB_e	One episode of *Source* substituted by a new one (different from all existing ones).	Cl00 Un00 Mv00 Bs00
SB_u	One episode of *Source* substituted by an existing episode.	Cl00 Un00 Mv00 Un00
SB_w	One episode of *Source* substituted by a variant of an existing episode (a different classification).	Cl00 Un00 Mv00 Wk10

metrics can be used in our context, since some are tailored for working on a particular application domain (linguistic for example) and require strings that are incompatible with our encoding of timelines. We refer the reader to the package documentation for descriptions of each metric.

Table 1 shows a set of synthetic timelines used in our comparison study. They are deliberately simplistic in their structure, as the purpose of this comparison is to identify general trends arising from the various similarity metrics, rather than evaluating their intrinsic power of discrimination.

The *Source* timeline is a string of four episodes of different type: college (`Cl00`), university (`Un00`), move (`Mv00`) and work (`Wk00`). Each episode has been encoded as a token, using the scheme described in the previous section. For the sake of clarity, and since this comparison does not rely on the full power of discrimination of the scheme, the episode classifications have been reduced to a single digit each (i.e. representing 0.0.0.0 as 0).

The target timelines represent a variety of alterations of the *Source* timeline that could occur in real-life situations: a totally similar timeline (i.e. the same sequence of episodes), a reordered timeline (i.e. the same episodes but totally reordered), adding an extra episode, removing an existing episode, substituting an episode by another one. Note that the set of target timelines listed in the table only represent the most representative of each group. In order to test the behaviour and consistency of the metrics, all possible combinations were generated for each group (e.g. timelines representing the addition of a new episode were generated considering every possible position in the *Source* timeline).

Table 2 summarises the results of the different similarity measures applied to every target timeline. The values shown in the table do not represent the distance between the two strings but their normalised similarity, i.e. the ratio between the calculated distance and the maximum distance. As mentioned earlier, the main aim of this comparison is not to focus on individual measures for assessing

Table 2. Normalised similarity between the source and the target timelines

	ID	RE	AD_w	AD_e	RM_w	RM_u	SB_e	SB_u	SB_w
Levenshtein	1	0	0.8	0.8	0.75	0.75	0.75	0.75	0.75
Needleman - Wunsch	1	0	0.8	0.8	0.75	0.75	0.75	0.75	0.88
Jaro	1	0.72	0.93	0.93	0.92	0.92	0.83	0.83	0.83
Matching Coefficient	1	1	0.8	0.8	0.75	0.75	0.75	0.75	0.75
Euclidean Distance	1	1	0.84	0.84	0.8	0.8	0.75	0.75	0.75
Block Distance	1	1	0.89	0.89	0.86	0.86	0.75	0.75	0.75
Jaccard Similarity	1	1	1	0.8	0.75	0.75	0.6	0.75	0.6
Cosine Similarity	1	1	1	0.89	0.87	0.87	0.75	0.87	0.75
Dice Similarity	1	1	1	0.89	0.86	0.86	0.75	0.86	0.75
Overlap Coefficient	1	1	1	1	1	1	0.75	1	0.75

their accuracy but to extract general conclusions regarding their behaviour when confronted with particular configurations. From these results, several conclusion can be drawn. First, all the similarity measures are indeed able to recognise complete similarity between timelines (as indicated by all 1 in the ID column). More interestingly, three groups of metrics emerge, as listed in Table 2.

The first group includes transformation-based metrics like Levenshtein, Jaro and Needleman-Wunsch that are able to discriminate between the basic operations of string manipulation (copy, substitution, addition, deletion). The non-zero result for the Jaro distance in the RE column can be explained by a threshold used for determining matching tokens (see the documentation of this metric); our test strings are not long enough (only four tokens) to allow proper discrimination. All these metrics do not take into consideration the position of the token involved in one of the string manipulations (whatever the location of the added or substituted episode, the scores are the same). The only exception is the Needleman - Wunsch distance, which gives a different score when a variant of the initial episode (i.e. same category but different classification) is substituted (score of 0.88 in SB_w, instead of 0.75 in SB_e and SB_u). This is due to the use of specific gap cost and distance functions that can be tailored to the particular nature of the data involved in the similarity measure and therefore could be adjusted for our particular use of the timelines (see Section 6).

The second group of metrics includes vector-based metrics such as Block Distance, Euclidean Distance and Matching Coefficient that are not able to discriminate between re-ordered strings, as indicated by 1 in the RE column. Whatever the order of the tokens in the string, both source and target are considered to be identical since they contain the same set of tokens. As with the metrics in the previous group, the results for addition, substitution and removal of tokens are position-independent.

The third group of metrics includes the rest of the vector-based metrics (Jaccard, Cosine, Dice Similarities and Overlap Coefficient) which, as with the previous group, do not discriminate between reordering of tokens. Moreover, this

group also fails to take into account the duplication of tokens in the string, as exemplified by the scores of 1 in the AD_w column (i.e. adding an episode that is already existing in the timeline) or the different scores for the SB_u column (i.e. substituting an episode with one that is already existing, resulting in fact in the deletion of this episode). Once again, this is because of the set-based algorithms used for these metrics, in particular the use of intersection/union procedures rather than summation as in the previous group. This is also reflected by the fact that substitution also depends on the nature of the episode substituted (the SB_u column give scores different from the other substitutions). In this group, the Overlap Coefficient is an extreme case, as it basically measures whether the source string is a subset of the target one (or the converse).

5 Searching for 'People Like Me'

What the comparison above shows is that different similarity metrics offer different degrees of support for the basic operations of string manipulation: copy, substitution, addition or deletion of a token. The important point here is that the comparison does not highlight one particular metric as being more useful or accurate for our purpose, precisely because our purpose (or, rather, the user's) is unknown. The assumptions made in Section 3 encompass a wide range of users' behaviour regarding the way they understand a 'people like me' functionality.

In order to validate these assumptions, a dedicated interface for such searches was therefore designed and implemented. It provides users with a three-step process for specifying their own definition of 'people like me'. The first step of a user's query specifies those attributes of the user's profile that should be matched with other users' profiles (age, qualification, location, etc.) and act as a filtering of the possible candidates before application of the similarity comparison on the timelines. The second step of the query specifies which part(s) of the timeline should be taken into account for the similarity comparison (currently by selecting the appropriate categories of episode). The final step specifies the nature of the similarity measure to be used (i.e depth of episode classification and metric). Once a definition of 'people like me' has been specified by the user, the search returns a list of all candidate timelines, ranked by relevance (i.e. their normalised similarity measure). The user now have the possibility to access any returned timelines and explore them.

This first approach to offering a 'people like me' functionality has given us the possibility to accumulate information about usage and expectations from users. It has offered us some insight into the context and relevance of particular configurations and how specific aims – such as looking for aspirational timelines or learning recommendations – could be supported. These issues and proposals for personalised support for the variety of activities they highlight will be investigated further in future work.

6 Discussion and Future Work

Lifelong learning requires technology to be used effectively to support learners in becoming more aware of their own learning and help them with planning of their learning throughout life under varying circumstance and settings. In this context, it is important to support user engagement and participation in lifelong learning and facilitate collaboration among lifelong learners for community building. In this paper we have discussed how string similarity measures could be used to encode and compare the timelines of lifelong learners. We have shown that existing metrics behave differently in identifying key aspects of timeline comparison, such as addition, substitution or deletion of episodes. Since the precise definition of what is a similar timeline is ambiguous and subjective, we have designed a new user interface for L4All such that learners can specify their own definition of 'people like me', offering them the possibility to decide which aspects of a timeline need to be considered or not for the matching. Evaluating the soundness and acceptability of this approach for users – as well as the usability of our user interface – are currently under evaluation. In the first evaluation phase – underway at the time of writing – we will be asking learner participants from three different learning institutions to explore the definition and the results of applying different similarity definitions on a predefined database of synthetic timelines. In the long term, several issues arising from our work will also be explored.

The encoding of timelines and episodes for similarity computation may need to be improved, in particular in determining by how much two episodes are similar. One way of dealing with this issue is by using the depth-adjusting encoding of episodes, where specific classification identifiers can be relaxed to one of their more general parents in the hierarchy, thus increasing the chance for two episodes to be compared as identical. But by doing so we are not only losing the descriptive power of episodes but also uniformly applying the filtering on all episodes in the timeline. An alternative, unfortunately only supported by distance metrics such as Levenshtein or Needleman-Wunsch, is to incorporate user-defined distance and gap cost functions, i.e. specifying a fine-grained analysis of the distance between two given tokens and of the cost of adding or removing a token in a string. Instead of the current binary comparison of episodes (i.e. their encodings are syntactically equal or not), we could adjust the distance between two similar episodes by the distance between their classifications (i.e. the sum of the distances to the closest common ancestor of each classification's element).

Similarly, our first two assumptions in Section 3 are clearly the most critical. The ongoing piloting of our techniques will certainly provide us with insights about the importance or not of taking temporal information into account. Extensions of our token-based encoding of timelines or even a specific similarity mechanism that maintains temporal tags will have to be considered.

Finally, a further important issue we still need to address is the question of providing lifelong learners with support for exploiting the results of a similarity search. Currently, we are relying on a pure visualisation approach, by displaying both the learner's own timeline and similar timelines returned by the search.

A specifically designed dynamic widget is used to allow the learner to scroll back and forward across each timeline, to access individual episodes, etc. Such an interactive visualisation of timelines is certainly helping learners to explore alternative timelines and is supporting them in elaborating future goals and aspirations, but more proactive supports will also be investigated. To enable the provision of feedback and on-demand support necessitates the ability to identify the reasons for a search considering two timelines as being similar. Again, metrics such as Needleman-Wunsch offer the possibility for such an identification by enabling backtracking of the distance computation and determining potential sequence alignments, i.e. the ability to identify alignment between pairs of tokens in matching strings.

Acknowledgments. The MyPlan – Personal Planning for Learning Through Life project (http://www.lkl.ac.uk/research/myplan/) is funded by the e-Learning Capital Programme of the Joint Information System Committee, UK.

References

1. Koper, R., Tattersall, C.: New directions for lifelong learning using network technologies. British Journal of Educational Technology 35(6), 689–700 (2004)
2. Koper, R., Giesbers, B., van Rosmalen, P., Sloep, P., van Bruggen, J., Tattersall, C., Vogten, H., Brouns, F.: A design model for lifelong learning networks. Interactive Learning Environments 13(1–2), 71–92 (2005)
3. de Freitas, S., Magoulas, G., Oliver, M., Papamarkos, G., Harrison, A.P.I., Mee, A.: L4all - a web-service based system for lifelong learners. In: Proceedings of eChallenges 2006 (Workshop on Next Generation in Technology Enhanced Learning), pp. 1477–1484 (2006)
4. de Freitas, S., Harrison, I., Magoulas, G., Mee, A., Mohamad, F., Oliver, M., Papamarkos, G., Poulovassilis, A.: The development of a system for supporting the lifelong learner. British Journal of Educational Technology 37(6), 867–880 (2006)
5. Peterson, D., Levene, M.: Trail records and navigational learning. London Review of Education 1(3), 207–216 (2003)
6. Baajour, H., Magoulas, G., Poulovassilis, A.: Modelling the lifelong learner in a services-based environment. In: Proceedings of ITA 2007- 2nd International Conference on Internet Technologies and Applications, pp. 181–190 (2007)
7. Gusfield, D.: Algorithms on Strings, Trees, and Sequences - Computer Science and Computational Biology. Cambridge University Press (1997)
8. Cohen, W.W., Ravikumar, P., Fienberg, S.E.: A comparison of string distance metrics for name-matching tasks. In: Proceedings of IIWeb 2003 – IJCAI Workshop on Information Integration on the Web, pp. 73–78 (2003)
9. Ainsworth, S., Clarke, D., Gaizauskas, R.J.: Using edit distance algorithms to compare alternative approaches to its authoring. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 873–882. Springer, Heidelberg (2002)

Developing a Computer-Supported Tutoring Interaction Component with Interaction Data Reuse

Chi-Jen Lin[1], Chih-Yueh Chou[2], and Tak-Wai Chan[1]

[1] Graduate Institute of Network Learning Technology, National Central University, Taiwan
[2] Department of Computer Science & Engineering, Yuan Ze University, Taiwan
dan@cl.ncu.edu.tw, cychou@saturn.yzu.edu.tw, chan@cl.ncu.edu.tw

Abstract. Most of intelligent tutoring systems (ITSs) were used to be developed by researchers or professionals with knowledge engineering background, due to knowledge model development. As a result, most of teachers, who are the best candidates of ITS developers in terms of their knowledge and motivation, are hindered from developing relevant systems. This study aims to explore a general approach for teachers to developing a computer-supported tutoring interaction component. Interaction data reuse (IDR) is proposed as the key concept for the design of the development approach.

Keywords: Interaction data reuse, knowledge model development, ITS development problem.

1 Introduction

ITS development is known for long to be both *difficult* and *labor-intensive* [1, 2, 3]. The causes of the ITS development problem are cognitive task analysis and artificial intelligence programming [2, 3], both of which are pertinent to *knowledge model development*. Generally, teachers are incapable of developing knowledge models, but in terms of their job nature and the potential of ITS in sharing their teaching load, teachers are best candidates for ITS development. Therefore, the workload on knowledge model development required by a particular ITS development approach is an indicator of the suitability of that approach for teacher use. Although there exist other factors than ITS development approaches that may influence the ITS development workload on knowledge model development, factors such as development decisions, tool support, and subject domains, the spectrum of ITS development approaches illustrated in Fig. 1 provides us an approximate overall understanding of some ITS development approaches.

Early ITS development heavily focused on knowledge model development, including representation of domain knowledge, student modeling knowledge, and tutoring knowledge. Even most ITS authoring tools focused on saving workload on ITS development; avoidance of knowledge model development was usually a second thought. Only ITS authoring tools for some special domains allow producing ITSs simply by configuration [1]. Knowledge engineering tools, such as DNA [4], have varied potential in reducing the complexity and workload of knowledge model development, but they can not completely avoid knowledge model development.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 152–161, 2008.

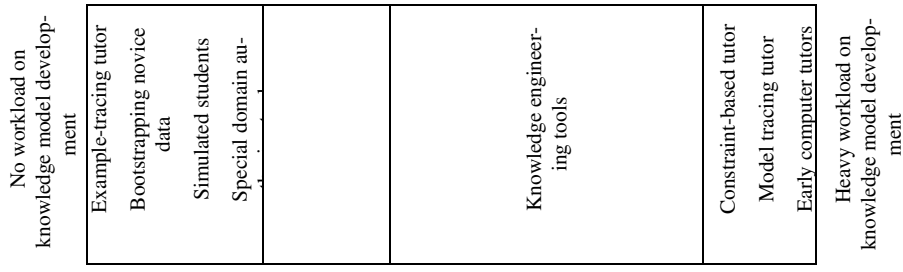


Fig. 1. A spectrum of ITS development approaches

Until recently, a data reuse approach, the development of Example-Tracing Tutor, is proposed for ITS development [2]. Model tracing is performed by reuse of existent student action data instead of production rules. By demonstrating potential student steps, Example-Tracing Tutor is successful in avoiding complex knowledge model development and allows teachers to develop Example-Tracing Tutors by their own. To improve Example-Tracing Tutor development, an approach called bootstrapping novice data was proposed to handle the problem of expert blind spots [5]; the technique of "programming by demonstration" was adopted to turn existent data into production rules to reduce the workload on demonstrating student steps [6]. Since the bootstrapping novice data approach requires two tools, one for data accumulation and the other for actual use, and thus two phases of system deployment, it still deserves more investigation into making data reuse and system development more intuitive and fluent.

It was confirmed that effective computer tutors capture some crucial aspects of the behavior of effective human tutors [7]; a general description on the behavior of most effective ITSs was recently proposed [8]. These works remind us to differentiate the goals from the means and signify the significance of alternative ITS development approaches that do not involve direct knowledge model development. In this study, we investigate such an alternative with interaction data reuse.

2 Interaction Data Reuse

The design of our *interaction data reuse* (IDR) approach was triggered by the work of Active Documents [9], which are aimed to design documents capable of answering reader queries. The key idea of Active Documents is to accumulate human question-answer pairs so as to reuse them in answering subsequent reader queries. If no reusable question-answer pair is found, the answer is delayed until some human expert provides one. Three components are identified in an IDR application. In order to be reused, interaction data must be accumulated. The collection of interaction data to be used in IDR applications is termed interaction data collection (IDC) hereinafter. Besides IDC accumulation, a data retrieval mechanism must also be introduced to locate potentially reusable data for a given situation in the IDC. Since the IDC in an IDR application is unlikely to contain reusable data for all given situations, a mechanism to expand existing IDC is also required when reusable data is missing. Therefore, the three major components of an IDR application are: IDC accumulation, IDC retrieval, and IDC expansion components. Table 1 lists some IDR applications as well as the implementations of each component.

Table 1. IDR applications and implementation of their components

IDR Applications	IDC accumulation	IDC retrieval	IDC expansion
Usenet FAQ	Authoring FAQ files	Reading FAQ files	Editing FAQ files
FAQFinder	Authoring FAQ files	Automated answer retrieval	Editing FAQ files
Active Documents	Automated accumulation of question-answer pairs	Automated answer retrieval	Delayed expert answers

Owing to the need for increased interaction efficiency, an early IDR application was developed by Usenet users, the users of newsgroups, in order to reduce resources spent in providing answers to repeated queries. When new users, called newbies, joined a newsgroup, they were likely to post the same set of questions collectively called *frequently asked questions* (FAQs). In order to avoid repeated efforts, most newsgroups create FAQ files for newbies to read before posting their questions. The maintainers of the FAQ files of a newsgroup update their FAQ files to reflect the changes in the interests of the newsgroup from time to time. In the perspective of IDR, the FAQ files are partial records of the user interaction, or the IDC. The FAQ files (IDC) are created by authoring. The FAQ files are "reused" to "answer" newbie queries when the newbies read them. Thus, the Usenet FAQ is a pure manual IDR application without the support of any IDR automation mechanism.

As the number of newsgroups increased and so did the sizes of the FAQ files, even selecting the right FAQ file for finding the desired answer was no longer simple for ordinary Usenet users. Spotting this information need, the FAQFinder project was launched to automate this answer retrieval task. Users can get their answers simply by input their questions into FAQFinder. FAQFinder retrieves potential answers to a natural language user question simultaneously from many FAQ files using techniques involving statistical information retrieval, syntactic parsing, and semantic concept matching [10]. In the perspective of IDR, FAQFinder simply improves the answer retrieve component of Usenet FAQ files without improving the IDC accumulation and IDC expansion components. If the retrieved answer was unsatisfactory or no answer was retrieved, the user had to turn to other resources. There is no automatic mechanism for expanding the FAQ files that FAQFinder uses. FAQFinder relies on the original authors to expand these FAQ files. In terms of an question-answering application, Active Documents is an improved version of FAQFinder with an mechanism of automatic "FAQ files" generation, in the form of a set of question-answer pairs, and a mechanism for expanding the set of question-answer pairs.

In order to apply the IDR concept to ITS development, a data model other than question-answer pairs must be proposed. Besides, a retrieval component that is capable of dealing with this new data model is also required. However, the assumptions of IDR applications are similar: (1) the same learning obstacles occur repeatedly; (2) The same learning materials are used repeatedly; (3) Missing interaction data become sparse as the IDC size increases above a certain threshold.

3 Tutoring Interaction Data Model

In certain perspective, one-to-one tutoring interactions can be seen as a sequence of student and tutor actions. In such a sequence of actions, the smallest unit is a single student action or a single tutor action. Generally, the number of student actions is greater than the number of tutor actions because eventually the student has to work on the problem sometimes without receiving any feedbacks from the tutor. On the other hand, tutor actions can typically be considered as responses to student actions because during tutoring, each student action generates a different situation to be handled by the tutor. Thus, tutor-initiated actions can also be considered as responses to student actions that are seen as opportunities to promote learning of the student. A tutor action paired with the student action that it responds to is called an *interaction episode* or simply an *episode*. In terms of an episode, the student action decides the *situation* to be handled by the tutor and the tutor action denotes the *response* of the tutor to the situation. Examples of student actions include an incorrect student step to problem solving, a request for hints on the next step, and a student query, and the correspondent tutor actions might be error-specific messages for incorrect student steps, hints on the next step, and answers for student queries. Thus, an episode is consisted of the situational information and the response information, which denotes the student action data and the tutor action data of the episode, respectively.

All contiguous episodes form an *interaction session*, which denotes a session of tutor-student interaction. The interactions that occur during different learning activities belong to different interaction sessions. All the data of interaction sessions comprise the IDC. Thus, interaction sessions, episodes, situational information and response information consist of a hierarchical data model of tutoring interaction. When each interaction session is only consisted of an episode and each episode is consisted of a question and an answer, the data model represents the application to question-answering. Thus, the interaction in question answering is considered a special case of the interaction of problem-solving tutoring.

3.1 The Context of Episodes and Similar Episodes

For the case of question answering, each question is considered unrelated with the others and is generally answered without interference from previously questions. Thus, the data reuse mechanism for question answering is not applicable for tutoring. The context of episodes must be considered. The information used for deciding the context of an episode is termed the contextual information of that episode. In terms of available information for the system, it is reasonable to assume that the maximal amount of contextual information of an episode is all the preceding episodes of that episode. If some other contextual information, such as examination results of the student, were used in the generation of a tutor action but that information was unavailable to the system, then it is unreasonable to expect that the system will be able to identify such differences in the context of episodes. Therefore, all contextual information must be available for the system. Currently, the maximal amount of contextual information of an episode is assumed to be all the preceding episodes of that episode. One way to determine whether two episodes are under similar context is to calculate the similarity of their contextual information.

For the purpose of generality, two categories of similarity measures are proposed in this study: one is literal string matching and the other is the document similarity measures that are used in the field of information retrieval, such as the inner products of the two vectors that represent the two texts to be matched in terms of words used in each text. The literal string matching mechanism works well for matching student problem-solving actions of equation solving, while the document similarity measures work well for question answering, as illustrated in the work of FAQFinder. The use of general similarity measures allows the design of general tools so that teachers can use them simply by configuring the tools.

Besides the choice of similarity measure, the choice of amount of contextual information is also influential in distinguishing the context of episodes. Some choices of contextual information are listed in the following:

(1) Perfect matching: all the situational information and response information of preceding episodes are used as the contextual information. This choice of contextual information is made when the context of episodes are highly sensitive.
(2) Problem-solving path matching: all preceding problem-solving actions are used as the contextual information. This choice is useful for delivering messages like hints on the next steps and error-specific messages for incorrect student steps.
(3) Current situation matching: no information of preceding episodes is used. Instead, the situational information of the episode in question is used as the contextual information. This choice is an approximated version of the previous choice when the problem-solving actions of a student are unlikely to overlap. This choice is typically made to increase the reusability of existent data.

Episodes that are under similar context and share similar situational information are called *similar episodes*. However, similar episodes do not necessarily have the similar response information. The similarity measure of contextual information introduced in last section is also applicable to situational information matching. When similar episodes are retrieved, their response information is used as the system responses. If multiple similar episodes with different response information are retrieved, this means a tutor has several options to respond to the student in that situation under the given context.

4 Generation of Tutoring Actions with IDR

A computer-mediated environment for tutoring is assumed if tutoring support is to be provided by reusing existent interaction data with computers. A consistent computer-mediated environment for data accumulation and data reuse is essential to the success of IDR applications, as different environments may render the meanings of the same tutoring messages different. A consistent computer-mediated environment also allows simultaneous occurrence of data accumulation and data reuse during system deployment. Fig. 2 gives an overview of our proposed approach to applying the IDR concept to provide tutoring support in such a computer-mediated environment, which is consisted of a student interface, a teacher interface, and a backend server.

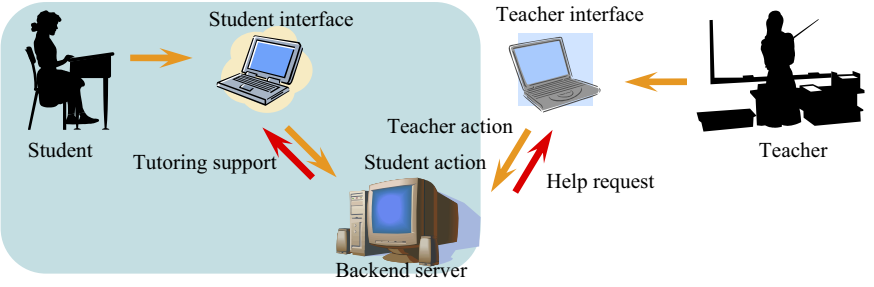


Fig. 2. IDR application overview

The student interface is for the student to work on given problems, to seek for helps, and to receive tutoring support provided by the teacher or the server. The teacher interface is for the teacher to provide tutoring support to the student either spontaneously or upon requests. The backend server receives both student action data and teacher action data from relevant interfaces, and executes the functions of the three components of an IDR application: IDC accumulation, IDC retrieval, and IDC expansion. Implementation of the three components is usually completed before commencing the tutoring application, though it is also possible to defer the implementation of IDC retrieval and IDC expansion components after data accumulation. Implementation of the three components is discussed in the next section.

The backend server carries out the dual roles of a provider of tutoring support and of a broker of the tutoring support provided by the teacher, and this characteristic of the backend server makes the integration of the two sources of tutoring support seamless. In an ideal scenario, the backend server can interact with the student without the involvement of the human teacher when sufficient amount of data has been accumulated, as shown in the shaded area of Fig. 2.

In order to be more general, the procedures are described more formally:

Definition: Given c, the content to be learned, let a be an *action* taken by *agent* p to attain the learning goal, f, the *feedback* given by *agent* q to assist p in attaining the goal; and h, the course of interaction between p and q before action a occurs. Define a *situation* of providing feedback as a pair $(h, (a, p))$ and an *episode* of interaction between p and q as a triplet $(h, (a, p), (f, q))$ with respect to c.

1. Collect and index episode $(h, (a, p), (f, q))$ in IDC M. Let s denote the situational part of an episode e, or $(h, (a, p))$, r denote the feedback part of e, or (f, q), and (s, r) also denote e. Therefore, $M = \{(s_1, r_1), (s_2, r_2), \ldots, (s_n, r_n)\}$, $S = \{s \mid (s, r) \in M\}$ denotes all the situations in M.
2. Select a difference metric Δ, a threshold δ, and a mapping g such that for a situation z:$g(z) = r_k$ if $\Delta(z, s_k) = min(\{\Delta(z, s) \mid s \in S\}) < \delta$, and $g(z)$ is undefined in other cases.
3. Given a situation s, the output is $g(s)$. If $g(s)$ is undefined or unsatisfactory, then query q for f, $M = M \cup \{(s, (f, q))\}$
4. Repeat step 3.

5 Tutoring Support Simulation

A quiz with three equation-solving problems was conducted to a class of 51 middle-school students, who had learned to solve such equations for three weeks prior to the quiz. The three equations are: $3x-4=2x-1$, $2[3(2x-5)-1]=7x+3$, and $\frac{2x+5}{4}-3(x-\frac{1}{2})=1$.

Each student answer is consisted of several student steps. The data repetition rate of the student answers is high but tends to drop as problem complexity increases. Most correct student steps are repeated and the repetition rate of both correct and incorrect student steps tends to increase as their number increases. However, the amount of data in this quiz is still small. Further investigation is needed before drawing a conclusion.

In similar quizzes, the teacher of the class will provide feedbacks to incorrect student steps as a kind of remediation during marking the test papers. To acquire such marking information as a source for simulation of tutoring support, a system was developed in accordance with the IDR approach to assist the teachers in marking test papers by providing suggestions by reusing previous marking information. To simulate the provision of tutoring support to students, the collected interaction data are separated into two sets: one as the set of interaction data to be reused, called the original set, and the other as incoming student actions, called the new set. For each student action from the new set, the original set is searched for responses to this student action. If there is an available response, this reused response is compared with the response contained in the new set given by the teacher. Thus, we can calculate the percentage of matched responses in this simulation. There are two kinds of responses to each student action. One is the correctness annotation of each student step, and the other is the comments on each student step. The comments on student steps are mainly error-specific messages. If the reused teacher responses match with the true teacher responses, it indicates that the reused teacher responses are appropriate. Although the reused teacher responses may be appropriate even if they do not match with true teacher responses, these cases are temporarily ignored. Therefore, the percentage of matched responses is an indicator of the appropriateness of the reused responses.

Table 2. Percentage of matched comments

Data size	Problem 1	Problem 2	Problem 3
5	100.00%	100.00%	95.80%
15	100.00%	100.00%	96.50%
25	100.00%	100.00%	87.50%
35	100.00%	100.00%	87.00%
45	100.00%	100.00%	83.90%

The simulation results show that correctness annotations on student steps are 100% match for all data sizes. However, with the choice of simple contextual information, the comments on student steps might be inappropriate, as shown in Table 2. However, the percentages of matched comments are still high.

6 Concluding Remarks

The IDR approach is a potential alternative approach to ITS development, though there are still several issues to be investigated. This study lays down a theoretical foundation for the IDR approach to helping teachers in providing tutoring supports with computers to students who are learning problem-solving skills. Specifically, we identify the main components of an IDR application and propose a data model of tutoring interaction data and a set of general procedures for producing tutoring actions. In addition, tutoring support simulation was performed. Comparison between the IDR approach and the common knowledge model development approaches is given in Table 3.

Table 3. Comparison of IDR and knowledge model development approaches

	Knowledge model development	IDR
Rationale	Model generality	Data repetition
Key implementation issue	Mechanisms for knowledge modeling (difficult and labor-intensive)	Mechanisms for data accumulation and matching
Response quality	Appropriate responses	No guarantee in response appropriateness
Required teacher participation	Development phase	Development and early deployment phases
Accumulate new data during deployment	No	Yes
Applicability to multiple learning tasks	Yes	No

The common rationale of developing knowledge models to provide tutoring support is that knowledge models are capable of capturing various knowledge application instances. The goal of knowledge model development is to develop knowledge models that are general enough in capturing desired knowledge application instances. On the contrary, the rationale of the IDR approach is that most of knowledge application instances are likely to be repeated and knowledge application data can be accumulated and reused for repeated instances. Thus, the mechanism to provide tutoring support is through data matching and the key implementation issue is the choice and implementation of data accumulation and data matching mechanisms. Implementation of data accumulation and data matching mechanisms are relatively easier than knowledge model development. This implementation benefit comes at the cost of appropriateness of system responses. If data matching mechanism is not good enough in distinguishing one knowledge application case from another or is compromised to obtain better data reusability, inappropriate data may be delivered as the system responses.

The differences in the rationales and the key implementation issues of the approaches are the sources of other differences. For example, to develop knowledge

models or to maintain knowledge models in order to include newly discovered student mistakes, teacher participation is required in this model development phase. Typically, this occurs when desired knowledge application instances are acquired. Before the completion of knowledge model development, desired tutoring support is unavailable. Once knowledge model development is completed, an additional benefit is that the knowledge models are typically applicable to multiple learning tasks. Therefore, when a lot of learning tasks are required, this benefit of knowledge model development might compensate for its price. Contrarily, early system deployment is an opportunity to accumulate and reuse data for the IDR approach and thus teacher participation is also required during this phase in addition to the development phase. However, the IDR approach is capable of providing tutoring support to the student by the system while accumulating interaction data between the human teacher and the student. This is beneficial because this is a convenient way to accumulate alternative student solutions, student mistakes, and relevant tutoring supports.

Table 4. Comparison of IDR tutor and example-tracing tutor development

	IDR tutor	Example-tracing tutor
Generality	General case of example-tracing tutor	Special case of IDR tutor
Data accumulation	Teacher-student interaction	Demonstration Bootstrapping novice data Authoring
Data expansion	Teacher-student interaction	Demonstration Authoring
Tool support	To be developed	Cognitive Tutor Authoring Tools

Example-tracing tutors are considered as a special case of IDR tutors, as shown in Table 4. The general architecture of IDR tutors may provide clues for new ways to tutor development. For example, it might be interesting to investigate whether there is any data expansion mechanism that can respond to the student without delay. The differences in the data accumulation and data matching mechanisms are also significant. Demonstrating student solutions and authoring relevant feedbacks is less straightforward than directly interacting with students while the students are solving the given problem. Additionally, teachers are unlikely to remember all the cases of student solutions. After deployment, if new student solutions are discovered, there still lacks of a mechanism to directly use the opportunity for data expansion. Instead, data expansion is performed after the deployment phase and is conducted in subsequent modifications.

Acknowledgement

This work is supported in part by NSC under the grants: 96-2524-S-008-001-.

References

1. Murray, T.: An Overview of Intelligent Tutoring System Authoring Tools: Updated Analysis of the State of the Art. In: Murray, T., Blessing, S., Ainsworth, S. (eds.) Authoring Tools for Advanced Technology Learning Environments: Towards cost-effective adaptive, interactive and intelligent educational software, Kluwer, Dordrecht (2003)
2. Koedinger, K.R., Aleven, V., Heffernan., T., McLaren, B., Hockenberry, M.: Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration. In: The Proceedings of 7th Annual Intelligent Tutoring Systems Conference, Maceio, Brazil (2004)
3. Mitrovic, A., Suraweera, P., Martin, B., Zakharov, K., Milik, N., Holland, J.: Authoring Constraint-Based Tutors in ASPIRE. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 41–50. Springer, Heidelberg (2006)
4. Shute, V.J., Torreano, L.A., Willis, R.E.: DNA – Uncorking the Bottleneck in Knowledge Elicitation and Organization. In: Goettl, B.P., Halff, H.M., Redfield, C.L., Shute, V.J. (eds.) ITS 1998. LNCS, vol. 1452, pp. 146–155. Springer, Heidelberg (1998)
5. McLaren, B.M., Koedinger, K.R., Schneider, M., Harrer, A., Bollen, L.: Bootstrapping Novice Data: Semi-Automated Tutor Authoring Using Student Log Files. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220. Springer, Heidelberg (2004)
6. Matsuda, N., Cohen, W.W., Koedinger, K.R.: Applying Programming by Demonstration in an Intelligent Authoring Tool for Cognitive Tutors. In: AAAI Workshop on Human Comprehensible Machine Learning. AAAI association, Menlo Park (2005)
7. Merril, D.C., Reiser, B.J., Ranney, M., Trafton, J.G.: Effective Tutoring Techniques: A Comparison of Human Tutors and Intelligent Tutoring Systems. The Journal of Learning Sciences 2(3), 27–305 (1992)
8. VanLehn, K.: The Behavior of Tutoring Systems. International Journal of Artificial Intelligence in Education 16, 227–265 (2006)
9. Heinrich, E., Maurer, H.: Active Documents: Concept, Implementation and Applications. Journal of Universal Computer Science 6(12), 1197–1202 (2000)
10. Hammond, K., Burke, R., Martin, C., Lytinen, S.: FAQ Finder: A Case-Based Approach to Knowledge Navigation. In: Working Notes of the AAAI Spring Symposium on Information Gathering in Heterogeneous Environments, pp. 69–73. AAAI (1995); Modified version also appeared in Proceedings of the 11th Conference on Artificial Intelligence for Applications. IEEE Computer Society Press, Los Angeles (1995)

Towards Collaborative Intelligent Tutors: Automated Recognition of Users' Strategies

Ya'akov Gal[1,2], Elif Yamangil[2], Stuart M. Shieber[2], Andee Rubin[3], and Barbara J. Grosz[2]

[1] MIT Computer Science and Artificial Intelligence Laboratory
[2] Harvard School of Engineering and Applied Sciences
[3] TERC

Abstract. This paper addresses the problem of inferring students' strategies when they interact with data-modeling software used for pedagogical purposes. The software enables students to learn about statistical data by building and analyzing their own models. Automatic recognition of students' activities when interacting with pedagogical software is challenging. Students can pursue several plans in parallel and interleave the execution of these plans. The algorithm presented in this paper decomposes students' complete interaction histories with the software into hierarchies of interdependent tasks that may be subsequently compared with ideal solutions. This algorithm is evaluated empirically using commercial software that is used in many schools. Results indicate that the algorithm is able to (1) identify the plans students use when solving problems using the software; (2) distinguish between those actions in students' plans that play a salient part in their problem-solving and those representing exploratory actions and mistakes; and (3) capture students' interleaving and free-order action sequences.

1 Introduction

We report on the development of algorithms for recognizing students' plans when interacting with pedagogical systems for data-generation and analysis. This work is a first step towards building a collaborative pedagogic agent that will support students in their problem-solving and teachers in their analysis of students' modeling and understanding of statistical data. TinkerPlots, the system we use in this paper, gives students great flexibility in representing and analyzing statistical data. It is in essence a data-analysis "construction kit" that allows students to create and analyze a large number of statistical models [8]. While this makes for a rich educational environment, it does pose significant problems for teachers. When an entire classroom of students is using TinkerPlots at the same time, there is no way for a teacher to keep track of what each child is doing, especially since they may be following divergent paths in solving the problem. Without some sort of support, teachers are left with the end-result of students' work on the computer screen, or looking over the shoulder of each student for at most a minute or two.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 162–172, 2008.

Automatic plan recognition is an open problem in AI, and the task of recognizing users' plans when interacting with software systems is particularly challenging. Ideally designed systems are flexible, allowing users the convenience of choosing among multiple action sequences for performing the same function, and the ability to perform these action sequences in relatively free order. Traditional algorithms for plan recognition assume a goal-oriented agent whose activities are consistent with its knowledge base and who forms a single encompassing plan [9]. In contrast, one of the objectives of flexible pedagogical software is to allow students to explore and experiment during their interaction process. In these settings, students may interchangeably pursue multiple, interleaving plans; they may be confused about which appropriate plan to take, and they may make mistakes. Recognizing students' actions by exhaustively considering every possible way in which a student can use these systems is infeasible.

This paper describes a computationally tractable algorithm for intelligently recognizing students' problem-solving strategies based on their complete interaction history with the system. The algorithm composes the action sequences from a user's interaction into a series of interdependent plans. It infers the plan that the user was using to complete each activity, and compares this plan with an ideal solution that was designed by domain experts. At the end of this process, the algorithm outputs a hierarchy of the plans that students used during the session and the extent to which they differed from the ideal solutions.

The algorithm was tested using the commercial system TinkerPlots, used world-wide to teach students in grades 4-8 about statistics and mathematics [5]. In TinkerPlots, students actively model stochastic events and construct models that generate data. TinkerPlots is highly flexible, allowing for data to be modeled, generated, and analyzed in many different ways using an open-ended interface. Our empirical studies focused on two different problems in which students used TinkerPlots to model and analyze stochastic data.

In AI, plan recognition has been used in a range of applications, such as modeling discourse structure from speech and inferring transportation routines from GPS data [11,10]. Past work in the intelligent tutoring domain has focused on inferring students' activities for the purpose of providing feedback by the tutor. These models have been used for modeling how students solve math [6,3] and physics problems [4,14], their help requests from pedagogical software or their misuse of it [13,2]. Many of these works construct a probabilistic model of students' problem-solving strategies that is subsequently used to update the tutor's beliefs about students' likely future actions given their behavior. In these cases, the tutor is an active participant in the student's learning process and ambiguities or uncertainties about the students' plan of action are resolved by querying the student [1].

In contrast, the work reported in this paper addresses the problem of recognizing students' actions given their complete interaction histories. The system does not intervene with the student's activities during the course of interaction. Straightforward adaptation of probabilistic techniques for this purpose is difficult, because the size of probabilistic models is typically exponential in the length

of the history they consider, and students' complete interaction histories often span hundreds of actions. In addition, the model parameters must be trained from data or stipulated by a domain expert. Both of these techniques require considerable effort in the domain we consider.

1.1 Example Scenario: The Two-Dice Problem

To illustrate the algorithm we will use the following example, drawn from a set of problems posed to seventh grade students using TinkerPlots. "We rolled two dice over and over a huge number of times and kept track of their sums. For example, when the first die came up 5 and the second die came up 6, we recorded their sum of 11. Using TinkerPlots, build a model that you can use to roll two dice 1,000 times and see whether 11 came up more often than 12."

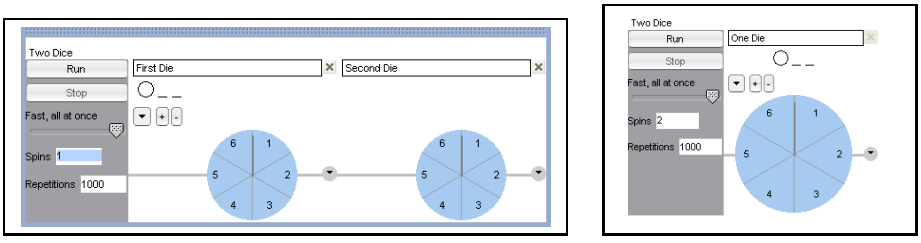
The purposes behind this exercise are for children to learn about the joint distribution of non-ordered random events and to explore how sample distributions vary, even if they are drawn from the same population. Each roll of two dice generates a pair of values, one for each die. There are two events that make up the sum 11, namely (5,6) and (6,5), while there is only one event that makes up the sum 12, namely (6,6). Since each of these events is equally likely, in theory the sum 11 will occur more often than the sum 12. (Of course, as students run their models, they will discover that, while this is generally true, there will be samples in which there are more 12s than 11s, especially if the sample size is small.)

One of the possible approaches towards modeling this situation using TinkerPlots is shown in Figure 1. The model includes a sampler device comprising two spinners, shown in Figure 1(a), each of which is a model of one die. The sampler will randomly select a value for each of its spinners every time it is run. Each spinner has six possible values. The surface area specified for each value determines its weight in the sample. Effectively, this sampler models a joint probability distribution over two independent random variables with six values distributed uniformly. The value of "Repetitions", set to 1,000 in this example, determines the number of times the sampler is run. The value of "Spins", set to "1" in this example, determines the number of rolls of the two dice at each run.

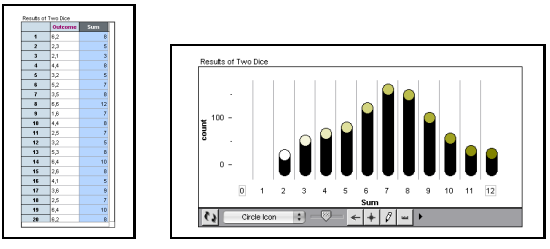
Figure 1(b) shows some of the data generated by the sampler once it has run. Each pair in the table represents a roll of two dice. This pair has been separated, by instigating a "Separate Individual Draws" function in the sampler. To the right is a graphical representation of all of the sums in the form of a histogram. Figure 1(a) shows an additional way to model this problem. Here, a single die is used that is thrown twice at each repetition, hence the value of "spins" is set to "2". There are many other ways to use TinkerPlots to solve this problem.

2 Recipes, Planning and Plan Recognition

Students interact with TinkerPlots through a series of rudimentary operations that create, modify or delete objects such as spinners, plots, and outcomes. We



(a) Two Possible Sampler Models



(b) Displaying Sampler Data as a Histogram

Fig. 1. TinkerPlots Sessions Snapshot

will use the term *basic actions* to refer to these operations, which can often be carried out by a single keystroke or mouse action. TinkerPlots interactions are recorded as a linear sequence of basic actions in order of their occurrence. Each basic action uses a unique tag to refer to an object, which is transparent to the user. A subset of such an interaction sequence is shown in the leaves of the trees in Figure 3. For example, the basic action $\mathsf{New}(\mathsf{Spinner}(S_1))$ adds a new spinner with ID S_1. These actions are serially labeled in order of occurrence. (Due to layout constraints, the leaves in this figure are not aligned on the same plane.)

We model students' reasoning about problems using abstract entities, called *complex actions*, which capture higher-level, more abstract TinkerPlots activities, such as adding two dice to a sampler, computing the sum of a roll of two dice, or fitting sampler data to plot. Complex actions can be decomposed into sub-actions [7]. A sub-action can be a basic TinkerPlots action or it can be a complex action itself. A useful distinction between complex and basic actions is that students can "see" both basic and complex actions, while the TinkerPlots system can only "see" and register basic actions.

A *recipe* for a complex action is an ideal sequence of operations for fulfilling the complex action. Formally, a recipe is a set of sub-actions and constraints such that performing those sub-actions under those constraints constitutes completing the action [12]. These sub-actions are referred to as the recipe's constituents. Figure 2 presents recipes for solving the two-dice problem and its constituent sub-actions. Each recipe for a complex action is represented as a tree of depth two, in which the leaves correspond to the recipe's constituent actions (whether basic or complex), and the root corresponds to the complex action. Basic actions

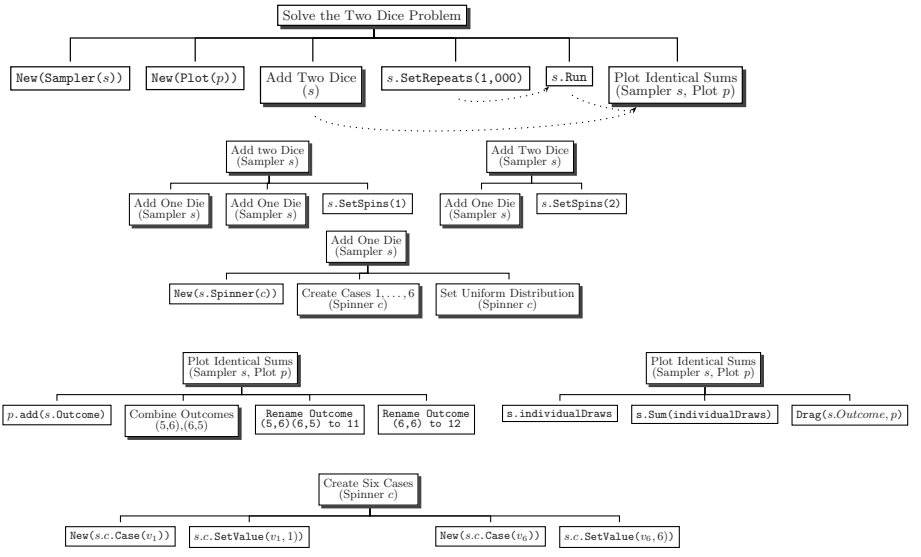


Fig. 2. Recipes for Solving the Two-Dice Problem. Dashed edges represent temporal constraints between actions.

are outlined in plain boxes, while complex actions are outlined in shadowed boxes. TinkerPlots objects are identified by a unique tag, and recipe actions use parameters to refer to the TinkerPlots objects they modify. For example the recipe for the complex action AddTwoDice (s) modifies the sampler object that is bound to the parameter s.

The order in which actions are performed in a recipe can be constrained by including temporal constraints between actions, represented as a dotted edge. Actions within the same recipe can occur in any order as long as they meet the specified temporal constraints. For example, in the recipe for the action SolveTheTwoDiceProblem, both actions AddTwoDice (s) and s.SetRepeats(1,000) can come in any order as long as they both occur before the basic action s.Run.

In addition to the constraints embedded in the recipes, some action combinations are disallowed by the TinkerPlots system itself. For example, it is impossible to add a spinner to a sampler until the sampler has been created. For expository convenience, we do not show these constraints in the recipes.

Recipes may be ambiguous, in the sense that there may be several recipes for completing the same complex action. For example, Figure 2 shows two possible recipes for completing the complex action AddTwoDice (s). One possible recipe uses a single die that is rolled twice. It includes the sub-actions AddOneDie (s) and s.SetSpins(1). The other recipe uses two dice that are rolled once. It includes two sub-actions AddOneDie (s) and the sub-action s.SetSpins(1). In addition, the same action may be a constituent of several different recipes. For example, the complex action AddOneDie (s) appears in both recipes for the complex action AddTwoDice (s).

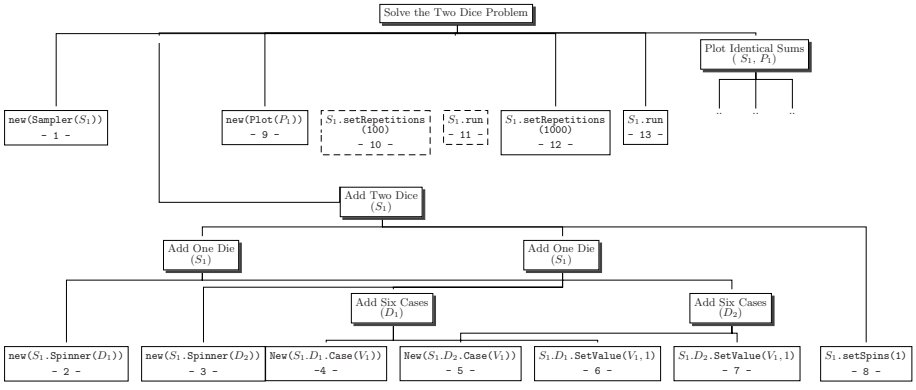


Fig. 3. A Sample Plan

2.1 Planning

Planning is the process by which students use recipes to compose basic and complex actions towards completing tasks using TinkerPlots. We say that a recipe for a complex action is *fulfilled* by a set of temporally-ordered sub-actions if (1) there is a one-to-one correspondence from each of the sub-actions to one of the recipe's constituents; (2) all of the sub-actions agree on the identification tags for the TinkerPlots objects that are modified by the recipe; and, (3) the order between sub-actions is consistent with the temporal constraints that are defined between recipe constituents. Formally, a plan is an ordered set of basic and complex actions, such that each complex action is decomposed into sub-actions that fulfill a recipe for some task. Each time a recipe for a complex action is fulfilled in a plan, there is an edge from the complex action to its sub-actions, representing the recipe constituents. For example, in Figure 3, the recipe for the complex action AddTwoDice(S_1) is fulfilled by the two AddOneDie(S_1) actions and the action S_1.SetSpins(1).

Each tree in Figure 3 represents a plan that was carried out by the student. The leaves of the trees represent the basic actions corresponding to the user's interaction history. (For expository convenience, we have only included a subset of this interaction history.) The plan that emanates from the complex action SolveTheTwoDiceProblem shows that the student was able to complete the two-dice problem.

In a plan, the constituent sub-actions of complex actions may interleave with other actions. In this way, the plan combines the free-order nature of TinkerPlots recipes with the exploratory nature of students' learning strategies. Formally, we say that two ordered complex actions *interleave* if at least one of the sub-actions of the first action occurs after some sub-action of the second action.

An example of interleaving actions in this plan are the two complex actions AddOneDie (S_1) We can see this because a constituent of the recipe for the first AddOneDie (S_1) (the action AddSixCases (D_2))) occurs after a constituent of the recipe for the second AddOneDie (S_1) (the action AddSixCases(D_1)). In

Figure 3, there are crossing edges between the constituent sub-actions of any two interleaving actions.

Also shown in Figure 3 are two basic actions outlined in dashed boxes (S_1.SetRepeats(100) and S_1.run) that were not necessary for solving the two-dice problem. This happened because the user first ran the sampler for 100 repetitions, before running the sampler for 1,000 repetitions, as required by the problem formulation. These could represent a student's exploration or a mistake.

3 Plan Recognition

The task of *plan recognition* in the TinkerPlots domain is to infer students' plans based on their interaction history and a set of recipes. A naive approach would search through the space of all possible plans that are consistent with a user's interaction, the recipes, and their constraints. This approach is not feasible. For each possible action in the plan, we would need to consider all possible expansions of basic and complex sub-actions as long as their order is permitted by the recipe constraints. In the worst case, the number of possible plans to consider will be factorial in the number of basic actions in an interaction sequence.

However, certain qualities of the TinkerPlots domain serve to constrain the search process. First, it is not possible to generate an infinite plan using TinkerPlots recipes. Therefore, we can choose the recipes to be fulfilled in an incremental fashion, ordered by depth. We define the "depth" of a recipe for a complex action as the maximum depth of the tree for any plan to complete the complex action[1]. Second, the sub-actions of a complex action will always agree on the identification tags of those TinkerPlots objects that are modified by the complex action. Therefore, we do not need to consider any action combination that disagree on the ID tags. We can also ignore those action combinations disallowed by the TinkerPlots system.

As a result, we can construct the following algorithm that incrementally builds a sequence of plans to explain a user's interaction history from the leaves upwards. Each step t of the algorithm maintains an ordered set of actions, denoted P_t. Each of these actions is a root of a tree that is a partial plan that explains some subset of the user's interaction history. P_0 is initialized to include all of the basic actions in the interaction history. Let G be the set of recipes, and let the recipe in G for a complex action C be denoted as R_C. The algorithm proceeds as follows:

For each R_C in G, sorted by depth
  Initialize P_{t+1} with P_t
  For any sequence S_{t+1} of actions in P_{t+1} that fulfill R_C:
    Add a new action C in P_{t+1}, positioned after the first action in S_{t+1}
    Let S_t be the set of actions in P_t corresponding to S_{t+1}
    Add edges from C in P_{t+1} to all actions in S_t
    Remove all actions in S_{t+1} from P_{t+1}

[1] For example, the depth of the recipe for solving the two-dice problem is three, because there is no possible plan for this task whose depth is greater than three.

A key step in this algorithm is the selection of actions in P_{t+1} to fulfill the recipe for R_C. We select these actions in any order that is allowed by the temporal constraints of the recipe for R_C. In particular, the actions in P_{t+1} may be non-contiguous; this allows the algorithm to capture interleaving plans. This step is greedy, because once a complex action is chosen to fulfill R_C, it is removed from P_{t+1} and will not be considered again. Therefore, there may be instances where the algorithm fails because it picked the wrong actions to fulfill a recipe in earlier steps. An interesting consequence is that the order in which we traverse the actions in P_{t+1} can affect the way in which the algorithm fulfills recipe, and thus, its output. We currently do so by traversing P_{t+1} sequentially, from the last action to the first. A different order may fulfill different recipes, and produce a different plan.

The complexity of this approach can be computed as follows. Let n be the length of a student's interaction sequence. The number of times a recipe can be fulfilled is bounded by n. In the worst case, it will take a complete pass over the actions in P_t to fulfill the recipe. The number of actions in P_t is bounded by n. Therefore, it will take at most n^2 steps to exhaust all of the possible applications of R_C. In fact, a slightly more sophisticated implementation complete this process in linear time. Given that the size of the recipes is constant, we conclude that the complexity of the algorithm is quadratic in the length of the interaction history.

We demonstrate part of this process in Figure 4. We show the complete interaction history of the user in Figure 4(a), outlined in bold. (This is the same interaction history of the plan in Figure 3). These actions are presented top-to-bottom in order of their occurrence. Each sub-figure in Figure 4 shows the partial plans that the algorithm maintains for recipes of a given depth. When fulfilling a recipe for a complex action, we draw directed edges from the sub-actions to the complex action. For instance, in the step shown in Figure 4 (b), the algorithm fulfills two separate instances of the recipe AddSixCases (d), one for spinner ID D_1 and one for spinner ID D_2. These complex actions interleave, as can be seen from the crossing edges.

In the step shown in Figure 4(d), the algorithm chooses to fulfill one of two possible recipes for completing the complex action AddTwoDice (s). This choice is possible because of the basic action S_1.SetSpins(1), which is a unique constituent action for one of the recipes for CreateTwoDice (s) but not for the other. In the step shown in Figure 4(e), the algorithm succeeds in collapsing the complex action Solve Two-Dice Problem, and terminates, because it cannot fulfill any more recipes. As shown in the final step in Figure 4(f), the algorithm has determined that two actions (S_1.SetRepeats (100) and S_1.Run) were redundant.

3.1 Evaluation

We collected eight TinkerPlots interaction histories of six people using TinkerPlots to solve the two-dice problem, and two people using TinkerPlots to solve another problem involving the modeling of ordered stochastic events. Two of these people were middle school students in an after-school TinkerPlots club. Three were adults who had experience with using technology in education, but not with TinkerPlots. One was an adult who was very familiar with TinkerPlots.

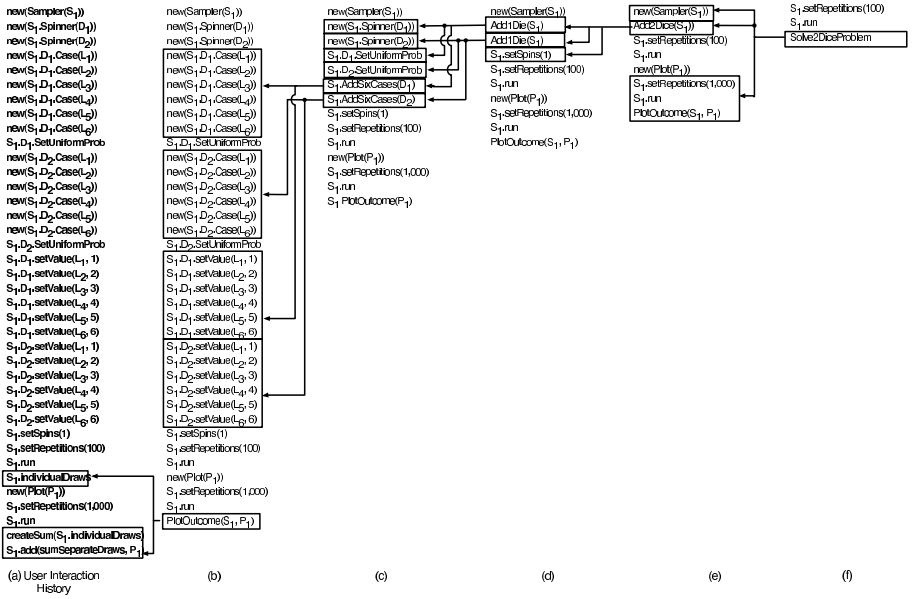


Fig. 4. The execution of the algorithm on a user's interaction history. Crossing edges represent interleaving actions.

The middle school students had been using TinkerPlots for several months and did the two-dice problem as part of their regular after-school work. The three adults who were not familiar with TinkerPlots watched a 5-minute introductory video, saw a brief demonstration of the Sampler functions, then did the problem. In all cases, an experimenter tracked the activities of each participant (e.g., what samplers were created, when actions were interleaved, etc.).

We considered the plan constructed by an algorithm to be "correct", if the actions in the plan corresponded to the students' actual activities using the software, including the interleaving of action sequences. The algorithm was able to recognize the strategies for all of these interaction histories but one. In this instance, a student solved the two dice problem twice, using the same sampler in both solutions. The algorithm recognized one solution, but not the other. This is because the plan recognition algorithm grows a sequence of trees, so the same action cannot simultaneously fulfill several recipes.

One approach to be able to consider all ways of fulfilling a recipe, is to build possible partial plans in parallel, rather than greedily. Once the current partial plan reaches a dead-end, the algorithm backtracks. Because partial plans in TinkerPlots may interleave, there is no straightforward way to accomplish this without having to construct a separate partial plan for each possible way a recipe can be fulfilled. This naive approach is exponential in the length of interaction, hence computationally intractable. We intend to see whether dynamic programming can be used to make this process more efficient.

Lastly, it is important to note that even a partial account of students' interactions can still convey information about the techniques they used and their approach that is useful to teachers. For example, the greedy algorithm was still able to recognize all of the constituent actions for the second application of the recipe for the two-dice problem.

4 Conclusion and Future Work

This work presented a simple and computationally efficient algorithm for recognizing students' interactions with data-modeling software. The algorithm is able to capture the nature of interaction of users with flexible computer software, which allow users to interleave their activities in relatively free order. We showed that the algorithm was successfully able to recognize students' plans when solving two separate problems using a commercially available application.

This work is a first step towards a pedagogical agent that is truly collaborative, in the sense that it provides the right machine-generated support for its users. For teachers, this support consists of notification of students' performance both after and during class. For students, this support will guide their problem-solving in a way that maximizes their learning experience while minimizing interruption.

To this end, we are currently pursuing work in several directions. First, we are constructing vivid, coherent representations of students' plans to show teachers. These presentations need to support a "birds' eye view" of class performance during a session, as well as the ability to focus on the behavior of individual students. We will develop algorithms that enable teachers to access the state of the system at critical points in students' work. The system state conveys different information from a plan, in that it provides a snapshot of the TinkerPlots objects a user is using at a given point in time, rather than a post-session analysis of students' interaction. Our future research will include developing algorithms for keeping track of the state of the system and experimenting with presenting teachers with some combination of plan information and state information. Lastly, we are pursuing a machine-learning approach towards learning recipes from data by observing students' interaction.

Acknowledgements

This work is supported by NSF grant number REC-0632544. Thanks very much to Heather Pon-Barry and Swapna Reddy for useful comments and discussion.

References

1. Anderson, J.R., Corbett, A.T., Koedinger, K., Pelletier, R.: Cognitive tutors: Lessons learned. The Journal of Learning Sciences 4(2), 167–207 (1995)
2. Baker, R.S., Corbett, A.T., Koedinger, K.R., Roll, I.: Generalizing detection of gaming the system across a tutoring curriculum. In: Proc. of 8th Internatioanl Conference on Intelligent Tutoring Systems (2006)

3. Beck, J.E., Wolf, B.P.: Using a learning agent with a student model. In: Proc. of 4th international conference on Intelligent Tutors (1998)
4. Conati, C., Gertner, A., VanLehn, K.: Using bayesian networks to manage uncertainty in student modeling. Journal of User Modeling and User-Adapted Interaction, 12(4), 371–417 (2002)
5. Miller, C., Konold, C.: TinkerPlots Dynamic Data Exploration 1.0. Key Curriculum Press (2004)
6. Corebette, A., McLaughlin, M., Scarpinatto, K.C.: Modeling student knowledge: Cognitive tutors in high school and college. User Modeling and User-Adapted Interaction 10, 81–108 (2000)
7. Grosz, B.J., Kraus, S.: The evolution of sharedplans. Foundations and Theories of Rational Agency, 227–262 (1999)
8. Hammerman, J.K., Rubin, A.: Strategies for managing statistical complexity with new software tools. Statistics Education Research Journal 3(2), 17–41 (2004)
9. Kautz, H.: A formal theory of plan recognition. PhD thesis, University of NY, Rochester (1987)
10. Liao, L., Patterson, D.J., Fox, D., Kautz, H.: Learning and inferring transportation routines. Journal of Artificial Intelligence Research 171, 311–331 (2007)
11. Lochbaum, K.: A collaborative planning model of intentional structure. Computational Linguistics 4, 525–572 (1998)
12. Pollack, M.: Plans as complex mental attitudes. MIT Press (1990)
13. Roll, I., Aleven, V., McLaren, B.M., Koedinger, K.R.: Can help seeking be tutored? In: International Conference on Artificial Intelligence in Education 2007 (2007)
14. Vanlehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R.H., Taylor, L., Treacy, D.J., Weinstein, A., Wintersgill, M.C.: The Andes physics tutoring system: Lessons learned. International Journal of Artificial Intelligence and Education 15(3) (2005)

Automatic Generation of Fine-Grained Representations of Learner Response Semantics

Rodney D. Nielsen[1,2], Wayne Ward[1,2], and James H. Martin[1]

[1] Center for Computational Language and Education Research, CU Boulder
[2] Boulder Language Technologies
{Rodney.Nielsen,Wayne.Ward,James.Martin}@ Colorado.edu

Abstract. This paper presents a process for automatically extracting a fine-grained semantic representation of a learner's response to a tutor's question. The representation can be extracted using available natural language processing technologies and it allows a detailed assessment of the learner's understanding and consequently will support the evaluation of tutoring pedagogy that is dependent on such a fine-grained assessment. We describe a system to assess student answers at this fine-grained level that utilizes features extracted from the automatically generated representations. The system classifies answers to indicate the student's apparent understanding of each of the low-level facets of a known reference answer. It achieves an accuracy on these fine-grained decisions of 76% on within-domain assessment and 69% out of domain.

1 Introduction

This paper presents a new representation for learner responses that captures their fine-grained semantics and can be automatically extracted from the discourse. The intent of the fine-grained representation is to facilitate more detailed assessment and consequently more specific and effective intelligent tutoring system (ITS) feedback (c.f. [7]). Two of the most significant reasons for automatic extraction of the representation are to support an ITS that can provide domain-independent tutoring in the long-term and to facilitate easier domain adaptation in the short-term.

Current automated tutors assess learner responses at a fairly coarse-grained level, often only classifying the response as having expressed the desired information or not. There are other researchers who are striving to provide more detailed feedback (c.f., [6], [7], [19], [22], [25]), but this work is all very domain-dependent, often requiring hand-generated knowledge representation ontologies, logic representations, extraction frames, and parsers, or requiring the collection of 100 or more student answers for each new question in order to train new machine learning classifiers. Similarly, work in the area of large-scale assessment requires significant tuning of information extraction patterns and knowledge representations or the retraining of classifiers, in each case necessitating the collection of 100 or more learner responses for each new question (e.g., [2], [11], [13], [20]).

The goal of the representation described here is to facilitate domain-independent assessment of student responses to questions in the context of a known reference

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 173–183, 2008.

answer and to perform this assessment at a level of detail that will enable more effective ITS dialog. We have two key criteria for such a representation: 1) it must provide a level of detail that facilitates a more fine-grained assessment of the learner's understanding, indicating exactly *where* and *in what manner* the answer did not meet expectations and 2) the representation and assessment must be *learnable* by a system – they should not require the handcrafting of domain-specific representations.

In the following section, we describe the corpus of student answers we utilize to evaluate automatic answer assessment according to our representation and the level at which this assessment is performed. In section 0, the focus of this paper, we detail the process of automatically generating our fine-grained semantic representation from answers given in natural language. In section 0, we discuss automated assessment experiments providing evidence for the reasonableness of our representation. We close with a discussion of future work and a summary of the contributions of the present work.

2 Creating a Test Corpus

We acquired data from 3rd-6th grade students utilizing the Full Option Science System (FOSS), a proven research-based system that has been in use across the country for over a decade. The data was gathered by the Assessing Science Knowledge (ASK) project [10]. FOSS includes sixteen diverse science teaching and learning modules covering life science, physical science, earth and space science, scientific reasoning, and technology. For each module, the FOSS research team designed a set of summative assessment questions with reference answers, from which we selected 287 constructed response questions in line with our goals. These questions had expected responses ranging in length from moderately short verb phrases to several sentences (not fill-in-the-blank or subjective). We generated a corpus by transcribing a random sample of the students' handwritten responses. In total, approximately 16,000 student responses were transcribed. While these answers were not taken from a tutoring session, the questions are representative of our target ITS. We are currently gathering data from live tutoring sessions to further evaluate our representation and assessment approach.

2.1 Reference Answer Representation

The ASK assessments included a reference answer for each of their constructed response questions. We decomposed these reference answers into low-level facets, roughly extracted from the relations in a syntactic dependency parse and a shallow semantic parse. The decomposition is based closely on these well-established frameworks, since these representations have been shown to be learnable by automatic systems (c.f., [4], [16]). These facets are the basis for assessing learner answers. The focus of this paper is on the *automatic generation* of a similar representation for both the reference answers and learner answers. Section 0 details the representation that is generated automatically, along with a description of the process for generating this representation. Here we simply sketch the makeup of the final assessed reference answer facets – the process of extracting these facets is detailed in [14].

Example 1 presents a reference answer from the Magnetism and Electricity module and illustrates the facets derived from its dependency parse, along with their glosses. These facets represent the fine-grained knowledge that the student is expected to address in their response.

(1) The brass ring would not stick to the nail because the ring is not iron.
(1a) NMod(ring, brass)
(1a') The ring is brass.
(1b) Theme_not(stick, ring)
(1b') The ring does not stick.
(1c) Destination_to_not(stick, nail)
(1c') Something does not stick to the nail.
(1d) Be_not(ring, iron)
(1d') The ring is not iron.
(1e) Cause_because(1b-c, 1d)
(1e') 1b and 1c are caused by 1d.

We refer to facets that express relations between higher-level propositions as inter-propositional facets. An example of such a facet is (1e) above, connecting the proposition *the brass ring did not stick to the nail* to the proposition *the ring is not iron*. In addition to specifying the headwords of inter-propositional facets (*stick* and *is*, in 1e), we also note up to two key facets from each of the propositions that the relation is connecting (b, c, and d in example 1). Reference answer facets that are assumed to be understood by the learner a priori, (generally because they are part of the information given in the question), are also annotated to indicate this.

There were a total of 2878 reference answer facets, with a mean of 10 facets per question (median of 8). Facets that were assumed to be understood a priori by students accounted for 33% of all facets and inter-propositional facets accounted for 11%. The experiments in automated annotation of student answers (section 0) focus on the facets that are not assumed to be understood a priori (67% of all facets); of these, 12% are inter-propositional.

A total of 36 different facet relation types were utilized. The majority, 21, utilize VerbNet thematic roles [8], (e.g., Agent, Patient and Instrument). Direction, Manner, and Purpose were added from PropBank adjunctive argument labels [18]. Quantifier, Means, and Cause-to-Know were added based on an analysis of the corpus. Additionally, copulas and similar verbs (e.g., be, become, do, and have) were themselves considered to be facet relation types connecting their arguments. Finally, anything that did not fit into the above categories retained its dependency parse type: VMod (Verb Modifier), NMod (Noun Modifier), AMod (Adjective or Adverb Modifier), and Root (Root was used when a single word in the answer, typically yes, no, agree, disagree, A-D, or a number, stood alone without a significant relation to the remainder of the reference answer; this occurred only 23 times, accounting for fewer than 1% of the reference answer facets). The seven highest frequency relations are NMod, Theme, Cause, Be, Patient, AMod, and Location, which together account for 70% of the reference answer facet relations.

2.2 Annotating a Corpus of Student Answers

After generating the reference answer facets, we annotated each student answer to indicate whether and how the student addressed each of the corresponding facets. After analyzing much of the data in the first science module, Physics of Sound, we settled on the five tutor-level annotation labels, Tutor-Labels, noted in Table 1. See [15] for a description and discussion of all of the labels utilized. The experiments in automatic assessment discussed later are all based on this level of classification.

Table 1. Facet Annotation Labels

Understood: Facets directly expressed or whose understanding is inferred
Contradiction: Facets contradicted by negation, antonyms, pragmatics, etc.
Self-Contra: Facets that are both contradicted and implied – self contradictions
Diff-Arg: The core relation is expressed, but it has a different modifier or argument (e.g., *the pitch is loud* instead of the reference answer facet *the pitch is high*)
Unaddressed: Facets that are not addressed at all by the student's answer

3 Generating a Fine-Grain Representation of Answer Semantics

This section, which is the emphasis of this paper, describes a process for automatically generating a fine-grained representation of the semantics of the reference answer and the student's answer. Before generating this representation, the answers go through a series of linguistic preprocessing steps. First, the answers are segmented into their distinct sentences. Then the text of sentences is tokenized, breaking it into the discrete linguistic units (e.g., words, numbers, punctuation) required by the subsequent processes. Next, the tokenized sentences are processed by a part-of-speech (POS) tagger. Finally, the POS-tagged sentences are provided as features to MaltParser [17] to generate dependency parses, representing the syntactic relations between the tokens in each sentence. The goal of most English dependency parsers is to produce a tree structure, where each node in the tree represents a word in the sentence, each link represents a functional category relation, usually labeled (e.g., subject, object...), between a governor (head) and a subordinate (modifier), each node has a single governor, and the tree is projective – the links do not cross (c.f., [16]). Finally, this dependency parse tree is the basis for building the representation of answer semantics described below. Fig. 1 illustrates one answer's dependency parse.

This dependency parse was then automatically modified in several ways. The rationale for the modifications, which we demonstrate in the descriptions below, is to

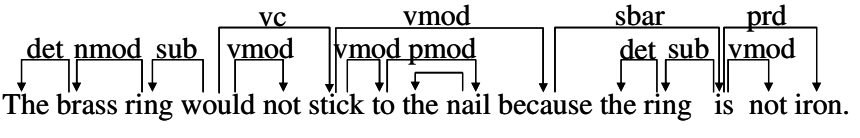


Fig. 1. Dependency parse of an example answer

increase the semantic content of the low-level components of the representation. First, we remove determiners, as they add little value to the semantics. Second, auxiliary verbs and their modifiers are reattached to the associated main verbs. In the example in Fig. 1, this involves reattaching the modal *would*, its subject *ring*, and the verb modifier *not* to the main verb *stick*, making it the new root of the dependency tree (see Fig. 2). The reason for this transformation is that modals generally carry very little semantic value. In our example, we have effectively replaced the dependency *the ring would*, Sub(*would, ring*), with the dependency *the ring sticks*, Sub(*stick, ring*), which provides far more information about the concepts involved in the answer.

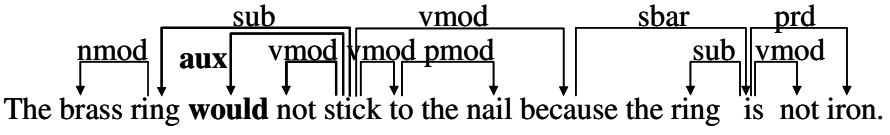


Fig. 2. Representation after removing determiners and demoting auxiliary verbs

Next, prepositions are incorporated into the dependency relation labels following [12]. In our example, this results in *nail* being reconnected to *stick* and setting its relation type to VMod_to, the conjunction of the preposition's relation type, VMod, and the preposition itself, *to* (see Fig. 3). Hence, the two dependencies VMod(*stick, to*) and PMod(*to, nail*), each of which carries little semantic value over its key lexical item, *stick* and *nail*, are combined into the single, more expressive dependency VMod_to(*stick, nail*). Likewise, the copula *is* in the subordinate clause is also reattached to *stick* and is given the relation type VMod_because.

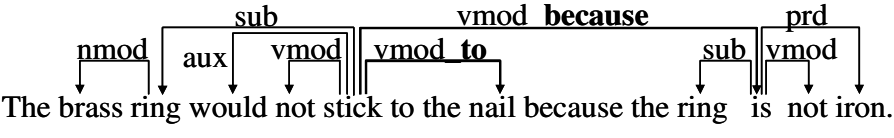


Fig. 3. Representation after incorporating prepositions into the dependency labels

Then copulas are incorporated into the dependency relations. The non-subject modifiers of the copulas are reattached to the subject and the relation type between the predicate and subject is updated to incorporate the copula. In the example, this means *iron* and the second instance of *not* are both reattached to the second instance of *ring* (see Fig. 4). The relation type connecting *iron* to *ring* is set to be_prd, reflecting its original predicate role and the incorporation of the copula *is*. This modification results in replacing the semantically impoverished dependencies Sub(*is, ring*) and Prd(*is, iron*) with the more meaningful dependency Be_Prd(*ring, iron*). We similarly integrate the verbs *do* and *have*.

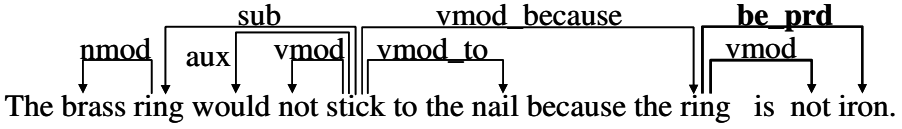


Fig. 4. Representation after incorporating copulas into the dependency relations

In an analogous modification, negation terms are appended onto the relevant dependency relations. In the example, both instances of *not* are appended to the dependency labels of each of their siblings (see Fig. 5). This provides an indication that the semantics associated with those dependencies have negative polarity.

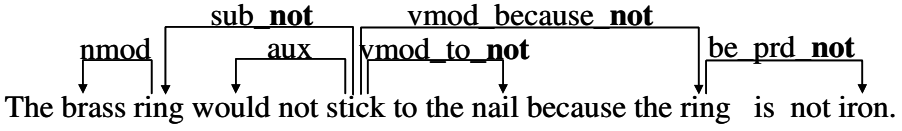


Fig. 5. Final representation after incorporating terms of negation

All of these modifications are made automatically. The intent of the revisions is to increase the likelihood that terms carrying significant semantic content are joined by dependencies that will be the focus of feature extraction. Specifically, comparing a dependency such as Sub(*would, ring*) from a reference answer to a learner's utterance provides little additional value over a purely lexical comparison involving *ring*. Whereas, the dependency Sub_not(*stick, ring*) carries far more semantic value.

This representation is generated for the question, the reference answer and the learner response. In section 0, we discussed the reference answer facet representation that is currently the basis for assessing the learner response – learners must express an understanding of those facets. The automatically generated representation of the reference answer described in this section is utilized to extract features for the machine learning classifier that are based on a representation that is more consistent with that of the learner answer. The long-term plan is to have a single automatically generated representation used for the assessment and feature extraction, but the current manual facet extraction in section 0 ensures that we are evaluating our assessment algorithms relative to a more accurate judgment regarding the significant aspects of the reference answer. A notable difference between the assessed reference answer facets and the representation generated automatically is that the latter does not utilize thematic role labels since we have not yet adapted our shallow semantic parser to appropriately handle children's utterances.

4 Automatic Assessment of Learner Responses

A high level description of the system classification procedure follows. We start with hand generated reference answer facets. We generate automatic parses for the reference answers and the student answers and automatically modify these parses per our

desired representation. From this data, we automatically extract features indicative of the student's understanding for each reference answer facet. Finally, we train a machine learning classifier on our training data and use it to classify the unseen examples in the test set. The system performs a separate classification based on the Table 1 labels to indicate the student's understanding of each facet of the associated reference answer.

Table 2. Machine Learning Feature Set

Lexical Features **Gov/Mod_MLE:** The lexical entailment probabilities for the reference answer facet governor (Gov) and modifier (Mod) following [5] [24]. **Gov/Mod_Match:** True if the Gov (Mod) stem has an exact match in the learner answer. **Subordinate_MLEs:** The lexical entailment probabilities for the primary constituent facets' Govs and Mods when the facet represents a relation between higher-level propositions.
Syntactic Features **Gov/Mod_POS:** Part of speech tags for the facet Gov & Mod **Facet/AlignedDep_Reltn:** The labels of the facet and aligned learner answer dependency – alignments were based on co-occurrence MLEs as with words, i.e., they estimate the likelihood of seeing the reference answer dependency in a document given it contains the learner answer dependency. **Dep_Path_Edit_Dist:** The edit distance between the dependency path connecting the facet's Gov and Mod (not necessarily a single step due to parser errors) and the path connecting the aligned terms in the learner answer. Paths include the dependency relations generated in our modified parse with their attached prepositions, negations, etc, the direction of each dependency, and the POS tags of the terms on the path. The calculation applied heuristics to judge the similarity of each part of the path (e.g., dropping a subject had a much higher cost than dropping an adjective). Alignment for this feature was made based on which set of terms in an *N*-best list (*N*=5 in the present experiments) for the Gov and Mod resulted in the smallest edit distance. The *N*-best list was generated based on the lexical entailment probabilities (see Gov/Mod_MLE).
Other Features **Consistent_Negation:** True if the facet and aligned student dependency path had the same number of negations. **RA_CW_cnt:** The number of content words in the reference answer.

4.1 Features

Many of the features utilized by the machine learning algorithm here are based on document co-occurrence counts. We used three publicly available corpora (English Gigaword, The Reuters corpus, and Tipster) totaling 7.4M articles and 2.6B terms. These corpora are all drawn from the news domain, making them less than ideal for assessing student's answers to science questions. The corpora were indexed and searched using Lucene, a publicly available Information Retrieval tool.

We investigated a variety of linguistic features and chose to utilize the features summarized in Table 2, informed by training set cross validation results from a decision table [9]. The features assess the facets' lexical similarity via lexical entailment probabilities following [5], part of speech (POS) tags, and lexical stem matches. They include information extracted from the modified dependency parses such as relevant relation types and path edit distances. Remaining features include information about polarity among other things (see Table 2). The revised dependency parses described earlier are used to align the terms and facet-level information for feature extraction, as indicated in the feature descriptions. Further details regarding these features can be found in [14].

4.2 Experimental Setup

The data was split into a training set and three test sets. The first test set, *Unseen Modules*, consists of *all* the data from three of the 16 science modules, providing a domain-independent test set. The second, *Unseen Questions*, consists of all the student answers associated with 22 randomly selected questions from the 233 questions in the remaining 13 modules, providing a question-independent test set. The third, *Unseen Answers*, was created by randomly assigning all of the facets from approximately 6% of the remaining learner answers to a test set with the remainder comprising the training set. All of the data in the Unseen Modules test set had been adjudicated; whereas, about half of the remaining data (training data, Unseen Questions and Unseen Answers) had not. We used the most recent annotation of the unadjudicated data during the experiments presented here. In the present work, we utilize only the facets that were not assumed to be understood a priori. This selection resulted in a total of 54,967 training examples, 30,514 examples in the Unseen Modules test set, 6,699 in the Unseen Questions test set and 3,159 examples in the Unseen Answers test set.

We evaluated several machine learning algorithms and C4.5 [21] achieved the best results in cross validation on the training data. Therefore, we used it to obtain results for assessing student answers according to the new representation described here. The effect of classifier choice will be analyzed in future work.

4.3 Results

Table 3 shows the classifier's accuracy in cross validation on the training set and each of our test sets. The columns first show two simpler baselines, the accuracy of a classifier that always chooses the most frequent label in the training set – Unaddressed, and the accuracy based on a lexical decision that chooses Understood if both the governing term and the modifier are present in the learner's answer and outputs Unaddressed otherwise, (we also tried placing a threshold on the product of their lexical entailment probabilities following [5], who achieved the best results in the first Recognizing Textual Entailment challenge, but this gave virtually the same results as the word matching baseline). The column labeled Table 2 Features presents the results of our classifier. (Reduced Training is described in the Discussion section which follows.)

Table 3. Classifier Accuracy

	Majority Label	Lexical Baseline	**Table 2** Features	Reduced Training
Training Set CV	54.6	59.7	**77.1**	
Unseen Answers	51.1	56.1	**75.5**	
Unseen Questions	58.4	63.4	61.7	**66.5**
Unseen Modules	53.4	62.9	61.4	**68.8**

4.4 System Results Discussion

The accuracy achieved, assessing learner answers within this new representation framework, represent an improvement of 24.4%, 3.3%, and 8.0% over the majority class baseline for Unseen Answers, Questions, and Modules respectively. Accuracy on Unseen Answers is also 19.4% better than the lexical baseline. However, this simple baseline outperformed the classifier on the other two test sets. It seemed probable that the decision tree over fit the data due to bias in the data itself; specifically, since many of the students' answers are very similar, there are likely to be large clusters of identical feature-class pairings, which could result in classifier decisions that do not generalize as well to other questions or domains. This bias is not problematic when the test data is very similar to the training data, as is the case for our Unseen Answers test set, but would negatively affect performance on less similar data, such as Unseen Questions and Modules.

To test this hypothesis, we reduced the size of our training set to about 8,000 randomly selected examples, which would result in fewer of these dense clusters, and retrained the classifier. The result for Unseen Questions, shown in the *Reduced Training* column, was an improvement of 4.8%. Given this promising improvement, we attempted to find the optimal training set size through cross-validation on the training data. Specifically, we iterated over the science modules holding a different module out for evaluation on each iteration and training on the other 12. For each training set module, we analyzed the learning curve varying the number of randomly selected examples per facet. We estimated the optimal accuracy for training set cross-validation by averaging the results over each module. We then trained a classifier on that number (8) of random examples per facet utilizing the full training set and tested on the Unseen Modules test set. The result was an increase in accuracy of 7.4% over training on the full training set. We are currently investigating other techniques to avoid this over-fitting.

5 Discussion and Future Work

We presented a novel low-level semantic representation that is largely derivable automatically using existing natural language technology and evaluated it in the context of assessing learner responses to questions similar to what we envision in the ITS we are constructing. A significant contribution of this representation is that it will facilitate more precise tutor feedback, targeted to a specific facet of the reference answer and pertaining to the specific level of understanding expressed by the student.

The representation's validity is partially demonstrated in the ability of annotators to reliably annotate inferences at this facet level, achieving substantial agreement (86% on the labels in Table 1, Kappa= 0.72 [3]). It was further demonstrated by the promising results on automatic assessment of student answers at this facet level. The classifier results for the Unseen Answers test set are 19% over the lexical baseline and the out-of-domain results are 15% and 6% over the most frequent class and lexical baselines respectively. Additionally, this is the first work to demonstrate success in assessing roughly sentence length constructed responses from elementary school children.

Utilizing a dependency parse to improve the accuracy of assessments has been successfully implemented by others [23]. However, this is the first work that decomposes reference answers into fine-grained facets based on dependency parses and performs automatic assessment at this fine-grained level. It is also the first work to use related information to extract features that are indicative of student understanding *independent of a specific subject matter domain*.

As is, the technique presented here does not require the collection of any example student answers to handle a new question or even domain. However, the assessed reference answer facets are generated manually. This will be addressed to facilitate quick migration to new topics, and in the long term, to enable the ITS to process self-generated questions. Other key areas of future research involve improving the current system accuracy and integrating the assessment system into an ITS. One area of particular interest is evaluating the impact of the representation detailed in this paper.

The corpus of learner answers described here will be made publicly available for other researchers to utilize in improving their tutoring and educational assessment technologies. This database of annotated answers provides a shared resource and a standardized annotation scheme allowing researchers to compare work and should stimulate further research in these areas.

Prior work on intelligent tutoring systems has largely focused on question-specific assessment of answers and even then the understanding of learner responses has generally been restricted to a judgment regarding their correctness or in a small number of cases a classification that specifies which of a predefined set of misconceptions the learner might be exhibiting. The domain-independent approach described here enables systems that can easily scale up to new content and learning environments, avoiding the need for lesson planners or technologists to create extensive new rules or classifiers for each new question the system must handle. This is an obligatory first step toward the long-term goal of creating intelligent tutoring systems that can truly engage children in natural unrestricted dialog, such as is required to perform high quality student directed Socratic tutoring.

Acknowledgements. This work was partially funded by Award Number 0551723 from the National Science Foundation.

References

1. Briscoe, E., Carroll, J., Graham, J., Copestake, A.: Relational evaluation schemes. In: Proc. Beyond PARSEVAL Workshop at LREC (2002)
2. Callear, D., Jerrams-Smith, J., Soh, V.: CAA of short non-MCQ answers. In: Proc. CAA (2001)

3. Cohen, J.: A coefficient of agreement for nominal scales. Educational and Psychological Measurement 20, 37–46 (1960)
4. Gildea, D., Jurafsky, D.: Automatic labeling of semantic roles. CL 28(3), 245–288 (2002)
5. Glickman, O., Dagan, I., Koppel, M.: Web Based Probabilistic Textual Entailment. In: Proc. PASCAL RTE Workshop (2005)
6. Graesser, A.C., Hu, X., Susarla, S., Harter, D., Person, N.K., Louwerse, M., Olde, B.: AutoTutor: An Intelligent Tutor and Conversational Tutoring Scaffold. In: Proc. AIED (2001)
7. Jordan, P.W., Makatchev, M., VanLehn, K.: Combining competing language understanding approaches in an intelligent tutoring system. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220. Springer, Heidelberg (2004)
8. Kipper, K., Dang, H.T., Palmer, M.: Class-Based Construction of a Verb Lexicon. In: Proc. AAAI (2000)
9. Kohavi, R.: The power of decision tables. In: Lavrač, N., Wrobel, S. (eds.) ECML 1995. LNCS, vol. 912. Springer, Heidelberg (1995)
10. Lawrence Hall of Science: Assessing Science Knowledge (ASK), UC Berkeley, NSF-0242510 (2006)
11. Leacock, C.: Scoring free-response automatically: A case study of a large-scale Assessment. Examens 1(3) (2004)
12. Lin, D., Pantel, P.: Discovery of inference rules for Question Answering. Natural Language Engineering 7(4), 343–360 (2001)
13. Mitchell, T., Russell, T., Broomhead, P., Aldridge, N.: Towards Robust Computerized Marking of Free-Text Responses. In: Proc. CAA (2002)
14. Nielsen, R.D.: Learner Answer Assessment in Intelligent Tutoring Systems. Ph.D. thesis, Dept. of Computer Science and Institute of Cognitive Science, CU Boulder (2008)
15. Nielsen, R.D., Ward, W., Martin, J., Palmer, M.: Annotating Students' Understanding of Science Concepts. In: Proc. Language Resources and Evaluation Conference (2008)
16. Nivre, J., Scholz, M.: Deterministic Dependency Parsing of English Text. In: Proc. COLING (2004)
17. Nivre, J., Hall, J., Nilsson, J., Eryigit, G., Marinov, S.: Labeled Pseudo-Projective Dependency Parsing with Support Vector Machines. In: Proc. CoNLL (2006)
18. Palmer, M., Gildea, D., Kingsbury, P.: The proposition bank: An annotated corpus of semantic roles. CL (2005)
19. Peters, S., Bratt, E.O., Clark, B., Pon-Barry, H., Schultz, K.: Intelligent Systems for Training Damage Control Assistants. In: Proc. Interservice/Industry Training, Simulation, and Education Conference (2004)
20. Pulman, S.G., Sukkarieh, J.Z.: Automatic Short Answer Marking. In: Proc. Workshop on Building Educational Applications Using NLP, ACL (2005)
21. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
22. Roll, I., Baker, R.S., Aleven, V., McLaren, B.M., Koedinger, K.R.: Modeling Students' Metacognitive Errors in Two Intelligent Tutoring Systems. In: Proc. User Modeling (2005)
23. Rosé, C.P., Roque, A., Bhembe, D., VanLehn, K.: A hybrid text classification approach for analysis of student essays. In: Dignum, F.P.M. (ed.) ACL 2003. LNCS (LNAI), vol. 2922. Springer, Heidelberg (2004)
24. Turney, P.D.: Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, pp. 491–502. Springer, Heidelberg (2001)
25. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The Andes physics tutoring system: Five years of evaluations. In: Proc. AIED (2005)

Automatic Construction of a Bug Library for Object-Oriented Novice Java Programmer Errors

Merlin Suarez and Raymund Sison

College of Computer Studies, De La Salle University – Manila
2401 Taft Avenue, Manila 1004 Philippines
{cruzm_a,sisonr}@dlsu.edu.ph

Abstract. Machine learning techniques have been applied to the task of student modeling, more so in building tutors for acquiring programming skill. These were developed for various languages (Pascal, Prolog, Lisp, C++) and programming paradigms (procedural and declarative) but never for object-oriented programming in Java. JavaBugs builds a bug library automatically using discrepancies between a student and correct program. While other works analyze code snippets or UML diagrams to infer student knowledge of object-oriented design and programming, JavaBugs examines a complete Java program and identifies the most similar correct program to the student's solution among a collection of correct solutions and builds trees of misconceptions using similarity measures and background knowledge. Experiments show that JavaBugs can detect the most similar correct program 97% of the time, and discover and detect 61.4% of student misconceptions identified by the expert.

Keywords: automatic bug library construction, Java errors, object-oriented programming, multistrategy learning.

1 Introduction

Artificial Intelligence in Education (AIED) as an area of research has matured through the years. It has produced significant, realistic approaches to solving issues in the development of intelligent software for educational support. Of the many topics under the area, it appears that the complexity of the student modeling problem has left researchers disinterested [10]. However, advancements in artificial intelligence techniques, particularly machine learning have led to the development of self-improving student models ([1], [3], [4], [11]), i.e., a student model that can automatically update its knowledge based on observed student behavior.

There are several approaches to building a student model. The focus of this work is the construction of a bug library for novice Java programmer errors. A *bug library* is a collection of commonly occurring errors and misconceptions. The main issues in bug library construction are *complexity* and *cost*. ASSERT [3] and MEDD [11] resolved these concerns by building self-improving bug libraries. These used machine learning approaches to automatically build and extend the bug library.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 184–193, 2008.

In the age of object-oriented programming, it is noted that only few ITSs (thus, student modelers) have been developed. This may be due to the fact that the complex process of object-oriented programming compounds the already difficult problem of student modeling for programming. While studies of Java programming errors exist ([5], [6], [12]), these focus mainly on commonly occurring syntax errors and compilation behavior, not necessarily on learning object-oriented concepts and programming.

There is a need to move beyond analyzing student syntax errors in studying object-oriented programming. To achieve this, programming solutions must be analyzed based on the student's *intention* [7]. An *intention* is the student's attempted strategy to solve any programming problem. Intention-based diagnosis also singles out one correct solution to compare the student's solution to. This is important in evaluating programming solutions because there are many correct ones. This approach is employed in ([11] and [9]). The latter, in particular, evaluates student solutions to object-oriented programming in Eiffel. It extracts differences between an intention and the student's solution. The human expert then interprets the meaning of these differences.

This paper presents JavaBugs. It constructs a bug library of Java novice programmer errors in object-oriented programming *automatically* using the multistrategy approach used in MEDD. By automatic we mean that it does not require the aid of a human expert to organize the bug library. It is organized as follows: Section 2 presents its input and output, followed by a discussion of the algorithms for intention-detection and discrepancy extraction, and misconception detection and discovery in Section 3. Test results and its analysis are presented in Section 4. The paper ends with a presentation of the conclusion and future direction of the work.

2 JavaBugs

Student modeling is the process of approximating student's knowledge of a lesson in an intelligent tutoring system. To "approximate" requires understanding of student's behavior in the software environment. It takes a student's final solution to a programming exercise and infers the student's knowledge by comparing it to its library of common errors and misconceptions (bug library).

The task of automatic bug library construction entails detecting the most similar correct program (*intention* expressed as *reference programs*), extracting the superficial differences (*discrepancies*) between the student's and the correct program and forming misconception definitions (*error hierarchies*) described by discrepancies based on similarity and causality heuristics.

2.1 Input

Inputs to JavaBugs include a student's program (in .java format), a knowledge base composed of the bug library, a set of correct programs called reference programs, and causality heuristics.

The bug library contains common errors students committed while learning object-oriented programming in Java. *Causality heuristics* are domain knowledge used by JavaBugs to sever nodes in the error hierarchy. For instance, discrepancies occurring

between two classes related by either inheritance or aggregation possess causal relationship, as shown in nodes A and B in Figure 2. If a child node does not have any causal relationship with its parent, then it is removed from its parent and re-clustered, causing a re-organization of the tree.

All Java programs, whether student or reference programs, are stored not as text files, but as a quadruple *(C, S, M, mE)*, extracting only the relevant features of a Java program. C, S and M are tables expressing the relevant features of a Java class, attributes and methods. mE is a set of extended control flow graphs [2] that represents various ways to define a method body correctly. A sample Java program and its equivalent representation are shown in Figure 1. The first row in the class table shows the relevant features of class Doctor: it was defined as a public class and inherits from another class, Person. It is used by two classes, class Clinic and the Driver class. The other two classes used in the solution are classes Clinic and Driver. The relevant features of the symbol table include the modifier, data type, and used-by. The relevant features for the methods table include its modifier, return type and arguments.

Class table, C

Id	modifier	extends	class	implements	interface	used by
Doctor	public	yes	Person	null	null	Clinic, Driver
Clinic	public	none	null	null	null	Driver
Driver	public	none	null	null	null	null

Modified ECFG

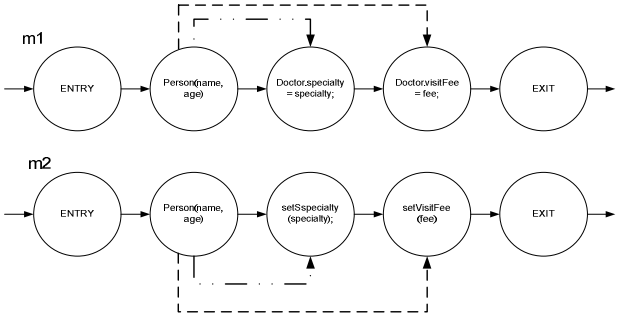


Fig. 1. Sample representation of a Java program

mE is the modified ECFG. It represents not only control flow dependencies as in [2], but data flow dependencies, too. Each node represents a Java statement in a method body, and the straight lines express control flow dependencies, while dashed lines express control flow dependencies. Multiple graphs in mE express the multiple correct solutions to a method body.

2.2 Output

The output of JavaBugs are the student's intention and his misconception(s) i.e., which cluster(s) it was added to. In the process of clustering, the bug library is updated. If the

misconceptions exhibited by a student solution were previously seen, it is classified in the error hierarchy without any structural modifications. In this case we say that the misconception was *classified*. In case a novel misconception is seen, then the error hierarchy will be updated, the hierarchy may grow in depth and/or breadth. In this case we say that a misconception has been *discovered*.

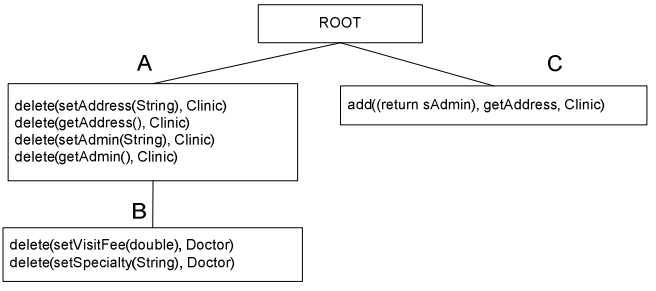


Fig. 2. Sample Error Hierarchy for a programming exercise

Figure 2 presents a sample error hierarchy. There is one error hierarchy for each reference program associated with a programming exercise. Therefore, there are *n* error hierarchies for *n* reference programs. The path from the root to node B describes a misconception on the concept of abstraction and information hiding, i.e. the student failed to define getters and setters for Clinic and Doctor class' setters because possibly he accesses these class members directly. Node C, on the other hand, reflects a *slip*, i.e. an error that may be due to fatigue, exhaustion or carelessness. The student possibly copied and pasted getters and failed to change attribute sAdmin to attribute sAddress as the value to be returned.

3 Automatic Bug Library Construction

The task of automatic bug library construction has two phases: (i) intention detection and discrepancy extraction, and (ii) misconception discovery and generalization. The first phase identifies the reference program and extracts differences between them. JavaBugs uses beam search to compare the student's solution to a set of reference programs[1] by comparing an entry in the class table one at a time. Tversky's Ratio Model ([13]) is used to compare class, symbol and method tables. After a correct match in a row is found, JavaBugs proceeds to compare the symbol table and methods table of the matched classes.

After the intention has been identified, the differences between the two programs are extracted and expressed in terms of first-order logic with the following predicates: *add*, *delete* and *replace*. The *add* predicate means that a Java program component was added, for instance, an attribute was added by the student, while the delete predicate

[1] A programming exercise always has more than one correct solution, thus there is a need to search for the student's intention.

expresses the student's removal of a Java program component, i.e. deleting an attribute. The *replace* predicate is used when a program component was used in place of another, i.e. instead of integer a double data type was used. Note that the replace predicate is used sparingly. In Figure 2, node B has two discrepancies: delete(setVisitFee(double), Doctor) and delete(setSpecialty(String), Doctor) This means that the setters setVisitFee with parameter of data type double and setSpecialty with parameter of data type String were not defined in class Doctor.

Phase 2 requires that misconception definitions be learned based on the objects it sees. In this case, an object is the discrepancy set from Phase 1. JavaBugs used MEDD's approach to form meaningful partitions over observed objects, thus creating an error hierarchy. It uses Tversky's Contrast Model ([13]) to measure similarity between two objects.

Suppose an object with the following discrepancies will be added to the tree in Figure 2: {delete(setAddress(String),Clinic), delete(getAddress(String),Clinic), delete(setAdmin (String), Clinic), delete(getAdmin (String), Clinic)), add(getName(), Doctor), add(setName(), Doctor)}. JavaBugs uses the Tversky formula to decide which between nodes A and C are most similar to the object to be added. It decides that A is the most similar, therefore the left subtree is traversed. At the next level, it decides whether the unmatched discrepancies {add(getName(), Doctor), add(setName(), Doctor)} match node B. Because it does not, a new node is created as a child of node A (node D). The resulting tree is shown in Figure 3.

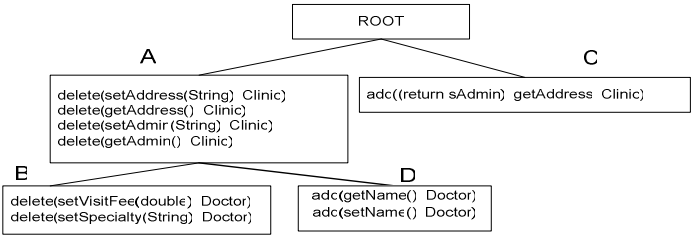


Fig. 3. Inserting a new object in the tree in Figure 2 results to a new node D created as a child of node A

JavaBugs severs a node from its parent, effectively re-classifying an object that is incorrectly classified. This is important because a student solution may possibly contain more than one unrelated misconception, and should therefore be found in a different location in the error hierarchy.

After node D is created as a child of node A, JavaBugs checks if D has a causal relationship with node A. There is no relationship between them as indicated by the discrepancies in A and D. The discrepancies in node D relate to adding a getter and setter for the attribute name, while discrepancies in node A relate to deleting getters and setters for attributes in Clinic. Because of this, node D is removed as a child of A and is re-classified into the tree from the root a second time. This re-classification compares node D with nodes A and C. Because they are not similar, node D becomes a child of the root. The final tree is shown in Figure 4.

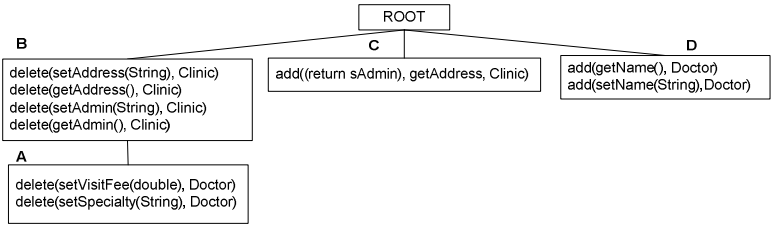


Fig. 4. Final tree produced by JavaBugs after inserting a new object into the tree in Figure 2

4 Test Results and Analysis

To test JavaBugs, a collection of student solutions in Java was collected from students enrolled in an introductory course on object-oriented programming in Java. While there were 149 students enrolled in 4 sections, only 111 students submitted their solutions to the exercises. Two exercises were given to the students, the Person-Employee and the Person-Doctor-Clinic exercises[2]. These students answered the exercises individually and submitted their solutions in electronic form. All solutions were assumed to be syntactically correct.

4.1 Intention Detection and Discrepancy Extraction

Table 1 presents the weights used by the Tversky Ratio Model for phase 1. These values were determined via experimentation.

Table 1 presents the accuracy of JavaBugs in identifying the student's intention for exercise 1 and 2. For exercise 1 (simple exercise), JavaBugs was able to correctly identify the student's intention all of the time (i.e., 100%), while it performed 94.11% for the more difficult exercise, exercise 2. It made mistakes on 5 programs because of the discrepancy language used.

Figure 5 presents a student program A which is a sample solution for Exercise 2. It shows a class definition for class Clinic. It has 4 attributes, sAddress, sAdmin, doctors

Table 1. JavaBugs' accuracy score in identifying the student's intention

Exercise 1	Frequency	Accuracy (%)
Correct	101	100
Incorrect	0	0
Exercise 2	Frequency	Accuracy (%)
Correct	80	94.11
Incorrect	5	5.88

[2] For both exercises, the Person class is given. Person-Employee tests the inheritance concept, i.e., requires that the student inherit class Doctor from class Person. Person-Doctor-Clinic tests inheritance and aggregation, i.e. Doctor should inherit from Person and there is a list of Doctors in class Clinic.

and nIndex. When JavaBugs compared it to reference program 1[3], four (4) discrepancies were found: {delete(private, doctors, Clinic), add(public, doctors, Clinic), delete(private, nIndex, Clinic), add(public, nIndex, Clinic)}. Comparing it to reference program 2, however, yielded only three (3) discrepancies: {delete(private, doctors, Clinic), add(public, doctors, Clinic), delete(nIndex, Clinic)}. While student program A should be matched to reference program 1, this did not happen because JavaBugs chose the most similar reference program, i.e. that with the least number of discrepancies ([7], [11]). In this case, it is reference program 2. The same is true for the rest of the programs it failed to classify (6%).

```
public class Clinic {
     private String sAddress;
     private String sAdmin;
     public Doctor[] doctors;
```

Fig. 5. Student Program A's attributes for class Clinic shows public was used instead of the private modifier

Because add and delete predicates were used to express differences in modifiers, a replaced modifier automatically counted for two (2) discrepancies while a removed attribute was represented using the delete predicate, yielding only one (1) discrepancy.

4.2 Misconception Discovery and Generalization

Aside from being evaluated on its capability to detect student intention, JavaBugs was also evaluated based on its capability to accurately discover classes of misconceptions based on discrepancies between a reference program and a student's solution as determined by human experts as in [11].

The Tversky Contrast Model required the use of three weights: α, β, ω. The values of the weights are as follows: $\alpha = 1$, $\beta = 0.1$, $\omega = 0.1$. α is the weight of the commonalities between the object and the node. β and ω are the weights assigned to the number of differences between the object to be inserted and the node. While other values for the weights were tried, these weights yielded meaningful clusters. Different values for the threshold were also tested.

To compute for the accuracy score, a discovered misconception is given a score of 1, and a partial discovery yields a score based on the number of discrepancies found/total number of discrepancies ([11]).

Table 2. JavaBugs' percentage accuracy in detecting misconceptions for exercise 1 and 2

Exercise 1	NO SEVER (t = -6)	SEVER (t = -6)	% Increase
LEH	58.00	65.00	7.00
Exercise 2	NO SEVER (t = -6)	SEVER (t = -6)	% Increase
LEH	43.66	57.80	14.14

[3] Reference program 1 has 4 attributes, namely the address, admin, list of doctors and index, while reference program 2 has 3 attributes, namely the address, admin and the list of doctors.

This test is performed using the best threshold (-6) resulting from experimentation. Table 3 shows the performance of JavaBugs in discovering and detecting misconceptions of Person-Employee (Exercise 1) and Person-Doctor-Clinic Exercises (Exercise 2). For Exercise 1, the results are as follows: using the tree created without sever to classify errors resulted in an accuracy score of 58%. When sever was used, the tree was able to classify 65% of objects, an increase of 7%.

The results for Exercise 2 are more telling. An increase of 14.14% was earned by JavaBugs from a local error hierarchy with no sever to an LEH with sever (43.66% vs. 57.80%).

From these results, it appears that there is a small increase in performance for Exercise 1. This is brought about by the fact that a majority of discrepancies in the Person-Employee exercise are all related because only one class was defined, i.e. class Employee. This means that the only errors a student may commit is on this class, or in the driver class that manipulates this class. This means that there is less need to disambiguate errors because there is a good chance that errors are related.

The contrary is true for Exercise 2. Because two classes are defined and a complex data type was used for one of the attributes (array data type for list of Doctors), there are more errors to untangle. This means that sever is needed to build better partitions in this case. This is supported by the increase in performance when sever was used, where the accuracy score increased by 14.14%, almost doubling the increase for Exercise 1 at 7%.

Here we observe that more complex discrepancy sets rely heavily on the sever operator. Essentially, the sever operator allows JavaBugs to use knowledge in forming its partitions. The sever operator allows re-classification of objects. This means that it has the capability to divide an object into several sub-objects based on relationship of discrepancies that describe it.

This test illustrates how the sever operator was able to (a) form meaningful partitions by removing the parent-child relationship between nodes if these are not causally related, (b) allow multi-classification of an object into the tree, (c) disambiguate common, co-occurring unrelated misconceptions of an object [8].

5 Conclusion and Future Directions

The objective of this research project is to automatically build a bug library for learning object-oriented programming in Java so as to provide software tutors the capability to (a) model student knowledge, and (b) remediate effectively given (a). The approach should be automatic because of the inherent difficulty of having to build a bug library manually.

JavaBugs performs intention detection at two-levels to determine the most similar program. It detects the intention at the design-level, matching classes, attributes and methods. Then it performs intention detection again at the *mE* level, determining the most similar approach to implementing the methods. Beam search was used to limit the number of candidate reference programs. Comparison was done across all classes, attributes and methods, rather than focusing on the analysis of a single class' attributes and methods. JavaBugs was able to correctly detect the intention of 97% of student programs. Its performance was affected by the discrepancy language that was

used. Essentially, it was affected by the fact that add and remove predicates were oftentimes used in place of the replace predicate.

JavaBugs built error hierarchies using both data and knowledge to ensure the quality of the error hierarchies. On the average, it was able to discover and detect 61.4% of all misconceptions identified by the expert. The sever operator complemented to improve the performance of JavaBugs.

This approach to Java program analysis and bug library construction, while promising, needs further study. For instance, the language representation for the discrepancies should be improved to provide more predicates, such as replace. In the current language, the replace predicate is used sparingly. In general, it is mimicked by using delete and add predicates used together. Because the basis for deciding on the most similar program is based on the number of discrepancies, using replace predicates will yield a higher accuracy rate in identifying similar programs.

Method body statements, instead of analyzing line per line, can be broken down to tokens and analyzed at a more detailed level. This will allow more accurate discrepancies at the mEFG level.

Acknowledgement

The authors wish to thank the Department of Science and Technology – Philippine Council of Advanced Science and Technology Research and Development (DOST-PCASTRD) for the generous funding it provided to this project.

References

1. Amershi, S., Conati, C.: Unsupervised and Supervised Machine Learning in User Modeling for Intelligent Learning Environments. In: 12th International Conference on Intelligent User Interfaces, pp. 72–81. ACM Press, New York (2007)
2. Apiwattanapong, T., Orso, A., Harrold, M.J.: JDiff: A Differencing Technique and Tool for Object-Oriented Programs. Automated Software Engineering 14(1), 3–36 (2007)
3. Baffes, P., Mooney, R.: Refinement-based Student Modeling and Automated Bug Library Construction. Journal of Artificial Intelligence in Education 7(1), 75–116 (1996)
4. Esposito, F., Licchelli, O., Semeraro, G.: Discovering student models in e-learning systems. In: Ulbrich, A., Pacnik, H. (eds.); Journal of Universal Computer Science (J.UCS) – Special Issue: Human Issues in Implementing eLearning Technology, 10 (1), pp. 47–57, Springer (2004)
5. Jackson, J., Cobb, M., Carver, C.: Identifying Top Java Errors for Novice Programmers. In: 35th ASEE/IEEE Frontiers in Education Conference. IEEE, New Jersey (2005)
6. Jadud, M.: A First Look at Novice Compilation using BlueJ. Computer Science Education 15(1), 25–40 (2005)
7. Johnson, W.L.: Understanding and Debugging Novice Programs. Artificial Intelligence 42(1), 51–97 (1990)
8. Lebowitz, M.: Experiments with incremental concept formation: UNIMEM. Machine Learning 2(2), 103–138 (1987)

9. Ng Cheong Vee, M.H., Meyer, B., Mannock, K.L.: Understanding novice errors and error paths in object-oriented programming through log analysis. Intelligent Tutoring Systems (2006)
10. Self, J.: Bypassing the intractable problem of student modelling. In: Gauthier, G., Frasson, C. (eds.) ITS 1988, Ablex, New Jersey, pp. 18–24 (1988)
11. Sison, R., Numao, M., Shimura, M.: Multistrategy Discovery and Detection of Novice Programmer Errors. Machine Learning - Special Issue on Multistrategy Learning 38(1), 157–180 (2000)
12. Sykes, E., Franek, F.: A Prototype for an Intelligent Tutoring System for Students Learning to Program in Java. Advanced Technology For Learning 1, 35–43 (2004)
13. Tversky, A.: Features of Similarity. Psychological Review 84, 327–352 (1977)

Helping Teachers Build ITS with Domain Schema

Brent Martin and Antonija Mitrovic

Intelligent Computer Tutoring Group
University of Canterbury, Christchurch New Zealand
{brent.martin,tanja.mitrovic}@canterbury.ac.nz

Abstract. Authoring ITS domain models is a difficult task requiring many skills. Tools such as ASPIRE that model domains using ontology reduce the problem by allowing the author to work at a higher level of abstraction (and thus avoid low-level code writing), but such tools tend to be complex and the task is not intuitive for many people. To overcome this problem we have developed a framework for domain schema: high-level abstractions that describe the semantics of the domain model for a class of domains. Using domain schema reduces the authoring effort to one of describing only those aspects that are unique to this particular domain; the schema provides the rest of the model. We describe the framework we have implemented and give some examples of domain types for which schema have been built.

1 Introduction

Intelligent Tutoring Systems increasingly show promise as a technology that will expand the horizons of education from those able to attend a bricks-and-mortar institution to anyone with an Internet connection. Acting as an enhancement to traditional distance learning offerings, they promise to augment laboratories and tutorials by allowing students to practice the skills they are learning from home. In recent years tutors such as the Geometry and Algebra tutors, and the Addison-Wesley database place suite (SQL-Tutor, ER-Tutor and NORMIT) have made it out of the lab and into the classroom [1], [2].

Constraint-Based Modeling (CBM) [3] is an effective approach for building Intelligent Tutoring Systems (ITS) that supports the building of domain and student models. Constraint-based tutors are effective: for example, students using our database design tutor have shown significant gains in learning after as little as one hour of exposure to this system [4]. Also, CBM seeks to minimize the authoring effort by requiring the author model only states, rather than solution paths [5]. Nevertheless, the task of building an ITS is still large. To reduce the authoring effort we have developed a number of tools, including WETAS [6] and ASPIRE, an authoring system that allows the complete development of an ITS without ever writing a line of code [7].

ASPIRE makes it feasible for teachers with no prior knowledge to develop ITS, by automating most tasks or providing GUI interfaces allowing the author to easily define the elements of the final system, such as the general domain parameters (procedural versus non-procedural etc) and the structure of solutions that will be submitted by the student. The most complex part of the authoring task, that of modeling the

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 194–203, 2008.

domain, is achieved by creating an ontology of the domain concepts using a custom graphical tool. ASPIRE was developed to support constraint-based modeling. The domain model used by the final system (i.e. the set of constraints) is generated automatically from the ontology. ASPIRE's approach dramatically reduces the effort required to build an ITS, but nonetheless it is still a formidable task. In particular, developing domain ontology is a process that does not come naturally to all authors. For example, from a group of 12 students at the 2006 e-learning summer school at the University of Dublin, only half produced usable domain models for a simple hypothetical search engine language, of which only one was completely correct; the other half had considerable difficulty grasping the complexity of the modeling task, while nearly all participants were unable to model the recursive nature of the domain [8]. Also some authors have developed domains in ASPIRE entirely independently, but others have required help. This is a common problem in ITS authoring: the more general the tool, the harder it is to use. Many authoring systems overcome this problem by being limited to a particular type of domain. For example, Demonstr8 [9] is tailored for arithmetic.

Our goal for ASPIRE is that it be a tool for authoring ITS for *any* domain. To do this it must be easily extensible. Since different authors will have differing semantic requirements it must be possible for new domain types to be supported without changes to the core ASPIRE system. To facilitate this we have developed an additional abstraction layer, *domain schema*. Domain schema define the behavior of ASPIRE for a subset of domains that share a common structure and task type. New schema can be added to ASPIRE at any time by creating the appropriate XML documents and uploading them. The schema automates the authoring process still further by performing those tasks that are consistent across all domains of this type, such as providing the main structure of the domain ontology. Authors then work with the appropriate schema, rather than ASPIRE directly.

The next section briefly introduces constraint-based modeling (CBM), and describes two CBM authoring systems. Section three outlines how domain schema work, with an implemented example in the area of critiquing images. In section four we show how the approach can be generalized to other domain types. We conclude in Section five and discuss our long-term goals; to created distributed ITS via the semantic web, and to disconnect the domain model from the modeling and reasoning approaches.

2 Constraint-Based Modeling, WETAS and ASPIRE

CBM is based on the theory of learning from performance errors [10]. It models the domain as a set of state constraints, where each constraint represents a declarative concept that must be learned and internalized before the student can achieve mastery. Constraints represent restrictions on solution *states,* and take the form:

If <*relevance condition*> *is true for the student's solution,*
THEN <*satisfaction condition*> *must also be true*

The relevance condition of each constraint checks whether the student's solution is in a pedagogically significant state. If so, the satisfaction condition is checked. If it

succeeds, no action is taken; otherwise the student has made a mistake and appropriate feedback is given. *Syntactic* constraints check that the solution is syntactically correct. Conversely, *semantic* constraints check whether the student's solution has solved the problem, usually by comparing it to an "ideal" solution supplied by the teacher. The constraints implicitly encode semantics by testing for all of the different possible encodings of the semantic concept they are attempting to test. The student is thus permitted to use a different problem-solving strategy to the author, or even to mix strategies, provided no fundamental domain concepts are violated.

WETAS is a constraint-based web-enabled tutoring engine that provides all of the domain-independent functions for text-based ITS. It is implemented as a web server, written in Allegro Common Lisp, and using the AllegroServe Web server [11]. WETAS performs as much of the implementation as possible in a generic fashion. In particular, it provides the following functions: problem selection, answer evaluation, student modeling, feedback, and the user interface. The author need only provide the domain-dependent components, namely the structure of the domain (e.g. any curriculum subsets), the domain model (in the form of constraints), the problem/solution set, the scaffolding information (if any), and possibly an input parser, if any specific preprocessing of the input is required. WETAS has been used to build several tutors, including EER-Tutor [2] and Collect-UML [12]. It has also been used for four years by a graduate University class in Intelligent Tutoring Systems at the University of Canterbury.

ASPIRE is a high-level authoring tool that automates the encoding of constraints based on an ontology that the author provides via a graphical ontology editing tool. Unlike WETAS, at all stages in ASPIRE the author interacts with a GUI tool when authoring the domain; no additional files are required. ASPIRE is a general tool that has been used to build tutors across a variety of domains, such as basic accounting, thermodynamics and solid mechanics. However, the ASPIRE approach can still be improved in at least two ways. First, the author has little control over the interface; any non-standard user-interaction must be provided via bespoke applets which are uploaded into ASPIRE. Second, authoring in ASPIRE is still far from a trivial exercise. In particular, the task of ontology authoring is specialized and difficult. We hypothesized that we could make it easier to build ontologies by providing an ontology schema that reduces the ontology vocabulary to only concepts required for a particular subset of domains, thus making the author's job much more straightforward. We combined this with the ability to specialize ASPIRE's student interface for such domains. Finally, we further hypothesized that the semantic interpretation of the ontology for all domains of a given subset would also be the same. The combination of ontology schema, interface and ontology semantics is a *domain schema*.

3 Domain Schema

A domain schema is a collection of documents that describe parts of the domain model that will be common to all domains of the same general type, such as critiquing a set of images. The documents tell ASPIRE how to perform many parts of the authoring process that would be otherwise performed manually. The documents are:

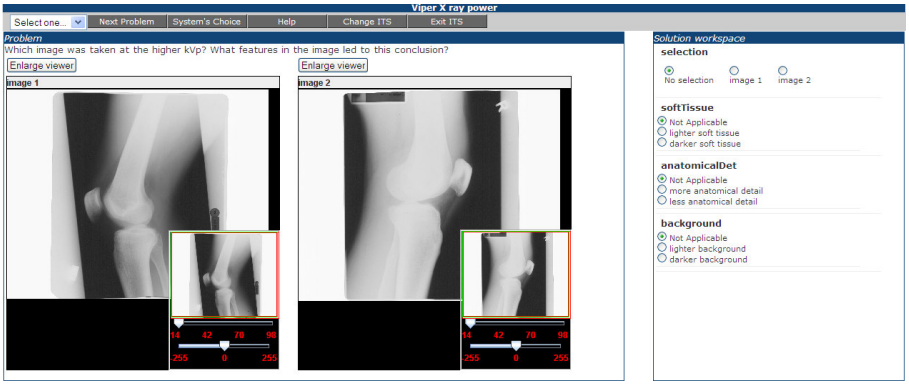


Fig. 1. Example tutor for critiquing x-ray images

- Ontology schema (XML) and ontology generation rules (XSLT)
- Constraint generation rules (XSLT)
- Solution structure generation rules (XSLT)
- Student interface (HTML, with optional Java applets)

In the following sections we will use an example domain type to illustrate how domain schemas work: for this domain type the student is shown a set of two or more images and is asked to choose the one with a particular characteristic and to identify features in the image that support their choice. This domain type could apply to many different subject areas (domains), such as: which of two buildings is Ionian; which x-ray image is better quality; which forest is the most damaged by acid rain; which painting is by Van Gogh; which x-ray shows an intestinal stricture. The interface consists of an applet for displaying, panning and zooming images, a control for selecting one of the images and a list of *features* that may or may not contribute to the decision; for each feature the student will select an appropriate *feature value*. Figure 1 shows this interface in action for an example of this domain type: x-ray power.

For each domain type the ontology will have the same basic form. The *ontology schema* defines this form by specifying concepts common to all domains of this type (typically the top of the ontology hierarchy), and describing the types of other concepts that the author can create and the relationship between these and the common concepts. Figure 2 shows part of the ontology for an ITS of the domain type "critique images", in this case the x-ray power domain, viewed using ASPIRE's ontology editor. All ontologies for this domain type contain the "feature", "image" and "selection" concepts. The "feature" concept is then specialized for the actual features that the student will look for in this domain. The author can also specify *abstract* features if they wish; these are used for adding information that is common to more than one of the actual features. In figure 2 the actual features are "anatomical detail", "background" and "soft tissue"; abstract features are "contrast technique" and "brightness technique". Each feature is then further specialized into *feature values*, which are the values the student can choose between, such as "more anatomical detail" and "lighter soft tissue". The "image" concept is used to describe the images being shown to the

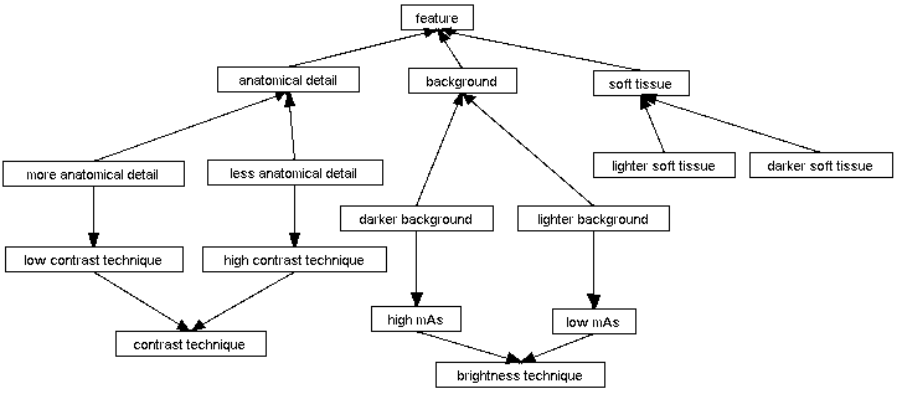


Fig. 2. Ontology for x-ray power

student, in terms of the features present in this image (whether or not they contribute to the correct answer). Finally, the "selection" concept represents the choice the student must make between images.

The ontology schema is shown in Figure 3. The second part of this ontology schema describes the two concept types the author can create (feature and feature value). For each it also describes the attributes of that concept the author will be required to provide; in this case a *feature* can have two feedback messages (one—hint—to use when the student has overlooked this feature, and the other—wrong—for when the feature has been erroneously used). Similarly, a *feature value* has a summary and detailed feedback message, and another (positive) to be displayed as reinforcement when the student has correctly answered the question. Finally, the author can specify that one concept is an example of another; in figure 2 "more anatomical detail" is an example of "low contrast technique". Once the author has filled in the details for the features and feature values, the information is saved as an XML document and converted to a standard ASPIRE ontology using XSLT.

The ontology is then converted to constraints using an XML transform. This XSLT encodes the semantic interpretation of the ontology, by specifying how each concept should be turned into one or more constraints. For the domain type under discussion the constraint generation rules are as follows:

1. **Correct selection:** for each "selection" concept check the student has supplied the correct selection value
2. **All features specified:** For each feature, if a value is specified in the ideal solution, the student must also have specified a value
3. **No extraneous features:** for each feature, if the student has specified a value, the ideal solution must also specify a value
4. **Correct feature value:** If the student has specified a feature value, and one was required, is it the same as that in the ideal solution
5. **Feature value supports selection:** if the student has selected a feature value that is present in their chosen selection, check that the selection is correct.

For each constraint the hint and feedback messages (for the concept from which it is generated) are incorporated into boilerplate text to give the actual messages the user will see when the constraint is violated. For this domain type the semantics are very straightforward; other domain types are more complex (see Section 4).

The domain schema also defines how to generate from the ontology the solution structure (i.e. what the student must submit) and the default interface, again using XSLT. By default the solution structure consists of all non-abstract concepts. In the case of the domain type under discussion, each concept of type "feature" becomes a field in the student solution. The feature values are used to create the appropriate interface widget (e.g. a set of radio buttons) that the student will use to select values. The default interface displays controls for the entire solution structure. The author may then specialize the interface by specifying which parts of the solution structure are to be used for a given type of question in this domain, and they may override how it will be displayed. For example, in this domain type the author can specify that for

```
<!-- Default ontology, inserted directly -->
<baseOntology>
  <concept id='1' label='selection' name='selection' abstract='false'>
    <property  name='value' id='value' type='Any' unique='false'
     max-cardinality='1' min-cardinality='1'/>
  </concept>

  <concept label='feature' name='feature' abstract='true'></concept>

  <concept id='2' label='image' name='image' abstract='false'>
    <attribute name="input" value="author"/>
    <property  name='name' id='name' type='Any' unique='false' />
    <property  name='URL' id='URL' type='Any' unique='false' />
  </concept>
</baseOntology>

<!-- concept types that the author can create -->
<conceptType name="feature" label="Feature" input="true"
     propertyOf="image">
  <attribute name="name" label="Name" type ="text"/>
  <attribute name="abstract" label="Abstract?" type="boolean"/>
  <attribute name="negativeHint" label="Hint" type="text">
  <attribute name="negativeWrong" label="Wrong" type="text"/>
  <relationship name="exampleOf" label="Example Of"
     type="specialisation" range="feature">

  <conceptType name="featureValue" label="Feature value">
    <attribute name="name" label="Name" type ="text"/>
    <attribute name="summary" label="Summary feedback" type="text"/>
    <attribute name="detail" label="Detailed feedback" type="text"/>
    <attribute name="positive" label="Positive feedback" type="text"/>
    <relationship name="exampleOf" label = "Example of"
            type="specialisation" range="featureValue"/>
  </conceptType>
</conceptType>
```

Fig. 3. Ontology Schema for "critique images" (in XML)

certain questions only the "soft tissue" and "anatomical detail" features will be presented to the student, and that they will be represented using combo boxes. This allows the same domain model to be used for a variety of (related) tutoring tasks.

4 Other Domain Types

We are using domain schema to develop VIPER (Virtual Instructional and Practice Educational Resource) in conjunction with the Christchurch Polytechnic Institute of Technology (CPIT). For this project there are five domain types, all of which are visual: critique images; label an image; identify a feature in the image (i.e. point to it); perform measurements on an image; experiment with the parameters of an image. In all cases the domain model is feature-based, and as a result the semantics are straightforward. Another domain type we are developing is programming languages. In this type of tutor the student is given a task to perform where they must write a snippet of code in free text form. The ontology for this type of ITS describes the grammar of the language being used. For example, consider the domain of writing logical expressions. In this domain each concept represents some part of the language (e.g. "conjunct"); concept properties represent the "part-of" relationship between a concept and the language constructs that make up that concept; for example a conjunct consists of an expression, followed by "and" followed by a second expression). The constraint generation rules for checking semantics of this domain type are as follows:

1. **Concept necessary:** for each concept, if it appears at least once in the ideal solution, it must also appear in the student solution;
2. **Concept superfluous:** for each concept, if it appears at least once in the student solution, it must also appear in the ideal solution;
3. **All concept instances present:** for each instance of each concept in the ideal solution where the student solution contains at least one instance of this concept, there must exist an equivalent instance in the student solution;
4. **No concept instances superfluous:** for each instance of each concept in the student solution where the ideal solution contains at least one instance of this concept, there must exist an equivalent instance in the ideal solution;
5. **Correct components:** for each concept instance in the student solution, if all but one component is equivalent to an instance in the ideal solution, the remaining component must also be equivalent.

For the logical expressions domain the author describes each of the concepts in the same way as they would describe a grammar in BNF. However, this is not sufficient because they also need to define equivalence. For example, "dog and cat" is equivalent to "cat and dog". They do this by defining additional concepts. In the previous example, conjunction is defined twice, with one definition being the exact reverse of the other. Each concept can then specify an "is equivalent to" relationship with

another. In some cases the concept will be one that does not already appear in the grammar. For example, for logical expressions we can define de Morgan's law:

$$\overline{(A \wedge B)} \equiv \overline{A} \vee \overline{B} \tag{1}$$

We specify this law by defining both de Morgan forms and indicating they are equivalent. The constraint generation rules then use this information as follows. First, whenever a concept detected in one solution (e.g. the ideal solution) is being looked for in the other solution (i.e. the student solution), the default logic is to look for the exact same concept instance in both solutions. However, if the concept instance is an example of a concept for which an equivalent form exists, the constraint will instead check that either the same concept instance exists in the other solution *or* an equivalent concept instance exists. Second, when checking for a particular concept instance, the constraint will also check whether it *forms part of* another concept that takes part in an equivalence relationship, and the alternate form exists in the other solution. If so, the check is dropped. For example, when checking for all "and"s, if the "and" in question is part of a De Morgan form and the student used the alternate form, the check for "and" will be dropped. We are currently evaluating this domain type in the areas of logical expressions, Java and SQL. This approach is also potentially useful for natural languages, provided the domain is sufficiently constrained. We are also exploring this possibility.

Another example of a completely different domain type is arithmetic procedural domains, such as multi-column addition. These can be catered for by extending the framework described as follows. First, for such domains the properties of a concept must be able to be collections. For example, an addition problem is made up of a collection of columns; each column contains a carry, a collection of addends and a sum. Second, the author must be able to specify arithmetic value restrictions for properties. For example (again from multi-column addition):

$$sum(n) = \left[carry(n) + SUM\left(addends(n)\right)\right] MOD\ 10 \tag{2}$$

$$carry(n) = \left[carry(n+1) + SUM\left(addends(n+1)\right)\right] DIV\ 10 \tag{3}$$

Note that *n* is the column number (more generally, *n* is the instance number). SUM and DIV are built-in primitives. As well as giving the formula for the restriction, the author also specifies two associated feedback messages: one that describes what the restriction means in words (used to correct the student when they violate the restriction) and one that describes the dependencies implied by the RHS of the restriction (used to indicate why the student should not be specifying this value yet, because the restriction cannot be tested). The constraints are now generated from both the concepts in the ontology plus the restrictions, as follows:

1. **All values specified:** For each concept instance, check whether this instance has been completed, e.g. "*You have not filled in the sum for column 3.*" Note that the restrictions imply dependencies between concept instances, which also need to be checked. If the dependent concept instances are not complete yet this constraint will not be relevant.

2. **Ordering:** For each concept that is on the LHS of a restriction, if the student has supplied an instance of this concept, check that the necessary parts in the

RHS have been specified and give the "dependency" error if not, e.g. "*You cannot compute the carry for a column until you have completed the column to the right.*"

3. **Correct value:** For each concept that is on the LHS of a restriction, test its value, and give an error if wrong, e.g. "*Check your sum in column 3. The sum should add up to the sum of addends in this column, plus the carry, if any*", or "*Check the value of the carry in column 2. The carry should be 1 if the addends and carry in the next column to the right add up to 10 or more.*"

This logic is sufficiently general to apply to other arithmetic domains, such as fraction addition.

6 Conclusions and Future Work

ITS authoring is a difficult task. Whilst generic authoring tools such as ASPIRE dramatically reduce the authoring effort required, domain authoring nevertheless remains a specialized task. We have introduced a framework, called domain schema, that allows a generic ITS authoring tool to be tailored to specific domain types to ease the authoring process, but which is still general in the sense that it can be readily extended to support new domain types. We are using this framework in the VIPER project to create an authoring environment suited to domains where the student tasks involve interacting with images. We have also shown how the approach is suited to other very different domain types such as programming languages and arithmetic. VIPER will be trialed by teachers at CPIT in mid 2008.

Most (if not all) existing ITS tools are monolithic: the student logs into the ITS system, which then serves them content. This is in direct contrast with other web-based educational content delivery approaches, which are *content-oriented*. In content-oriented systems, learners seek out appropriate educational content from any source that their client software supports (e.g. SCORM). In the ITS equivalent teachers would develop exercises for their students, who complete them and submit their answer to an appropriate reasoning engine for evaluation. The reasoning engine would be a lightweight expert system that can run rules written in some standard language (e.g. ruleML [13]). Each page of content contains links to domain content in a form similar to what we have just described: the domain is represented by an ontology plus additional information indicating how the ontology should be used to generate evaluation rules. This would allow re-use of the same ontology for different types of evaluation. For example, the domain information for multi-column addition described in section four is sufficient to generate production rules for a model-tracing tutor [14]. Further, ontology could be cobbled together from existing ones, or customized by overriding some parts. The framework we have described is a step towards this because it separates the reasoning rules, ontology and reasoning engine.

Intelligent tutoring systems are a promising tool for delivering education remotely. To date a key problem has been the effort required to build such systems, even when sophisticated authoring tools are used. Domain schema is a promising step towards making ITS a realistic option for education practitioners everywhere.

References

1. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A.: Intelligent Tutoring Goes To School in the Big City. International Journal of Artificial Intelligence in Education 8, 30–43 (1997)
2. Mitrovic, A.: Large-Scale Deployment of three intelligent web-based database tutors. In: Proceedings of ITI, Cavtat, Croatia, pp. 135–140 (2006)
3. Ohlsson, S.: Constraint-Based Student Modeling. In: Greer, J., McCalla, G. (eds.) Student Modeling: The Key to Individualized Knowledge-Based Instruction, pp. 167–189. Springer, New York (1994)
4. Suraweera, P., Mitrovic, A.: An Intelligent Tutoring System for Entity Relationship Modelling. Int. J. Artificial Intelligence in Education 14, 3–4, 375–417 (2004)
5. Mitrovic, A., Koedinger, K.R., Martin, B.: A Comparative Analysis of Cognitive Tutoring and Constraint-Based Modelling. In: Brusilovsky, P., Corbett, A.T., de Rosis, F. (eds.) UM 2003. LNCS, vol. 2702, pp. 313–322. Springer, Heidelberg (2003)
6. Martin, B., Mitrovic, A.: WETAS: A Web-Based Authoring System for Constraint-Based ITS. In: Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Malaga, pp. 543–546. Springer (2002)
7. Mitrovic, A., Suraweera, P., Martin, B., Zakharov, K., Milik, N., Holland, J.: Authoring constraint-based tutors in ASPIRE. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 41–50. Springer, Heidelberg (2006)
8. Martin, B., Mitrovic, A., Suraweera, P.: Domain Modelling with ontology: a case study. In: International workshop on authoring adaptive education systems at UM 2007, Corfu, Greece, pp. 4–11 (2007)
9. Blessing, A.: Programming by Demonstration Authoring Tool for Model-Tracing Tutors. International Journal of Artificial Intelligence in Education 8, 233–261 (1997)
10. Ohlsson, S.: Learning from Performance Errors. Psychological Review 3(2), 241–262 (1996)
11. Allegroserve, `http://opensource.franz.com/aserve/`
12. Baghhaei, N., Mitrovic, A.: A Constraint-based Collaborative Environment for Learning UML Class Diagrams. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 176–186. Springer, Heidelberg (2006)
13. Boley, H., Tabet, S., Wagner, G.: Design Rationale of RuleML: A Markup Language for Semantic Web Rules. In: The first Semantic Web Working Symposium, SWWS 2001, Stanford, California USA, pp. 381–401 (2001)
14. Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive Tutors: Lessons Learned. Journal of the Learning Sciences 4(2), 167–207 (1995)

Evaluating an Authoring Tool for Model-Tracing Intelligent Tutoring Systems

Stephen B. Blessing[1] and Stephen Gilbert[2,3]

[1] University of Tampa, Department of Psychology, 401 W. Kennedy Blvd., Tampa, FL USA
[2] Clearsighted, 2321 N Loop Dr Ste 110, Ames, IA USA
[3] Iowa State University, Department of Psychology, 1620 Howe Hall., Ames, IA USA
sblessing@ut.edu, gilbert@iastate.edu

Abstract. We have been creating an authoring tool, the Cognitive Model SDK, which allows non-cognitive scientists and non-programmers to produce a cognitive model for model-tracing tutors [1, 2]. The SDK is in use by developers at Carnegie Learning to produce their commercial Cognitive Tutors for math. However, it has never been evaluated with regards to the strong claim that non-cognitive scientists and non-programmers could, without much effort, produce useful cognitive models with it. The research presented here shows that this can be done, using a task that past researchers have used [3]. The models are evaluated across several metrics to see what characteristics of either them or their creators may distinguish better models from worse models. The goal of this work is to establish a baseline for future work examining how cognitive modeling can be opened up to a wider class of people.

1 Introduction

Model-tracing tutors have shown themselves to be one of the more effective types of Intelligent Tutoring Systems (ITSs) in terms of student learning gains [4, 5, 6]. However, they are very labor intensive to create, typically requiring a highly-qualified team of people to produce the end product. The experts needed on this team include cognitive scientists, programmers, and pedagogy and content experts. Estimates of how long it takes to create such a tutor range as high as over 100 hours of development time for 1 hour of instruction [7, 8]. For these tutors to see widespread use, this ratio needs to be greatly decreased. We see two ways of doing this: provide authoring tools that are 1) not only easier to use, but 2) also do not require high levels of expertise. The work presented here describes an evaluation of such a tool, the Cognitive Model Software Development Kit (SDK). While perhaps no authoring tool will obviate the need for a team of people from different disciplines to collaborate on building an ITS, the results from our evaluation indicate that non-cognitive scientists, historically the class of people who created the cognitive model, can produce a basic cognitive model.

1.1 The Cognitive Model SDK

The Cognitive Model SDK assists in the development of the cognitive model that forms the backbone of a Cognitive Tutor, a model-tracing tutor that is based on the

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 204–215, 2008.

ACT Theory of cognition [9]. Carnegie Learning is a commercial company that produces Cognitive Tutors, primarily for topics in high school math. They have had great success in this endeavor, both commercially and in terms of student learning gains. The company currently uses the SDK to develop their Cognitive Tutors. To date six large-scale cognitive models have been built within the SDK, providing tutoring on over 5500 problems. However, no formal evaluation of this tool had been conducted until the present study.

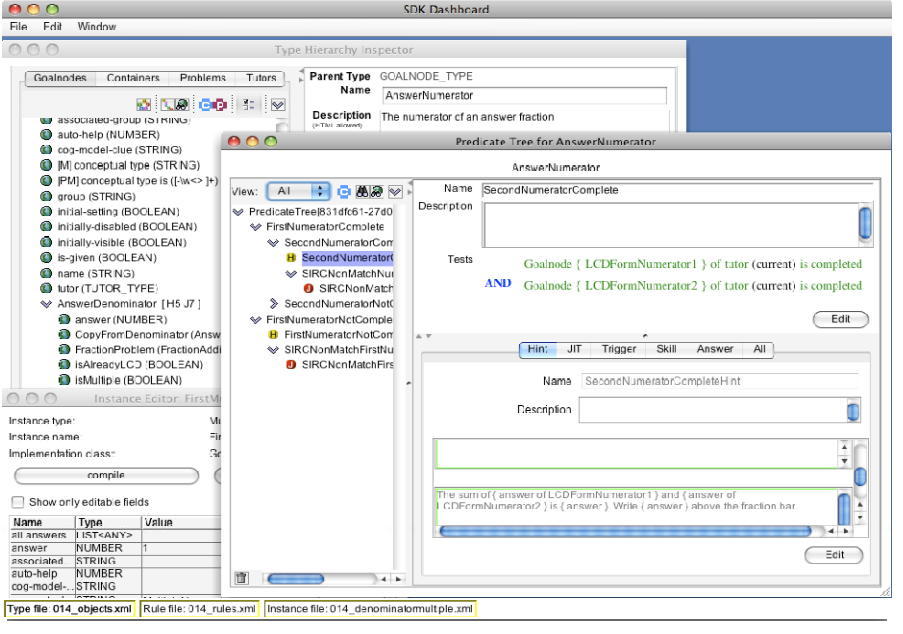


Fig. 1. Screenshot of Cognitive Model SDK

A full description of the Cognitive Model SDK is beyond the scope of the current paper, but can be found elsewhere [1, 2]. In short, there are three main pieces to the SDK and to creating a cognitive model. First, the type hierarchy used by the tutor must be defined. In ACT Theory terminology, this is the declarative knowledge of the model, containing the objects and properties that the student needs to be aware of in order to solve problems. For instance, in a domain like fraction arithmetic, the type hierarchy would contain information about what fractions are, their numerators and denominators, and other aspects of the task. "Goalnodes" are one important piece of this hierarchy. They represent the subgoals within the problem. There is typically an interface element (e.g., an entry box) that will allow the problem solver to complete a goalnode. The second SDK piece is the Rule Predicate Editor, where the rules of the task are defined (the procedural knowledge, in ACT parlance). This is where one can discriminate between different problem types in the domain (e.g., adding fractions where the denominators are already the same versus adding fraction that have different denominators). The help and just-in-time messages that are common to model-tracing tutors are stored here. The last

piece of the SDK is the instance editor, where a cognitive modeler can define problem instances. The SDK contains a simple testing interface by which the cognitive modeler can ascertain if the cognitive model is producing the correct behaviors, without attaching the cognitive model to the interface. Figure 1 contains a screenshot of the SDK, where the cognitive modeler is currently working on part of the predicate tree. Windows for the Type Hierarchy and instance editor can be seen in the background.

1.2 Past Evaluations of Authoring Systems

Evaluations of tutors built using authoring systems have been conducted in the past (e.g., [7, 10]), but evaluations of the authoring systems themselves are much more rare. Henry Halff and his colleagues conducted an early set of studies on the XAIDA authoring system [11]. While XAIDA did not produce model-tracing tutors, the studies that examined how authors used the system serve as a good model and can inform for future research in terms of the qualitative and quantitative data they collected. These studies showed very promising results, with findings like 30 hours of development time for 1 hour of instruction. Other authoring systems have also been evaluated in a similar manner (e.g., [7,12]).

We were motivated by two recent studies. First, Suraweera and his colleagues [3] performed a study looking at their Constraint Authoring System (CAS), a tool for developing a constraint-based tutor. In the context of a graduate student ITS class, 12 of 13 students were able to produce a constraint-based tutor for fraction arithmetic in an average of 31.3 hr. That almost all of the students could create an ITS, when creating ITSs may or may not have been the focus of their graduate studies, shows that the tool lowered the bar with regards to the expertise needed to create an ITS. We used the procedure outlined in this study to model the design of our own.

Second, researchers at Carnegie Mellon University are also developing an authoring tool for Cognitive Tutors. This tool, the Cognitive Tutor Authoring Tools (CTAT), approaches the task of developing a cognitive model from a different angle than ours [13]. Whereas a focus of their tool is to make it easy to create focused, more specialized tutors that center on a particular problem, what they term Example-Tracing Tutors, our authoring tool focuses on creating cognitive models that are more generalizable in nature. In a sense, our two different approaches are complementary in nature, perhaps with the long-term goal of finding that middle spot that exists between generality and ease-of-use.

1.3 This Evaluation

What follows is a description of an initial evaluation we did of the Cognitive Model SDK. We had conducted a study that examined whether or not the representations used in the SDK (e.g., the object/property view when working with types) were usable by undergraduates [1]. We found that they were. Participants in that study, however, did not create any working cognitive models. Furthermore, we have anecdotal evidence that non-cognitive scientists at Carnegie Learning and Clearsighted could create working cognitive models in a fraction of time that previous cognitive models were constructed [2]. This current study is the first controlled evaluation of the Cognitive Model SDK. Given that, it will serve mostly as a baseline for future evaluations.

Our main interest is to determine if non-cognitive scientists/non-programmers can create usable cognitive models with the tool.

2 Method

2.1 Participants

Seventeen graduate students from Iowa State University participated in this study. Six of these participants were students in the first-year HCI course at Iowa State University who chose to do this assignment for course credit. The other eleven participants were recruited from among the HCI and instructional design graduate students at Iowa State University. These students were paid $150 for their participation.

None of these participants had cognitive psychology or cognitive science as their home department, nor had any done cognitive modeling before. Some had programming experience, and this will be highlighted in the results.

2.2 Materials

An assignment similar to the one used by [3] was used. In this assignment, participants were asked to create a cognitive model, consisting of the needed goalnodes, properties, hints, and just-in-time messages, of a fraction arithmetic task. For our version of the assignment, participants were told that their model had to provide distinct and specific hints for three types of problems: 1) ones in which the two fractions started with the same denominators (e.g., 1/5 + 2/5); 2) ones where one denominator is a multiple of the other (e.g., 1/5 + 1/10); and 3) ones in which the least common denominator is neither of the two given fractions (e.g., 1/5 + 1/7).

Participants were shown a picture of an interface (see Figure 2) in which students would be given two fractions to add. The students would need to write both fractions in terms of a common denominator and would then need to compute the final, but unreduced, answer. Note that the participants did not actually have access to this interface, but had to use the more rudimentary interface that is constructed automatically by the SDK itself in its Goalnode Testing Tutor (GNTT) interface. For this particular task, the two differ only by the placement of the boxes. The GNTT provides just a linear list of the goalnodes, represented by entry boxes, defined by the cognitive model and the current problem instance.

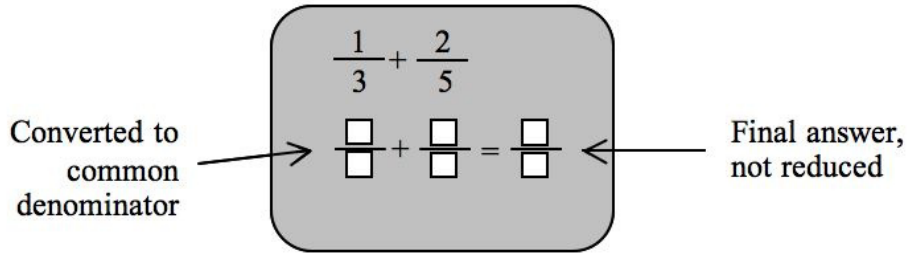


Fig. 2. Example interface shown to participants for the fraction addition task

Four pieces of background information were provided to the participants. First, some general information concerning Cognitive Tutors was given. For the students in the HCI class, this was a short in-class demonstration of the Carnegie Learning Algebra I Cognitive Tutor. For the other participants, this was a similar demo done via a screen capture movie (8.30 min in length) of the same demonstration. Second, participants were given a four-page document that introduced the vocabulary and basic concepts of cognitive modeling. It explained the difference between hints and JIT messages, introduced the concepts of hierarchical types, predicates, instances, and goalnodes

The other two pieces of information amounted to worked examples of cognitive models constructed using the SDK. The first worked example was a completed cognitive model of multi-column addition. The model could add two multi-digit numbers together, including problems involving a carry. A screen capture movie (25.97 min) was created that walked the viewer through creating all parts of the model using the SDK: the type hierarchy, the rules, and an instance. An additional problem instance was provided to participants, as well as the movie's transcript. The purpose of this information was to provide students a reasonably real-world example of a cognitive model within the SDK. While both this model and the model that the participants were asked to create both involved addition, the participants could not simply take this model, make a few simple modifications, and have a fraction addition model. The creation of the fraction addition model required the student to start from the beginning, constructing all new goalnodes, hint messages, and instances. This model served as inspiration and reference to show what is possible to do within the SDK at an appropriate level.

The last piece of information was another screen movie (10.70 min) and its transcript that stepped participants through the creation of a very simple tutor. The tutor in this movie asked students to indicate if the presented number is even or odd. This gave the participants an example of how to construct the bare minimum framework for a tutor from the start. That is, this example had one goalnode, one defined property, and one hint, plus enough glue to test the tutor within the GNTT.

The version of the SDK used by the participants was the full version of the SDK used by Carnegie Learning. In addition, it also logged how long they spent performing each action. In this way we were able to determine not only how much total time it took to complete the model, but also how long each participant spent doing the individual components of authoring a cognitive model, such as creating properties on goalnodes or writing hint messages. Participants were also given an exit questionnaire concerning their experiences, and also asked for certain demographic information such as previous programming experience.

2.3 Procedure

Participants were first given the demo of the Algebra I Cognitive Tutor. They were then given the assignment and the two worked examples as described above and asked to complete a cognitive model of fraction addition using the SDK. They could choose to do with these examples as they willed. In all, they were given about 45 min worth of instruction, and as much time as they desired to complete the assignment, though we suggested they plan between 8-12 hours for the assignment.

3 Results

The results are divided into three parts: 1) quantitative and qualitative measurements of the cognitive models; 2) timing data concerning cognitive model creation; and 3) exit questionnaire data.

3.1 Cognitive Model Measurements

Of the 17 participants, 13 of them completed a runnable cognitive model. One person's rule file became corrupted and so could not be scored, and three people did not complete the assignment, all of which were paid participants.

The first author scored the 13 completed cognitive models on a 5-point scale. The criteria used and the number of models within each criterion is in Table 1. We rated models on behavioral characteristics, not on implementational aspects. The average score is 3.31, indicating that the average model at least met expectations. The mean time to complete the assignment was 7.68 hr.

We split models based on model quality. There were 6 "better" cognitive models that scored either a '4' or a '5.' Seven "poorer" models scored a '3' or below. These categories will be used in later analyses. Three models each came from the class and paid participants.

Table 1. How the cognitive models were scored

Score	Description	Models Meeting Criterion
5	A model that produces behaviors close to an ideal model for fraction arithmetic, in terms of hints and just-in-time messages	4
4	A very good model that is beyond just being sufficient	2
3	A sufficient model, one that provides distinct and specific hints	3
2	An adequate model, but lacking in one or two ways (e.g., hints for only 2 of the 3 problem types)	2
1	Lacking in multiple ways; while it produces hints, it does not meet the specifications of the assignment (e.g., hints largely static)	2

Participants had to write a model that provided hints on 6 different entry boxes (see Figure 2). A cognitive modeler could use a single goalnode type to provide all hints, creating properties to distinguish them, or the modeler could use up to 6 goalnode types. Participants in our study used all possible numbers of goalnodes, with an average of 2.93. There were no statistically significant differences between the better and poorer cognitive models (2.50 v. 3.00, $t < 1$). All participants produced a flat object

model, meaning all defined goalnode types hung directly from the main default goalnode type. Participants defined 6.27 properties per goalnode type on average (the better and poorer cognitive models did not differ on this measure, 6.83 v. 6.22, $t < 1$).

We also examined participant's rule trees. The tree contains the hints that students will receive, as well as the just-in-time messages. Each node of the rule tree contains a predicate that tests aspects of the current problem's object instantiation and state. On average, the trees contained 13.12 nodes and were 2.08 nodes deep. The better and poorer cognitive models did not differ on either of these measures (for number of nodes, 13.17 v. 13.14, and for depth, 2.17 v. 2.00). The better and poorer cognitive models did differ somewhat on the number of just-in-time message defined (4.50 v. 0.86, $t(11) = 1.73$, $p = .1$), but it was the case that to be considered a better cognitive model it had to have at least one just-in-time message.

It is hard to be evaluative as to whether better models should be deeper on either the object model or rule tree. We have debated with other modelers whether "bushes" (flat models with more properties) or "trees" (deeper models with more object types) are better with regards to the object model, which has consequences for the predicated tree. Among our colleagues, there are a variety of opinions, so perhaps it is not surprising there is little difference on these measures.

3.2 Timing Data Concerning Cognitive Model Creation

As stated above, the average time to complete a cognitive model was 7.68 hr. The participants who produced the better cognitive models spent on average almost the same amount of time (7.67 hr, with a range of 4.98 hr to 13.08 hr) than the participants who produced the poorer models (7.68 hr, with a range of 3.42 hr to 13.82 hr).

The logging produced by the SDK provided much more detail than this. Each action that the participant performed within the tool's interface was time stamped with millisecond precision. Table 2 shows how much time the participants spent performing the component actions of creating a cognitive model, time spent on 1) the objects (creating objects and defining properties); 2) the rules (predicates, hints, and just-in-time messages); 3) defining instances; and 4) testing the model. One sees no differences on these measures between the better and poorer participants.

Table 2. Average time spent on various aspects of cognitive model construction ($n = 13$)

Category	Time (hr)	Percentage
Objects	2.45	31.8%
Rules	3.07	39.9%
Instances	1.65	21.4%
Testing	0.52	6.8%
Total	7.68	100%

A further analysis was performed that examined what actions the participants were performing during the time course of creating the cognitive model. Did most participants create their object model at the very beginning, and then turn to rule writing? Or, was there more give-and-take between working on the object model and the rules?

We created graphs for each participant that divided their progress in writing the model into deciles. Within each decile we calculated what percentage of the time was spent on object actions, rule actions, instances, and testing. Figure 3 shows two of these graphs, the one on top illustrating one of the participants who produced a better cognitive model, and the one on the bottom showing a poorer cognitive modeler.

The average participant produced 1156 actions (e.g., working on a hint, defining a property) as they worked on their cognitive model. These include edits and re-edits to the same entity. Again, there are no differences between the better and poorer participants on this measure (1181 v. 1205).

We examined the quantitative and qualitative aspects of these graphs for each participant. One difference that stands out, and can be seen in Figure 3, is the proportion

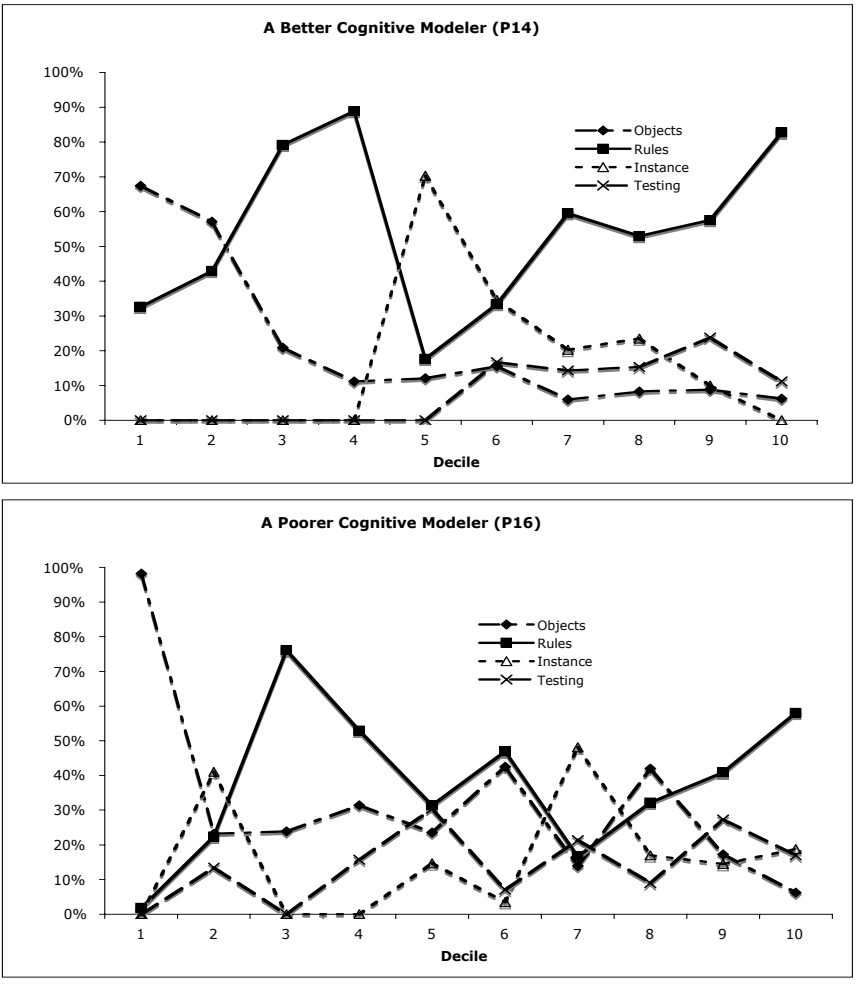


Fig. 3. Activity graphs of two participants

of time spent on the object model during the first half of the participant's time on task versus the second half. Some participants spent relatively less time on this task during the last half of their time than during the first half. That is, these participants appeared to have gotten the object model mostly right up front, and then did not modify it much after that. To quantify this observation, we counted as an "up-fronter" any participant who did not spend 30% of their time for more than one decile working on the object model during the last half of their editing. Under this definition, all of the better cognitive modelers were "up-fronters," whereas only 3 of the 7 poorer cognitive modelers were. This is a marginally significant difference by a chi-square test, $\chi^2(1, n = 13) = 2.86$, $p < .1$. Getting the model "right" (there are many potential "right" models) early appears important in creating a successful cognitive model. We investigated the possibility that some attribute associated with the cognitive modeler correlated with the ability to produce a "right" model early.

3.3 Exit Questionnaire Data

Participants completed an exit questionnaire. We asked demographic information such as undergraduate major, current department, and number of programming courses. Examining undergraduate majors, there was 1 communication major, 1 journalism major, and 2 who did not specify a major. The rest had majors associated with technology (5 computer scientists, 2 engineers, and 2 information technologists). Their graduate degree programs reflect a similar trend (5 HCI, 2 engineers, 1 CS, 1 economics, 1 journalist, 1 information systems, 1 instructional technology, and 1 did not specify). There were no psychologists or cognitive scientists in the group. In looking across who created the better versus the poorer cognitive models, there is no clear trend. There were roughly equal numbers of computer scientists and engineers who created better models than who created poorer models. And, one could find the "non-technology" majors represented in both groups.

Participants had taken an average of 4.36 programming classes prior to working on the cognitive model. Examining this measure with regards to the better versus poorer cognitive modelers does yield a significant difference, with the better cognitive modelers having taken more programming classes, 6.83 v. 1.57, $t(11) = 3.37$, $p < .01$.

Participants were also asked free response questions, where they reflected on their experiences doing the activity. In particular, they were asked to consider the challenges they encountered in creating the cognitive model, as well as the benefit in the approach. Almost all participants saw both positive and negative features of these kinds of cognitive models in general, and using this tool specifically. The software is still somewhat of a beta quality, and the documentation is not complete. Indeed, the screen capture movies and example models were created specifically for this assignment in order to obviate the need for more complete documentation. Furthermore, the means by which properties are referred to in predicates and hint messages, tutorscript (in Figure 1, one can see a couple examples of tutorscript inside curly braces), is not well defined. In all, 10 of the 13 participants mentioned either insufficient documentation or bugs in their response, with 5 specifically mentioning tutorscript. However, despite some issues with this particular tool, 11 of the 13 participants mentioned the generality of the tool; that is, it could be used to create tutors in a variety of different domains. Most (8) qualified their answer to include only domains with specific, finite

answers, such as chemistry, physics, and math. In actuality, this tool can be used to create any tutor appropriate for a model-tracing tutor. Lastly, a number of participants mentioned that in order to create a cognitive model, one needed to be very explicit about the steps the students should and can take, and that the steps needed to be complete in order to have a useful cognitive model. This is an accurate statement regarding these kinds of cognitive models, and we believe can be taken as both a strength and a weakness. Interestingly, it was 4 of the 6 better cognitive modelers who made such an observation, none of the poorer cognitive modelers made such a statement. A quote from one of the better cognitive modelers sums up many of our participants' reactions, "I was skeptical at first. It seemed like it took much more time to think through the model than just design a more direct system that would provide feedback to the user for each empty box…. However, my opinion changed when I got to the phase of the project where I could create instances of the problems. It went much faster than I expected, and I began to see that this system is much more flexible than I gave it credit for."

4 Discussion

We would like to highlight three main results from this study. First, the fact that the majority of participants (13 of 17, 77%) created a usable cognitive model with minimal instruction (less than 1 hr) is remarkable. Of the four who did not create a model, one experienced computer issues resulting in file loss and two had partial models. Historically, creating a cognitive model of this type required a Masters or Ph.D. level cognitive scientist with much training in ITSs and in the particular ITS tool. Indeed, the tool used to create the cognitive model was often directly within a programming language or within rudimentary tools built on top of a programming environment, requiring much programming knowledge. None of our participants were psychologists or cognitive scientists, and none were in their graduate training to produce ITSs. They created their models quickly, in under 8 hr on average. Accounting for watching demos and other instructional activities, these participants went from ITS neophytes to having a model in about 10 hr. A proper interface, more problems, and refinements to even the best model would still have to be made (in general, the "better" cognitive models would require little work to be usable by a student, but the "poorer" ones would have needed much additional help), so it is unclear how to gauge this effort with regards to hours of development per hour of instruction. However, it would be closer to 10:1 than it would 100:1. While the tools and methodology are different, it is somewhat interesting to compare these results to those of [3]. In their study, 12 of 13 graduate students (92%) taking a class specific to ITSs completed constraint-based tutor for a similar fraction addition task. It took them on average 31.3 hr to do so, but that included much time to manually transfer pseudo-code into a runnable system. To do the initial constraints, it took them 6.5 hr. However, that time does not include writing hints and the just-in-time messages that our participants wrote, in addition to the predicates. On par then it seems like these are at least somewhat similar results.

The second notable observation is how similar the better and poorer cognitive models were across a number of measures. In terms of time to construct the model, including an examination of working on certain subcomponents, along with certain

quantitative aspects of the model (number of objects and predicates, for example), there were no differences between the better and poorer models. This would make it difficult to look for certain markers (such as average depth of the predicate tree) or time measures and quickly determine if the model appears to be a quality model, if this observation holds true in future studies.

Perhaps the main difference we found between the better and poorer cognitive models is the third observation. The creators of the better cognitive models had taken more programming courses prior to the experiment. Given the correlational nature of this observation, it is hard to know the causal influences; did having more programming courses help them create better cognitive models, or did the fact that they might be better cognitive modelers to begin with attract them to programming courses? It seems reasonable to assume that programmers, either by their nature or because of their training, are better equipped to think about tasks that are conducive to putting them into a cognitive model. Specifically, we can think of at least two reasons why this might be so. First, much programming now is object-oriented in nature, and this is the representation used by the SDK. If a person is used to thinking about objects, properties, and inheritance, then being able to represent a task in the SDK should be easier. Second, although the rules are also represented in a hierarchy, they still have the "if-then" quality of production rules. Again, this kind of thinking is probably more natural to programmers.

Given the nature of the Cognitive Model SDK, particularly in relation to CTAT and its goal of addressing more specifically non-programmers [13], the association between programming and producing better cognitive models using the SDK is not surprising. Future lines of research should investigate this relationship in more depth, to examine which pieces of programming knowledge most helps in creating a cognitive model. In such a way we can move towards more people being able to make higher quality cognitive models.

Acknowledgments. This material is based upon work supported by the National Science Foundation under Grant No. OII-0548754. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We would also like to thank Matt McHenry, Tristan Nixon, Steven Ritter, Janea Triplett, and Leslie Wheeler for their help on this project.

References

1. Blessing, S.B., Gilbert, S., Ritter, S.: Developing an authoring system for cognitive models within commercial-quality ITSs. In: Proceedings of the Nineteenth International FLAIRS Conference, pp. 497–502. AAAI Press, Melbourne (2006)
2. Blessing, S., Gilbert, S., Ourada, S., Ritter, S.: Lowering the bar for creating model-tracing intelligent tutoring systems. In: Proceedings of the 13th International Conference on Artificial Intelligence in Education, Marina del Rey, CA, pp. 443–450. IOS Press, Amsterdam (2007)
3. Suraweera, P., Mitrovic, A., Martin, B.: Constraint authoring system: An empirical evaluation. In: Proceedings of the 13th International Conference on Artificial Intelligence in Education, Marina del Rey, CA, pp. 451–458. IOS Press, Amsterdam (2007)

4. Corbett, A.T.: Cognitive computer tutors: Solving the two-sigma problem. In: Bauer, M., Gmytrasiewicz, P.J., Vassileva, J. (eds.) UM 2001. LNCS (LNAI), vol. 2109. Springer, Heidelberg (2001)
5. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A.: Intelligent tutoring goes to school in the big city. International Journal of Artificial Intelligence in Education 8, 30–43 (1997)
6. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The andes physics tutoring system: Lessons learned. International Journal of Artificial Intelligence and Education 15(3) (2005)
7. Murray, T., Blessing, S., Ainsworth, S.: Authoring tools for advanced technology educational software. Kluwer Academic Publishers (2003)
8. Woolf, B.P., Cunningham, P.: Building a community memory for intelligent tutoring systems. In: AAAI 1987, pp. 82–89 (1987)
9. Anderson, J.R.: Rules of the Mind. Erlbaum, Hillsdale (1993)
10. Ainsworth, S.E., Fleming, P.F.: Evaluating a mixed-initiative authoring environment: is redeem for real? In: Proceedings of the 12th International Conference on Artificial Intelligence in Education, pp. 9–16. IOS Press, Amsterdam (2005)
11. Halff, H.M., Hsieh, P.Y., Wenzel, B.M., Chudanov, T.J., Dirnberger, M.T., Gibson, E.G., Redfield, C.L.: Requiem for a development system: Reflections on knowledge-based, generative instruction. In: Murray, T., Blessing, S., Ainsworth, S. (eds.) Authoring tools for advanced technology educational software, Kluwer Academic Publishers (2003)
12. Mathan, S., Koedinger, K., Corbett, A., Hyndman, A.: Effective strategies for bridging gulfs between users and computer systems. In: Proceedings of HCI-Aero 2000: International Conference on Human Computer Interaction in Aeronautics, Toulouse, France, September 27- 29, pp. 197–202 (2000)
13. Aleven, V., Sewall, J., McLaren, B.M., Koedinger, K.R.: Rapid authoring of intelligent tutors for real-world and experimental use. In: Kinshuk, R., Koper, P., Kommers, P., Kirschner, D.G. (eds.) Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies (ICALT 2006), pp. 847–851. IEEE Computer Society, Los Alamitos (2006)

Open Community Authoring of Targeted Worked Example Problems

Turadg Aleahmad, Vincent Aleven, and Robert Kraut

Human Computer Interaction Institute
Carnegie Mellon University, Pittsburgh PA 15213, USA
{turadg,aleven,robert.kraut}@cmu.edu

Abstract. Open collaborative authoring systems such as Wikipedia are growing in use and impact. How well does this model work for the development of educational resources? In particular, can volunteers contribute materials of sufficient quality? Could they create resources that meet students' specific learning needs and engage their personal characteristics? Our experiment explored these questions using a novel web-based tool for authoring worked examples. Participants were professional teachers (math and non-math) and amateurs. Participants were randomly assigned to the basic tool, or to an enhanced version that prompted authors to create materials for a specific (fictitious) student. We find that while there are differences by teaching status, all three groups make contributions of worth and that targeting a specific student leads contributors to author materials with greater potential to engage students. The experiment suggests that community authoring of educational resources is a feasible model of development and can enable new levels of personalization.

1 Introduction

Traditionally in the development of educational resources, one person or cohesive group produces each artifact, be it paper textbook or computer tutor. This model can require great resources from one group, particularly for tutoring systems and even more so for individualized tutoring. Traditional model-based tutors cost 100-1000 hours of time from skilled experts [1,2]. Newer approaches [3] such as CTAT [4], REDEEM [5] and ASSISTment [6] lower the expertise necessary to create a tutor, but still require coordination of a group to create useful tutors. Like most tutors, they are also limited in the dimensions by which they can personalize to the student. In this paper we describe and study a prototype tutor authoring system that uses community volunteers to create personalized instruction.

Over the last decade we have seen new development models that take advantage of the openness of the World Wide Web. Encyclopedias, web browsers, computer operating systems, and other complex artifacts have been created by loose networks of volunteers building on each other's contributions. These openly developed products often meet and sometimes exceed the quality of more cohesive sources and in general lower their costs. In education, there are a number of

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 216–227, 2008.

open access initiatives such as MIT's OpenCourseWare or CMU's Open Learning Initiative. These systems are open in that they are free to use, but not in the sense of open to contributions. Wikipedia is an example of a system open to all (even anonymous) contributors and is excellent for sharing *about* something, though it disallows instructional information. Connexions [7] has been successful in supporting collaborative development of textbooks and newer site like Curriki, Wikibooks and Wikiversity are trying to adapt existing wiki software to the development of educational materials. However, these systems are all focused on traditional inert learning resources and are limited by available wiki software. Community authored intelligent tutoring systems will need semantic content structures and affordances for personalized instruction.

We propose a four phase cycle of a new system for open resource development: Generation, Evaluation, Use, and Improvement. This study is an exploration into the Generation phase. Will volunteers produce materials of sufficiently high quality that they can be put in the hands of students with little or no editorial oversight? We report results obtained with a simple web-based authoring interface for worked example problems, a tutoring resource that is very intuitive and hence can be authored by lay volunteers. Worked example problems are versatile for computer tutoring systems. They can be presented to a student as a full problem and solution, as a problem with a partial solution, or as simply a problem, as on a homework sheet. In more intelligent tutoring systems, scaffolds contained within the problem can be faded as the student masters each skill. Because worked example problems enhance problem solving in intelligent tutoring systems, they also complement what already exists [8,9]. Pedagogically, worked-examples both instruct and help to foster self-explanation [10]. Thus they are a useful educational resource, although we do not test their usefulness in this paper.

To explore the impact of open development and diverse levels of expertise, our study was open to all comers. Reasonably this would lead to a volume of content without much value and this motivated our first hypothesis. *H1: Identifying the good from the bad contributions is easy. We expect that all contributions are good, easily fixed, or easily filtered.* To assess the impact of expertise, we asked each participant whether they were math teachers, other teachers, or not teachers at all. We used this data to assess *H2: Math teachers submit the best contributions.* While math enthusiast amateurs may have the appropriate *content knowledge* and non-math teachers may have the appropriate *pedagogical knowledge*, neither will have much *pedagogical content knowledge* (as defined by Shulman [11]).

A goal of the system is to facilitate personalized instruction. Personalization has already been shown to improve both student engagement and test scores. Fourth grade math students make better pretest-to-posttest gains with personalized instruction and also perform significantly better on both the pretest and posttest problems[12]. Similar effects have been found with 5^{th}th and 6^{th}th grade students [13]. Personalized instruction has also been demonstrated to increase the engagement and learning outcomes of minority groups [14, e.g. Hispanic]. To facilitate personalization, we created a feature in the tool in which authors are

prompted to target their instruction to a specific student, described through a student profile. Thus, *H3: Student profiles lead to tailored contributions.*

Because being shown a specific individual to help is likely to draw out more altruistic behavior, we believed that the profiles may motivate authors to make better contributions [15], leading to two further hypotheses. *H4: Student profiles increase the effort of authors. H5: Student profiles lead to higher quality contributions.*

The main analysis reported in this paper looks at the contributions targeted at one specific skill, in order to control for variability among skills. The skill of understanding and applying the Pythagorean Theorem was chosen for three reasons. The first is that it is difficult to learn. Data from the ASSISTment assessment system [16] show it is the most difficult skill for students to acquire. The second is that because it has a visual component, a variety of problems are more difficult to generate by machine than with non-visual math skills. The third is that it affords a variety of real-world scenarios to demonstrate it.

In this paper, we describe the web-based community authoring tools, report the volunteer contributions that were submitted through the web site, and analyze the data to see if they support the hypotheses described above. We discuss the implications for web-based community authoring of instructional materials.

2 Methods

2.1 Apparatus

The experiment took place over the internet through interactions with a newly-built prototype web application for community-based authoring. Participants followed a URL to an experiment registration form. This form explained the goal of producing open educational resources and asked for their consent in the study. Participants were asked for their age and teacher status (math teacher, other teacher, or non-teacher / amateur). Those under 18 were directed straight to a survey without participating in the experiment.

Participants were then presented a page explaining what a worked example problem is and that their task is to create a worked example problem to teach the Pythagorean Theorem. They were provided a search box to look up on the web anything they wanted to learn or refresh themselves on and some simplified pedagogical principles to remember in creating their worked example. The next screen was the authoring tool. The tabs at the top guided the user in the task. The first sentence of the Start tab was specific to the condition within the sub-experiment. In the student profile experimental condition it read, Please create a worked-out example to provide practice to the student above in understanding and applying the Pythagorean Theorem. In the control generic condition, the words to the student above were stricken. Authors wrote the text of the problem statement in the large box to the left and drew their diagrams using a simple pen tool in the box to the right. After developing their problem statement and diagram, they clicked Add Step to add each step of the solution. The solution area had three columns, one for the work for the student to perform towards the solution, one for the explanation of the work in each step.

In the student profile experimental condition, the top of the authoring tool showed a floating description of the student to target with the example-to-be-authored. The instruction to the participant also differed, as described above. The participants in the experimental condition were presented a student profile to target. Profiles were randomly selected and participants saw a new one for each problem authored. Student profiles were designed to vary on six dimensions that might differentiate the learning patterns of real students. They varied on three dimensions of skill to increase variation of the submissions on skill-level appropriateness. These were proficiency in at the Pythagorean Theorem, general proficiency in math, and verbal skill. They were also varied on cultural attributes to prompt creativity of the participants and increase the personal relevance of the examples to students. These were gender, hobbies/interests, and home environment. Four hobbies were crossed with four home environments to create 16 unique student profiles. Distributed evenly among them were four skill profiles and the two genders. Additionally, each was assigned a favorite color to round out the description presented. Figure 1 shows an example profile.

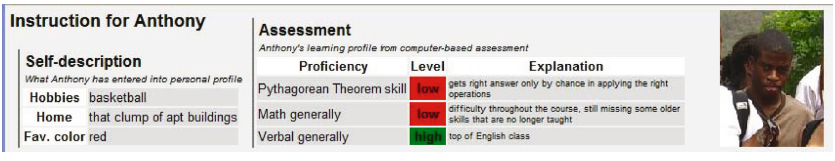


Fig. 1. Sample profile in experimental condition

Participants. The URL to participate was advertised on various web sites both related to education and not. Participants could earn up to $12 for their contributions, regardless of their quality. During the experiment 1427 people registered on the site to participate. After seeing the task in detail most did not continue, but 570 participants did use the system to submit 1130 contributions. Table 1 shows by teacher status the number of registered and contributing participants during the experiment.

Since completion of the experiment, more participants have contributed to the site but they are not included in the analysis that follows. At the close of the experiment, the web site was disabled. At the request of people who still wanted to participate, two months later it was restored with the compensation removed. In the four months that have elapsed, 93 people have registered and submitted 93 contributions, of which 40 pass machine vetting (see below). These extras have not been further tested for quality.

Machine Coding. The submissions were analyzed by software as a first-pass filter for minimum quality. The criteria were that the problem description had a length between 50 and 1000 characters and that at least one step was provided towards the solution. This machine filtering of the 1130 raw submissions left 551 machine-vetted submissions from 281 participants. Table 1 shows by teacher status the number of participants whose contributions passed machine vetting.

Table 1. Count of participants by teacher status and depth of participant

Participation	Math teachers	Other teachers	Amateurs
Registered	131	170	1126
Contributed also	70	72	428
Passed vetting also	26	35	220

Expert Coding. After the automatic filter, the remaining submissions were coded for quality by human experts. In a production version of the site, human coding would be drawn from the community. For this analysis, the two coders were a retired and a beginning math teacher. They coded using a custom web application that ensured they were blind to the sources. Submissions were coded on three criteria: quality of the problem statement (Statement), quality of the work shown (Work), and quality of the explanation of the solution (Explanation). The following are the ratings and definitions that they used:

(0) **Useless** No use in teaching and it would be easier to write a new one than improve this one.
(1) **Easy fix** Has some faults, but they are obvious and can be fixed easily, in under 5 minutes.
(2) **Worthy** Worthy of being given to a student who matches on the difficulty and subject matter. Assume that the system knows what's in the problem and what is appropriate for each student, based on their skills and interests.
(3) **Excellent** Excellent example to provide to some student. Again, assume that the system knows what's in the problem and what is appropriate for each student, based on their skills and interests.

For quantitative analyses the categories were assigned the integers 0-3. The work and explanations were averaged to create a Solution quality and all three components were averaged to determine a Whole quality. Inter-rater reliability of the Statement quality was alpha=0.61, for the Solution quality was alpha=0.81, and for the Whole quality was alpha=0.78.

Ordinal variables were modeled as continuous in order to model the participant as a random effect. This requirement was due to a limitation in the statistic software available.

3 Results of Open Authoring

To test H1 we looked at the quality of all problems submitted and the work needed to classify them. Of 1130 raw submissions, 11% of whole problems (statements with solutions) were classified as Worthy, meaning that they are fit for use immediately. 39% were at least Fixable, meaning that they would be valuable with some additional effort. In general the statements were of higher quality than the solutions. 27% of statements were Worthy and 4% were Excellent as is. Figure 2 shows counts in each quality classification, plus a fifth column indicating

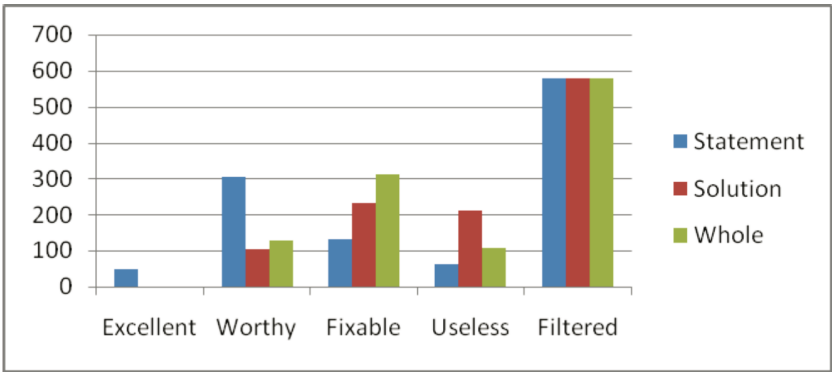


Fig. 2. Count of submissions in each quality classification

the 579 that were filtered by the machine vetting process. Classification into the Filtered category was a trivial computation. Classifying the three contribution components into the four-rating rubric took experts a median time of 36 seconds per contribution.

To test H2 we looked at the quality of each contribution as a whole, revealing no superior quality by teacher status ($F(2)=1.53$, $p=0.22$). Further analysis revealed that the effect on quality of teacher status interacted with the problem component.

Math teachers were best at writing problems statements. A comparison across teacher status showed a marginally significant effect ($F(2)=2.39$, $p=0.093$). Math teachers' contributions rated at $M=1.79$, followed by amateurs ($M=1.45$) and other teachers ($M=1.45$). A comparison of math teachers with the rest showed a significant effect ($F(1)=4.80$, $p=0.015$, one-tailed).

Contrary to H2, amateurs were best at writing solutions. A comparison across teacher status showed a marginally significant effect ($F(2)=2.73$, $p=0.067$). Amateurs did best ($M=0.72$) followed by math teachers ($M=0.60$) and then other teachers ($M=0.48$). A comparison of amateurs with the rest showed a significant effect ($F(1)=4.87$, $p=.028$).

4 Discussion on Open Authoring

Through automated methods and software supports for human judges, all the contributions were rated easily, supporting H1. In a short amount of time about 1500 people registered to contribute to a commons of educational materials. While not all came with the same intentions, over half walked away after determining that they did not want to participate fully. Of the raw submissions made, over half were trivial to filter by simple automated methods. Of the remaining, a novice and a veteran teacher were able to rate each of them on three attributes in less than a minute each. About 1/10th were ready to help students learn without

needing any modification. Many more were rated as fixable, meaning that with some additional work they would be ready. Statements were the highest quality components and solutions were the most difficult parts to author well.

Teacher status had an important impact on the quality of the components of contributions. As predicted in H2, math teachers were best at authoring problem statements. Surprisingly, amateurs authored the best worked solutions. Perhaps this is because they are better able to adopt a student's perspective. Math teachers performed worse than amateurs but better than non-math teachers. Perhaps this is because their pedagogical content knowledge helps compensate (but not fully) for their expert blind spots.

Overall, it is clear that, at least for worked examples of the Pythagorean Theorem, participants of all teaching statuses were likely to make contributions of value. Math teachers do a better job at some parts of the process, but even laymen do fairly well. This is fortunate because there are many more amateurs in the world than math teachers. In this study each participant made each contribution independently, but the best resources may come from collaboration. For example, a math teacher writes a problem statement and an amateur writes the solution. Educational content systems can benefit from opening the channels of contribution to all comers.

5 Results of Student Profile

Tailoring was analyzed to test H3 and measured as the degree to which various attributes of the contributed problem matched that of a student profile. Matching took two forms: using words primed by the student profile and matching the difficulty (reading or writing) to the skill levels in the profile. The use of words in the submission was analyzed using LIWC, a word counting tool, with its default dictionary [17] plus the word piano in the music category (to go with guitar, instrument, concert, etc.). Table 2 summarizes the results for the word matching. Mentioning an attribute drew out significant increases in authoring with that attribute on almost every measure, both over the generic condition and other profiles.

To test whether authors tailor their contributions to the verbal skill of the student, we compared the verbal skill level of the student profile presented to the author with the reading level of the authored submission. The reading level was measured using the Flesch-Kincaid Grade Level Formula. [18] This indicates U.S. school reading grade level, making the problem selection easy in real learning contexts. The text analyzed is the concatenation of the problem statement and all the explanation steps. Because readability metrics aren't calibrated to math expressions, the work steps were omitted from readability analysis. Outliers were curtailed by removing the top and bottom 2.5% percentile in the distribution of Flesch-Kincaid Grade Level (leaving -1.32 to 11.71). An F-test showed the differences across profile verbal skill levels to be significant ($F(2)=2.95$, $p=0.023$, one-tailed). Table 3 shows the results of pair-wise t-tests. Additionally, it is worth noting that authors sometimes took the student's verbal skill level as a cue for the subject matter of the problem statement, as in the following submission:

Table 2. Probabilities of authoring matching an attribute

Attribute	With generic (G)	With profiles not mentioning attribute (N)	With profiles mentioning attribute (M)	F-test (G-M)	F-test (N-M)
Female pronoun	5%	4%	16%	9.68*	12.82**
Male pronoun	19%	14%	19%	0.004	1.19
Sports word	9%	9%	24%	18.01**	11.89**
TV word	4%	4%	10%	8.36*	2.63†
Music word	2%	2%	9%	6.92*	8.93**
Home word	14%	n/a	20	3.60*	n/a

†p<.10 *p<.05 **p<.001

"Shakespeare sat down one day" and had a revolutionary idea. He would write text diagonally across a page rather than horizontally! He imagined the reader's surprise when they turned the page and saw such a change of events! However, he needed to carefully plan out how much space he had to write in, if he only wanted to write one line across the diagonal of the book. Unfortunately, Shakespare had no rules. After all, he thought he would never need one being a writer! Luckily for him, he remembered that the blank books and pages he ordered were exactly 6 inches wide and 8 inches tall. How much space does Shakespare have to write in?

Table 3. Correspondence of verbal and math skill levels with the authoring interface

Verbal Skill in Profile	Sign. Diffs	Mean Reading Level of Contribution	Std Err	General Math Skill in Profile	Sign. Diffs	Probability of Using 3-4-5 Triangle	Std Err
High	A	3.78	0.24	High	A	16%	0.05
Medium	A B	3.56	0.32	Medium	A B	26%	0.05
Low	B	2.93	0.33	Low	B	27%	0.04
None (control)	B	3.20	0.16	None (control)	A B	21%	0.03

Math difficulty was measured more simply because there is no established metric available. Since all problems were on the Pythagorean Theorem, we chose to measure math difficulty by whether the problem uses only the 3-4-5 triangle, the least challenging numerical solution. An F-test showed the differences across profile general math proficiency levels to be marginally significant ($F(1)=2.35$, $p=0.063$, one-tailed). Table 3 shows the results of pair-wise t-tests.

Effort was analyzed to test H4. It was measured by both the length of each contribution and the time spent on it by the author. While authors in the generic control condition wrote an average of 766 characters per contribution, authors in the student profile condition wrote 847 characters, a marginally significant difference ($F(1)=2.35$, $p=0.063$, one-tailed). Most of that difference is accounted for by the problem statements, for which the generic condition wrote 204 characters versus 250 characters with profiles. This 23% increase in length was significant

($F(1)=8.61$, $p=0.01$). A similar analysis of time spent authoring (normalized) revealed no significant differences ($t(235)=-0.15$, $p=0.56$).

Quality was analyzed to test H5. The quality of the statement, the solution, and the whole were compared between the experimental and control conditions. F-tests showed no effects of the student profiles on the quality of submissions.

6 Discussion on Student Profile

All features of the profile display were accounted for in the problems submitted. Participants were more likely to mention a particular hobby when shown it in the profile. They were also more likely to make mention of some home environment (a feature of every profile). Particularly striking is the increase in likelihood of including a female in the problem statement. Without a profile, males were used in 19% of problem statements and females in just 5%. (The rest used only it or no pronouns.) Female student profiles bring female pronoun usage up to 16%, almost on par with males. Male pronoun usage is clearly the default of most authors since the usage without any profile is just as high as with a male profile. Furthermore, male pronoun usage was not much suppressed by the female profiles.

Participants shown the student profiles also tailored the skill level of their contributions. High and low reading levels differed by almost a grade level. Submissions for profiles with high general math skill level were one third less likely to make use of simple 3-4-5 triangle problems. Participants shown profiles of students wrote problem statements that were 25% longer.

It is perhaps odd then that they didn't spend significantly more time on these statements. One explanation is that the time typing is negligible compared to the time required to generate an idea. That the statements in the profile condition are so much longer suggests that the profile prompts ideas that are more involved.

One hypothesis on the student profiles condition did not bear out: that profiles would lead to contributions of higher quality on an absolute scale. Instead the contributions maintained quality. In other words, the tailoring came at no cost to the generic quality of the contributions.

7 General Discussion and Conclusions

This study looks at the feasibility of an open development model for developing resources for tutoring systems and a particular design feature to solicit to draw more and better work from contributors. We found that while over half the contributions were useless there were some gems. Importantly, it required little effort to separate the wheat from the chaff, confirming H1. Both professional educators and amateurs contributed a large portion of useful materials. Contrary to the prediction H2, contributions from math teachers were not superior to those from others. This is encouraging because there are many more people who aren't math teachers than who are.

Math teachers did write the best problem statements but amateurs wrote the best solutions. This suggests a model in which math teachers contribute the problem statements and amateurs write the solutions. In general, it suggests that users of different aptitudes and abilities be directed to different tasks within the collaborative authoring system, a solid design implication.

The student profile feature of the interface successfully drew out personalized resources (H3). On every attribute the profile increased the likelihood of targeting it. The profiles also drew out more effort on the part of participants (H4). While the profiles didn't measurably improve the quality of their contributions (H5), it is important to note that they did not impair them either.

An important limitation of the study is that there are no measures yet of how these contributions actually aid learning. The expert ratings were taken as proxies for the utility in real learning contexts, but the true test will be using the materials to teach real students and measure their gains versus alternative materials. One potential pitfall is that the personalizing details in the tailored resources distract students from learning. Of course, the improvements to their motivation might offset this. A real-world study is necessary to answer these questions.

Another key limitation of the findings here is the ecological validity of paying participants for their contributions. The problem is not that participants were incented to contribute. One can imagine a future system with incentives such as peer status or competitions with non-monetary awards. (e.g. [19]) Certainly, volunteers are always motivated by some incentive, external or internal. How though do contributions differ under more ecologically valid incentives? Because participants were paid for any contribution, there is good reason to believe that real world volunteers would be more dedicated and likely to produce higher quality materials on average.

This study focused on phase one, Generation, of a proposed cycle of development. It also touched upon phase two, Evaluation, through the finding that experts can evaluate each contribution in about half a minute. We will turn next to the fourth phase, Use, in order to begin answering the above limitations. We will first develop a personalized homework system that creates homework assignments for students based on their real personal profiles. This will also require expanding the Generation aspects of the system to facilitate a wider variety in the skills taught. Farther in the future, we will explore the potential for developing interactive tutoring around the worked example problem artifact, through dynamic scaffolding of its subcomponents.

This study has positively, if partially, demonstrated the feasibility of an open development model for resources for tutoring. Volunteers regardless of professional expertise are able to make useful contributions and features of the authoring interface can incline contributions to have different features and make instruction more socially inclusive.

This work was supported in part by Graduate Training Grant awarded to Carnegie Mellon University by the Department of Education (#R305B040063). The research reported here was supported by the Institute of Education

Sciences, U.S. Department of Education, through "Effective Mathematics Education Research" program grant #R305K03140 to Carnegie Mellon University. The opinions expressed are those of the authors and do not represent views of the U.S. Department of Education. The photo shown in the student profile included in this paper came from Flickr user *jenrock* under a Creative Commons Attribution-Noncommercial 2.0 Generic license.

References

1. Anderson, J.R.: Rules of the mind. Erlbaum, Hillsdale (1993)
2. Murray, T.: Authoring intelligent tutoring systems: An analysis of the state of the art (International Journal of Artificial Intelligence in Education) 10, 98–129
3. Murray, T., B.S.A.S. (eds.): Tools for Advanced Technology Learning Environments. Kluwer Academic Publishers, Amsterdam (2003)
4. Aleven, V., Sewall, J., McLaren, B.M., Koedinger, K.R.: Rapid authoring of intelligent tutors for real-world and experimental use. In: Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies (ICALT 2006), pp. 847–851. IEEE Computer Society, Los Alamitos (2006)
5. Ainsworth, S., Fleming, P.: Evaluating a mixed-initiative authoring environment: Is redeem for real? In: Proceedings of the 12th International Conference on Artificial Intelligence in Education, pp. 9–16. IOS Press, Amsterdam (2005)
6. Turner, T.E., Macasek, M.A., Nuzzo-Jones, G., Heffernan, N.T.: The assistment builder: A rapid development tool for ITS. In: Proceedings of the 12th Annual Conference on Artificial Intelligence in Education, pp. 929–931 (2005)
7. Baraniuk, R., Burrus, C.S., Hendricks, B., Henry, G., Hero, A., Johnson, D.H., Jones, D.L., Nowak, R., Odegard, J., Potter, L., Reedstrom, R., Schniter, P., Selesnick, I., Williams, D., Wilson., W.: Connexions: Education for a networked world. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, Orlando, ICASSP (2002)
8. McLaren, B.M., Lim, S.J., Gagnon, F., Yaron, D., Koedinger, K.R.: Studying the effects of personalized language and worked examples in the context of a web-based intelligent tutor. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 318–328. Springer, Heidelberg (2006)
9. Schwonke, R., Wittwer, J., Aleven, V., Salden, R., Krieg, C., Renkl, A.: Can tutored problem solving benefit from faded worked-out examples. In: European Cognitive Science Conference, pp. 23–27 (2007)
10. Renkl, A., Atkinson, R.K.: Learning from examples: Fostering self-explanations in computer-based learning environments. Interactive Learning Environments 10(2), 105–119 (2002)
11. Shulman, L.S.: Those who understand: Knowledge growth in teaching. Educational Researcher 15(2), 4–14 (1986)
12. Ku, H.Y., Sullivan, H.: Student performance and attitudes using personalized mathematics instruction. Educational Technology Research and Development 50(1), 21–34 (2002)
13. Anand, P.G., Ross, S.M.: Using computer-assisted instruction to personalize arithmetic materials for elementary school children. Journal of Educational Psychology v79(n1), 72–78 (1987)
14. López, C., Sullivan, H.: Effect of personalization of instructional context on the achievement and attitudes of hispanic students. Educational Technology Research and Development 40(4), 5–14 (1992)

15. Small, D.A., Loewenstein, G.: Helping a victim or helping the victim: Altruism and identifiability. Journal of Risk and Uncertainty 26(1), 5–16 (2003)
16. Feng, M., Heffernan, N.T., Koedinger, K.R.: Predicting state test scores better with intelligent tutoring systems: Developing metrics to measure assistance required. In: Proceedings of the 8th International Conference on Intelligent Tutoring Systems, pp. 31–40. Springer, Berlin (2006)
17. Pennebaker, J., Francis, M., Booth, R.: Linguistic inquiry and word count: LIWC. Erlbaum Publishers, Mahwah (2001)
18. Kincaid, J., Fishburne, R., Rodgers, R., Chissom, B.: Derivation of new readability formulas for navy enlisted personnel, 8–75 (1975)
19. Cheng, R., Vassileva, J.: Design and evaluation of an adaptive incentive mechanism for sustained educational online communities. User Modeling and User-Adapted Interaction 16(3-4), 321–348 (2006)

Agent Shell for the Development of Tutoring Systems for Expert Problem Solving Knowledge*

Vu Le[1,2], Gheorghe Tecuci[1], and Mihai Boicu[1]

[1] MSN 6B3, Learning Agents Center, Volgenau School of Information Technology and Engineering, George Mason University, 4400 University Dr., Fairfax, VA 22030, USA
vu.le@baesystems.com, tecuci@gmu.edu, mboicu@gmu.edu
http://lac.gmu.edu
[2] Advanced Information Technology, BAE Systems, 3811 N. Fairfax, Arlington, VA 22203

Abstract. This paper introduces the concept of learning and tutoring agent shell as a general and powerful tool for rapid development of a new type of intelligent assistants that can learn complex problem solving expertise directly from human experts, can support human experts in problem solving and decision making, and can teach their problem solving expertise to non-experts. This shell synergistically integrates general problem solving, learning and tutoring engines and has been used to build a complex cognitive assistant for intelligence analysts. This assistant has been successfully used and evaluated in courses at US Army War College and George Mason University. The goal of this paper is to provide an intuitive overview of the tutoring-related capabilities of this shell which rely heavily on its problem solving and learning capabilities. They include the capability to rapidly acquire the basic abstract problem solving strategies of the application domain, directly from a subject matter expert. They allow an instructional designer to rapidly design lessons for teaching these abstract problem solving strategies, without the need of defining examples because they are automatically generated by the system from the domain knowledge base. They also allow rapid learning of test questions to assess students' problem solving knowledge. The proposed type of cognitive assistant, capable of learning, problem solving and tutoring, as well as the learning and tutoring agent shell used to build it, represent a very promising and expected evolution for the knowledge-based agents for "ill-defined" domains.

Keywords: ITS-building tool, ITS authoring, agent-based tutoring systems, knowledge acquisition, machine learning in ITS, intelligent agents, ill-defined domains, intelligence analysis, teaching problem solving expertise, lesson design and generation, test learning and generation.

* This material is based on research partially sponsored by the Air Force Office of Scientific Research (FA9550-07-1-0268) and the Air Force Research Laboratory (FA8750-04-1-0257). The US Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official polices or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, the Air Force Research Laboratory or the U.S. Government.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 228–238, 2008.

1 Introduction

Building intelligent tutoring systems is notoriously hard. It requires teams that typically include software developers, knowledge engineers, subject matter experts, and instructional designers, which are estimated to need between 300 and 1000 hours to produce an hour of instructional material [1, 2]. Although impressive successes have been demonstrated by many advanced ITS authoring systems, such as CAT [3] or RIDES [4], these and the vast majority of the developed systems are for "well-defined" domains (which allow for a clear distinction between good and bad answers or solutions), in spite of the fact that many domains, such as design, law, medical diagnosis, history, intelligence analysis, and military planning, are "ill-defined" [5].

This paper presents a new approach for building tutoring systems that can teach new professionals how experts solve problems in a complex "ill-defined" domain [5]. This approach leads to the development of a new type of cognitive assistant that in addition to tutoring has powerful capabilities for learning and problem solving. It can:

- rapidly *learn*, directly from a subject matter expert, the problem solving expertise which currently takes years to establish, is lost when experts separate from service, and is costly to replace;
- *tutor* new professionals the problem solving expertise learned from the subject matter expert;
- assist a professional to *solve complex problems*, through mixed-initiative reasoning, allowing a synergistic integration of the professional's experience and creativity with the agent's knowledge and speed, and facilitating collaboration with complementary experts and their agents.

The developed methods for building such cognitive assistants have been implemented into a new type of tool, called learning and tutoring agent shell. This shell can be taught by a subject matter expert, and can then teach students in ways that are similar to how it was taught. The shell has been used to build a cognitive assistant for intelligence analysts that has been successfully used in courses at the US Army War College and George Mason University [6]. Because of the number and complexity of the methods integrated into the shell, in this paper we will present only those related to its tutoring capabilities, giving only minimal information about the others, which are described in [7], [8]. Moreover, even the tutoring capabilities will be presented at a very general, conceptual level. Our goal is to provide an as intuitive as possible view on this new approach to building systems for tutoring complex problem solving expertise in "ill-defined domains". A detailed description of this approach is presented in [9].

The next section introduces the concept of learning and tutoring agent shell and the associated methodology for building cognitive assistants. Section 3 introduces the intelligence analysis domain, as well as the problem solving approach implemented in the shell, which is pedagogically tuned. Section 4 presents our approach to the abstraction of the reasoning trees which facilitates the identification and tutoring of the main problem solving strategies of a domain. Section 5 presents the lesson design and generation process and Section 6 presents the learning and generation of test questions. This is followed by a presentation of some experimental results, a summary of the main contributions, current limitations and future research directions.

2 Learning and Tutoring Agent Shell (LTAS)

Fig. 1 shows the overall architecture of an LTAS. This is an extension of the concept of *learning agent shell* [7] which is itself an extension of the concept of *expert system shell* [10].

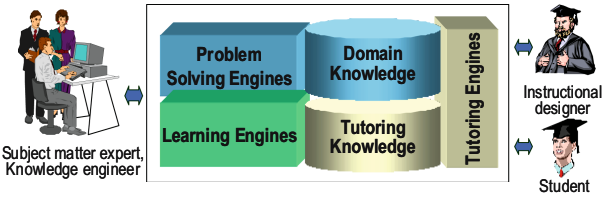


Fig. 1. Learning and Tutoring Agent Shell

The problem solving engines of our LTAS employ a general, divide-and-conquer, approach to problem solving, called problem-reduction/solution-synthesis, which is applicable in a wide range of domains [7], [11]. In this approach, which will be illustrated in the next section, a complex problem is successively reduced to simpler and simpler problems, the solutions of the simplest problems are found, and then these solutions are successively composed, from bottom up, until the solution of the initial problem is obtained. To exhibit this type of problem solving behavior, the domain knowledge base should contain an object ontology (which describes the objects from an application domain) and a set of problem reduction or solution synthesis rules (expressed with the objects from the ontology). A problem reduction rule expresses how and under what conditions a generic problem can be reduced to simpler generic problems. A solution synthesis rule expresses how and under what conditions the solutions of generic subproblems can be combined into the solution of a generic problem. The conditions are complex first-order logical expressions [6], [8].

The learning engines employ general mixed-initiative, multistrategy methods that allow a subject matter expert to teach the agent in a way that is similar to how the expert would teach a person [7], [8]. For instance, the expert will show the agent how to solve a specific problem, will help it to understand the corresponding reasoning process, and will supervise and correct its behavior when the agent attempts to solve similar problems. As a result, the agent will learn general reduction and synthesis rules and will extend its ontology. Moreover, the acquired knowledge will be pedagogically tuned [12] because the agent will solve new problems and will explain its reasoning process similarly to how the expert did it.

The tutoring engines, which will be described in more detail in this paper, allow the acquisition of pedagogical knowledge, the design and generation of lessons, and the learning and generation of test questions.

To build a cognitive assistant, the subject matter expert first teaches the agent shell (LTAS) and develops its domain knowledge base (consisting of the object ontology, the problem reduction rules and the solution synthesis rules). Then the expert teaches the agent the elementary abstract reasoning strategies of the application domain, as discussed in Section 4. After that, the instructional designer designs the lessons based on the abstract reasoning strategies. The instructional designer also teaches the agent how to generate test questions, as discussed in Section 5.

We will provide an intuitive overview of the tutoring-related capabilities based on the Disciple-LTA analyst's cognitive assistant that was developed with the learning and tutoring shell for the "ill-structured" domain of intelligence analysis [6], [13].

3 Disciple-LTA: Analyst's Cognitive Assistant

Disciple-LTA solves intelligence analysis problems, such as, "Assess whether Iran is pursuing nuclear power for peaceful purposes" or "Assess whether Al Qaeda has nuclear weapons, based on partial, uncertain, and even false information from open-source pieces of evidence (such as, newspaper articles, web sites, news agency reports, books, etc.).

As indicated in the previous section, the way the agent solves an intelligence analysis problem is similar to how the expert solved such problems when he or she taught the agent. It is as if the agent is "thinking aloud", asking itself questions that guide the problem reduction process, as illustrated in Fig. 2 and copied below:

I need to: Assess whether Al Qaeda has nuclear weapons.
Q: What factors should I consider to determine whether Al Qaeda has nuclear weapons?
A: Characteristics associated with possession of nuclear weapons and current evidence that it has nuclear weapons.
Therefore I need to solve two subproblems:
Assess the possibility that Al Qaeda might have nuclear weapons based on the characteristics associated with the possession of nuclear weapons.
Assess the current evidence that Al Qaeda has nuclear weapons.
Q: What are the characteristics associated with possession of nuclear weapons?
A: Reasons, desire, and ability to obtain nuclear weapons.
Therefore I need to solve three sub-problems:
Assess whether Al Qaeda has reasons to obtain nuclear weapons.
Assess whether Al Qaeda has desire to obtain nuclear weapons.
Assess whether Al Qaeda has the ability to obtain nuclear weapons.

In this way, the initial problem is successively reduced to simpler and simpler problems which are shown with a blue background in Fig. 2. Then the solutions of the

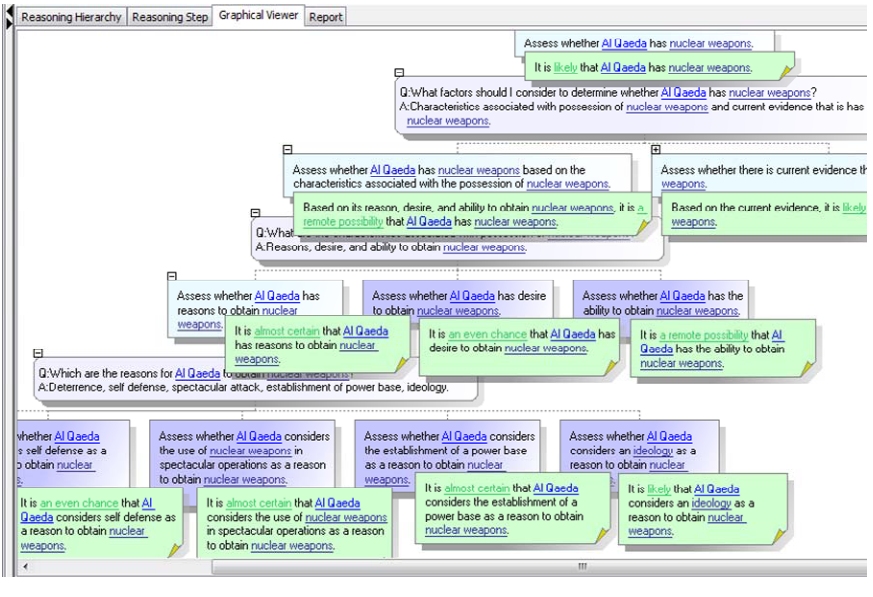


Fig. 2. Hypothesis analysis through problem reduction and solution synthesis

simplest problems are found, and these solutions (which are shown with a green background) are successively composed, from bottom up, until the solution of the initial problem is obtained (i.e. "It is likely that Al Qaeda has nuclear weapons.").

The intelligence analysts who have used our system, evaluated this type of reasoning as being very appropriate for teaching new analysts because it is very explicit and natural. However, the reasoning trees generated by the agent for real-world problems are very large. For example, Fig. 2 shows only the top 23 nodes of a tree that has 1,758 nodes. The question is how to systematically teach new analysts based on such complex trees. Our solution is described in the next sections.

4 Abstraction of Reasoning Trees

Although the reasoning trees generated by the agent are very large, its parts are repeated applications of a few abstract reasoning strategies. This is illustrated in Fig. 3 where the blue-bordered subtrees from the left-hand side are concrete applications of the abstract reduction strategy shown with a red-border in the right-hand side. Indeed, each of the blue subtrees represents the following abstract strategy:

> *In order to assess to what extent a certain piece of evidence (e.g. EVD-Reuters-01-01c, a fragment from a Reuters News Agency report) favors a certain hypothesis (e.g. "Al Qaeda desires to obtain nuclear weapons."), one has to solve two subproblems: 1) Assess to what extent that piece of evidence favors that hypothesis, assuming that the piece of evidence is believable, and 2) Assess the believability of that piece of evidence.*

There are other abstract strategies for analyzing the believability of direct testimonial evidence, or the believability of testimonial evidence obtained at second hand, or the credibility of tangible evidence, or the competence and credibility of primary or intermediary sources of information, etc. [13].

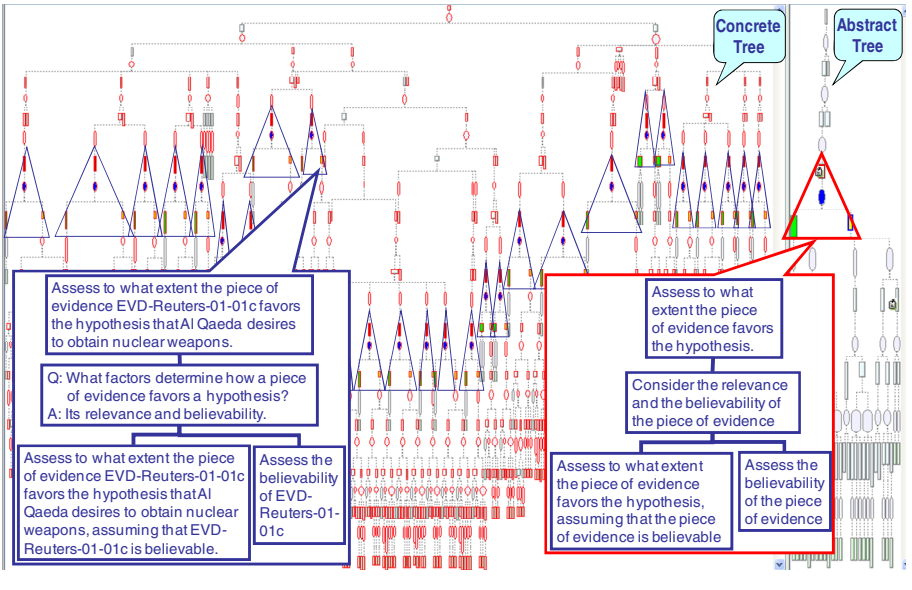


Fig. 3. Concrete and abstract reasoning trees

Our LTAS includes an abstraction learning module that allows the agent to learn abstraction rules from a subject matter expert. In essence, the expert abstracts a concrete reasoning subtree CT (such as the one from the bottom left of Fig. 3) into an abstract subtree AT (i.e. the one from the bottom right of Fig. 3). From this example of abstraction (CT → AT) the system learns a general abstraction rule AR. AR incorporates the complex reasoning rule(s) R that generated the concrete subtree CT. The abstraction rule AR allows the system to automatically identify concrete applications CT_i of the abstract reasoning strategy AT in a concrete reasoning tree. In all, there are only 22 abstract reduction strategies and 22 abstract synthesis strategies that are repeatedly applied to generate the large reasoning tree for solving the problem Assess whether Al Qaeda has nuclear weapons. As a consequence there are only 217 abstract nodes in the abstract reasoning tree that correspond to 1758 nodes in the concrete tree.

In conclusion, *the abstraction process helps to identify the problem solving strategies based on which complex reasoning trees are generated. Therefore, an approach to teach new professionals how to solve problems is to teach them the abstract reasoning strategies of their domain, illustrating them with concrete examples,* as will be discussed in the next section.

5 Lesson Design and Generation

Each basic abstract strategy, or a small set of related ones, is the foundation of a lesson designed with our LTAS. For example, Fig. 4 shows a fragment of the lesson that teaches an analyst how to assess the support provided by a piece of evidence to a hypothesis. The top part of Fig. 4 teaches the analyst how to solve this problem at the abstract level and the bottom part illustrates this abstract reasoning with examples generated from the domain knowledge base. The abstract reasoning teaches four basic abstract strategies but only the top strategy and one of the three bottom strategies are visible in Fig. 3. Each of the three bottom strategies shows an alternative way of assessing the extent to which the information provided by a piece of evidence is believable, depending on the type of evidence (i.e. direct testimonial evidence, testimonial evidence obtained at second hand, or testimonial evidence about tangible evidence). The tutor fades out the two strategies which are not illustrated by the example from the bottom part of Fig. 4 (see the right hand side of Fig. 4). A student can request additional examples that illustrate the other strategies, or could directly select them from a list. He or she could also click on the blue hyperlinks to receive brief or detailed definitions or even entire presentations on important concepts such as believability or objectivity [13]. Some lessons may display these descriptions automatically, as part of the lesson's flow, which may also use spoken text.

Fig. 4 illustrates only the first part of the lesson which teaches the reduction strategies for assessing the support provided by a piece of evidence to a hypothesis. The second part of the lesson teaches the corresponding synthesis strategies.

LTAS allows an instructional designer to create lessons and organize them into a curriculum, by using a set of drag and drop operations to specify the basic abstract strategies to be included into the lesson, the order in which they will be taught, as well as additional explanations and definitions. The result is a lesson script that is executed by the lesson generation module to create a lesson like that illustrated in Fig. 4.

Notice that the instructional designer does not need to specify the lesson's examples because they are automatically generated from the domain knowledge base. This significantly speeds up the lesson design process. Moreover, this provides a high level of generality to the developed curriculum and offers a solution to the customized tutoring of students who have different domain knowledge and interests. Indeed, a student may select a specific domain knowledge base of interest (e.g. for drug trafficking or crime investigation, instead of nuclear proliferation), and the tutoring system will generate examples from it, without any change in the lessons.

There are additional ways in which the tutoring is automatically customized. For instance, the lessons will only teach the reasoning strategies that can be illustrated in the selected domain knowledge base, and the test questions for assessing a student will also be generated from that knowledge base, as discussed in the next section.

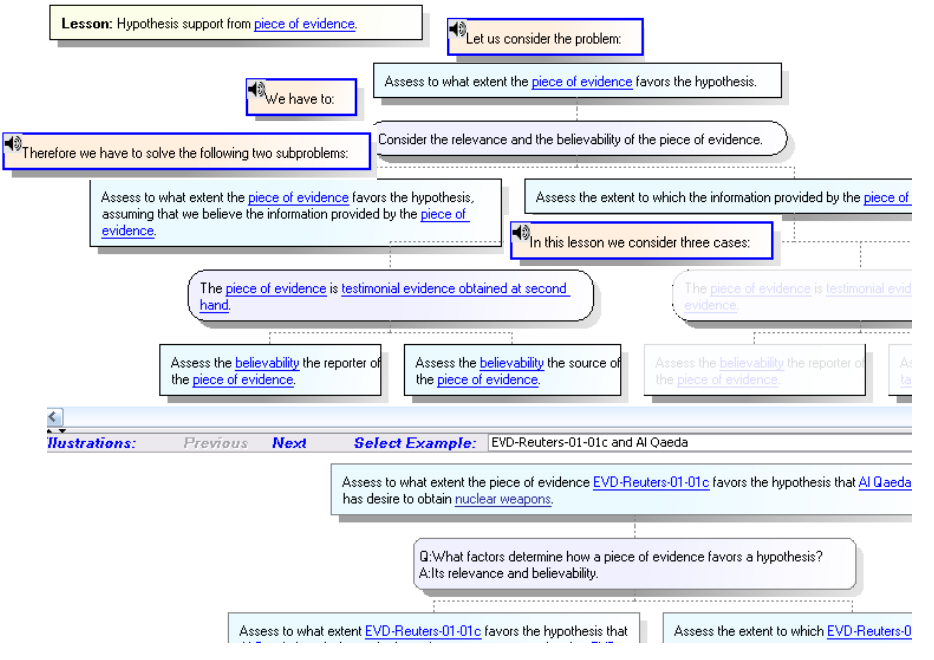


Fig. 4. The top-left part of a lesson interface

6 Learning and Generation of Test Questions

LTAS includes a module that allows the instructional designer to rapidly teach the agent how to automatically generate several types of test questions to assess students' problem solving knowledge. An example of generated test question is shown in Fig. 5 where a red-bordered problem is reduced to two red-bordered subproblems (see bottom of Fig. 5), in the context of a larger reasoning tree. The student is asked whether this reasoning is complete (i.e. includes all the necessary subproblems of the reduced problem), or incomplete (misses some subproblems but the present ones are correct),

or incorrect (includes incorrect subproblems), by clicking on the corresponding answer in the upper right-hand of Fig. 5. The student will receive an appropriate feedback confirming the answer or explaining the mistake. He or she may also request a hint and, in the case of self-testing mode (as opposed to assessment), may review the corresponding lesson by clicking on the "Go To Lesson" button.

The test question in Fig. 5 is of type omission, because one of the subproblems was omitted, and tests the student at the knowledge level [14]. A test of type modification (where no subproblems are omitted, but some of them have been modified) tests the student at the comprehension level. Finally, a test of type construction (where the student selects the correct subproblems of a given problem, from a larger list of potential subproblems) tests the student at the analysis level.

To teach the agent how to generate a class of test questions, the instructional designer selects a reduction step from a reasoning tree generated by the agent. This step, represented by a problem and its subproblems, is an example E of a previously learned problem reduction rule R from the domain knowledge base. Then the instructional designer changes the reduction step E, either by deleting some subproblems, or by modifying them, or by adding additional deliberately incorrect subproblems, to create an omission, modification, or construction test example, respectively. The instructional designer also provides a specific hint for the test example, as well as specific feedback for each of the possible student answers (i.e. correct, incomplete, and incorrect). The result is a specific test example TE which is an extension and modification of the reduction example E. By performing corresponding extensions and modifications of the general rule R (which generated E), the agent learns the general test question rule TR. TR can generate a class of test questions similar to TE, based on the current domain knowledge base.

The lesson scripts and the test generation rules are stored in the pedagogical knowledge base which also represents the student model. The model stores specific information about a particular student, such as the lessons taken and his or her

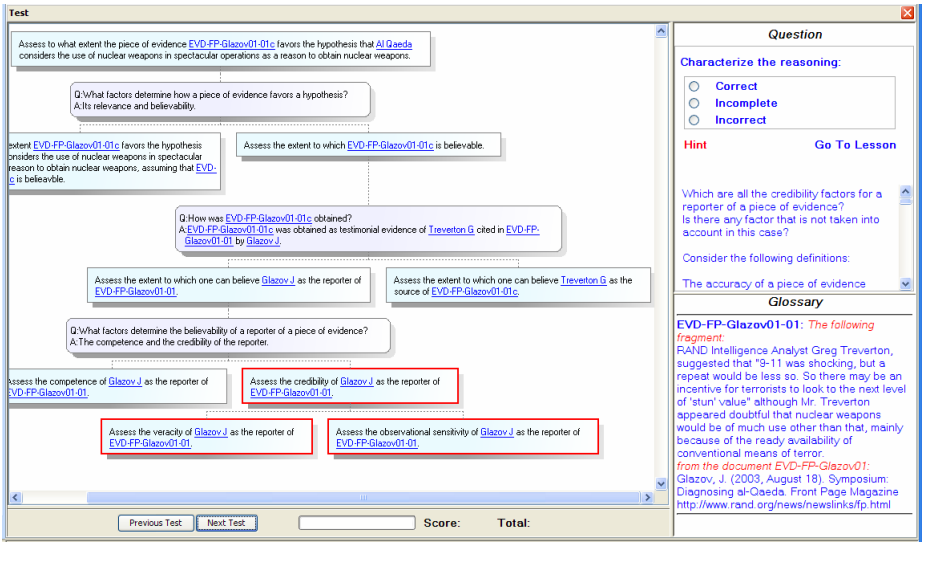


Fig. 5. Sample test question

performance on the test questions. It informs the test generation module to only generate tests related to the lessons taken by the student, as well as additional tests similar with those failed by the student.

7 Experimentation

The Disciple-LTA cognitive assistant developed with the LTAS tool has been used and evaluated in several courses at the US Army War College and at George Mason University [9]. The Army War College students were high ranking military officers that were either experienced intelligence analysts or users of intelligence. In contrast, George Mason students were computer science graduate students with no significant intelligence analysis experience. In both cases the students followed the lessons defined in the system and then were assessed based on the test questions generated by the system. As expected, the system was perceived as more useful by the novice analysts. However, even the expert analysts from the Army War College considered that the system was useful in teaching them a rigorous systematic approach for the "ill-defined" domain of intelligence analysis. Both before and after taking the lessons, the students from George Mason University (i.e. the novice analysts) were asked to subjectively assess their knowledge of several topics taught, on a 6-point scale, from none to very high. The results showed that the students considered that their post-lessons knowledge of the application domain was much better than their pre-lessons knowledge. Moreover, their self-assessed post-lesson knowledge was confirmed by the good results obtained by the students at the tests questions generated by the agent.

8 Summary of Contributions, Limitations and Future Research

While significant results have been demonstrated by many ITS authoring systems (e.g. [3], [4], which have advanced machine learning and authoring capabilities), most of the work has been done in the context of "well-defined" domains, such as physics, chemistry or mathematics [1], [3-5].

In this paper we have presented an overview of our research on a new approach to the development of systems for tutoring expert problem solving knowledge in an "ill-defined" domain [5]. This approach is based on methods from the areas of expert systems, machine learning, and intelligent tutoring systems, which we have developed over many years, and have integrated into a new type of tool, called learning and tutoring agent shell or LTAS (see Fig. 1). This tool allows rapid development of a new type of cognitive assistant that can be taught by subject matter experts and can then teach new professionals, as well as support them in problem solving and decision-making. LTAS is applicable to a wide range of ill-defined domains, due to its use of the general problem-reduction/solution-synthesis approach to problem solving. It also allows rapid acquisition of an expert's problem solving knowledge, based on powerful mixed-initiative, multistrategy learning methods that integrate learning from examples, learning from explanations, learning by analogy, and learning by abstraction. These capabilities have been used to develop problem solving and learning agents for complex problems, such as military course of action critiquing [15], military center of gravity analysis [8], and emergency response planning [16].

The tutoring capabilities of LTAS, introduced in the previous sections, have been only recently developed and rely heavily on its problem solving and learning capabilities. They include the capability to rapidly acquire the basic abstract problem solving strategies of the application domain, directly from a subject matter expert. They allow an instructional designer to rapidly design lessons for teaching the abstract problem solving strategies, without the need of defining examples because they are automatically generated by the system from the domain knowledge base. They also allow rapid learning of test questions. These capabilities confer a high degree of generality to the tutoring system that can be applied to several related application domains (such as, nuclear proliferation, drug trafficking, crime investigation, or law) with no change to the lessons.

REDEEM [2] is an authoring environment that allows classroom teachers to easily create a simple ITS by importing a computer-based training course as the domain content and by selecting among a wide variety of strategies for teaching it. While also focusing on instructional delivery (rather than coached problem solving) the teacher customizations allowed by the current version of Disciple-LTA are not as varied and as easy to perform, but they involve a much more complex ITS. Also, Disciple-LTA allows the students themselves to customize the lessons, by selecting not only the lesson's examples that illustrate the taught problem solving strategies, but even the domain knowledge base, to better fit their interests and knowledge.

Using LTAS we have developed a complex cognitive assistant for intelligence analysts which has been successfully used in several courses with both expert analysts and novice analysts.

Tutoring problem solving expertise is not only important for teaching new professionals, but also for teaching any person who desires to use a complex problem solving and decision-support assistant. Indeed, such an assistant will generate complex reasoning trees, as shown in Fig. 2 and Fig. 3, which the user needs to understand, browse, modify and extend. This makes tutoring a necessary capability of any complex assistant. On the other hand, as demonstrated by our research, existing problem solving and learning capabilities greatly facilitate the development of the tutoring capabilities. We therefore consider that *the proposed type of cognitive assistant, capable of learning, problem solving and tutoring, as well as the learning and tutoring agent shell used to build it, represent a very promising and expected evolution for the knowledge-based agents.*

The developed tutoring-related methods and their implementation in the current version of LTAS have several limitations which point to future research directions. For example, the lesson design module requires that a lesson should first introduce an abstract strategy and then illustrate it with examples. It is easy to extend this module to allow an instructional designer to define other types of lesson organizations, such as introducing first examples and then their abstraction, as in REDEEM [2].

The types of test questions currently learned and generated by the system are not very complex and diverse, each test question being based on a single problem solving rule from the domain knowledge base. It would not be very difficult to learn more complex test questions that are based on several related reasoning rules. It is also necessary to imagine new and more challenging types of test questions.

The current student model is quite limited and more research is needed both to develop a more complex model, and to more effectively use it in tutoring. In addition, more work is needed to significantly improve the interaction with the student.

References

1. Murray, T.: An overview of intelligent tutoring system authoring tools: Updated analysis of. In: Murray, T., Blessing, S., Ainsworth, S.E. (eds.) Tools for Advanced Technology Learning Environments, pp. 491–544. Kluwer, Amsterdam (2003)
2. Ainsworth, S.E., Fleming, P.F.: Teachers as instructional designers: Does involving a classroom teacher in the design of computer-based learning environments improve their effectiveness? In: Gerjets, P., Kirschner, P.A., Elen, J., Joiner, R. (eds.) Instructional design for effective and enjoyable computer-supported learning. Proceedings of the first joint meeting of the EARLI SIGs Instructional Design and Learning and Instruction with Computers, pp. 283–291 (2004)
3. Aleven, V., McLaren, B.M., Sewall, J., Koedinger, K.: The Cognitive Tutor Authoring Tools (CTAT): Preliminary evaluation of efficiency gains. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 61–70. Springer, Heidelberg (2006)
4. Munro, A., Johnson, M.C., Pizzini, Q.A., Surmon, D.S., Towne, D.M., Wogulis, J.L.: Authoring Simulation-Centered Tutors with RIDES. International Journal of Artificial Intelligence in Education 8, 284–316 (1997)
5. Lynch, C., Ashley, K., Aleven, V., Pinkwart, N.: Defining Ill-Defined Domains; A literature survey. In: Aleven, V., Ashley, K., Lynch, C., Pinkwart, N. (eds.) Proceedings of the Workshop on Intelligent Tutoring Systems for Ill-Defined Domains at the 8th Int. Conference on Intelligent Tutoring Systems, Jhongli (Taiwan), pp. 1–10 (2006)
6. Tecuci, G., Boicu, M., Marcu, D., Boicu, C., Barbulescu, M., Ayers, C., Cammons, C.: Cognitive Assistants for Analysts. Journal of Intelligence Community Research and Development (2007)
7. Tecuci, G.: Building Intelligent Agents: An Apprenticeship Multistrategy Learning Theory, Methodology, Tool and Case Studies. Academic Press, London (1998)
8. Tecuci, G., Boicu, M., Boicu, C., Marcu, D., Stanescu, B., Barbulescu, M.: The Disciple-RKF Learning and Reasoning Agent. Computational Intelligence 21, 462–479 (2005)
9. Le, V.: Abstraction of Reasoning for Problem Solving and Tutoring Assistants. Ph.D. Dissertation in Information Technology. Learning Agents Center, Volgenau School of IT&E, George Mason University(Spring 2008)
10. Clancey, W.J.: NEOMYCIN: Reconfiguring a rule-based system with application to teaching. In: Clancey, W.J., Shortliffe, E.H. (eds.) Readings in Medical Artificial Intelligence, pp. 361–381. Addison-Wesley, Reading (1984)
11. Nilsson, N.J.: Problem Solving Methods in Artificial Intelligence. McGraw-Hill (1971)
12. Clancey, W.J.: Knowledge-Based Tutoring: The GUIDON Program. MIT Press (1987)
13. Schum, D.A.: The Evidential Foundations of Probabilistic Reasoning. Northwestern University Press (2001)
14. Bloom, B.S.: Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain. David McKay Co Inc., New York (1956)
15. Tecuci, G., Boicu, M., Bowman, M., Marcu, D., Burke, M.: An Innovative Application from the DARPA Knowledge Bases Programs: Rapid Development of a High Performance Knowledge Base for Course of Action Critiquing. AI Magazine 22(2), 43–61 (2001)
16. Tecuci, G., Boicu, M., Marcu, D., Barbulescu, M., Boicu, C., Le, V., Hajduk, T.: Teaching Virtual Experts for Multi-Domain Collaborative Planning. J. of Software 3, 38–59 (2008)

Balancing Cognitive and Motivational Scaffolding in Tutorial Dialogue

Kristy Elizabeth Boyer[1], Robert Phillips[1,2], Michael Wallis[1,2], Mladen Vouk[1], and James Lester[1]

[1] Department of Computer Science, North Carolina State University
[2] Applied Research Associates, Inc.
Raleigh, North Carolina, USA
{keboyer,rphilli,mdwallis,vouk,lester}@ncsu.edu

Abstract. A key challenge in the design of tutorial dialogue systems is identifying tutorial strategies that can effectively balance the tradeoffs between cognitive and affective student outcomes. This balance is problematic because the precise nature of the interdependence between cognitive and affective strategies is not well understood. Furthermore, previous studies suggest that some cognitive and motivational goals are at odds with one another because a tutorial strategy designed to maximize one may negatively impact the other. This paper reports on a tutorial dialogue study that investigates motivational strategies and cognitive feedback. It was found that the choice of corrective tutorial strategy makes a significant difference in the outcomes of both student learning gains and self-efficacy gains.

1 Introduction

Recent years have seen the emergence of a broader view of learning as a complex process involving both cognitive and affective states. To empirically explore these issues, a number of intelligent tutoring systems (ITSs) (e.g., AutoTutor [1], Betty's Brain [2], ITSpoke [3], M-Ecolab [4], and MORE [5]) are being used as platforms to investigate the impact of tutorial interactions on affective and motivational outcomes (e.g., self-efficacy) along with purely cognitive measures (i.e., learning gains). A central problem in this line of investigation is identifying tutorial strategies (e.g., [6]) that can appropriately balance the tradeoffs between cognitive and affective student outcomes [7]. While a rich set of cognitive and affective tutorial strategies is emerging (e.g., [8]), the precise nature of the interdependence between these types of strategies is not well understood. The extent to which each type of strategy, and specific instances of it in certain contexts, may be used to enhance tutorial effectiveness is an important question to designers of ITSs.

This paper reports on an empirical study to compare the impact of certain cognitive and motivational tutorial strategies on student learning and self-efficacy in human-human tutoring. Specifically, we consider the motivational strategies of *praise* and *reassurance* [7] and the category of informational tutorial utterances termed *cognitive feedback* [2, 8]. Following the approach of Forbes-Riley and colleagues [3, 9],

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 239–249, 2008.

utterances from a corpus of human-human tutorial dialogues are annotated with dialogue acts. Then, adopting the approach proposed by Ohlsson *et al.* [10], statistical modeling techniques are employed to quantify the relative impact of these different tutorial strategies on the outcomes of interest (in this case, learning and self-efficacy gains). By mining a corpus of human-human tutorial dialogues for naturally-occurring corrective strategies (i.e., tutorial moves in response to plausibly incorrect student problem-solving actions), we induce tutorial dialogue strategies that embody the regularities of effective tutorial dialogue across multiple tutoring sessions.

A key finding of the study is that the choice of corrective tutorial strategy has a significant impact on both the learning gains and the self-efficacy gains of students. The results reinforce related findings (e.g., [2, 7, 11]) that suggest some cognitive and motivational goals are at odds with one other because a tutorial strategy designed to maximize one set of goals (e.g., cognitive goals) can negatively impact the other. However, the study reported here also found that a strategy that provides students with positive cognitive feedback as a corrective approach can strike a "delicate balance" [1] and achieve desirable motivational and cognitive outcomes.

2 Related Work

Much of the research on motivation conducted in the ITS community is theoretically grounded in frameworks developed in the cognitive science community over the past two decades (e.g., [12, 13, 14]). Chief among these results is Keller's theory that student motivation plays a key role in the learning process [12]. This view is seconded by Lepper [7], who states that the most effective tutors give equal attention to both the motivational and cognitive concerns of students. Lepper *et al.* [7] refine Keller's model by postulating that motivation is comprised of confidence, challenge, control, and curiosity. Lepper [7] further identifies the two strategies of *praise* and *reassurance* as direct means of bolstering student confidence. These strategies are a form of "verbal persuasion," also identified by Bandura [15], as one way of increasing self-efficacy.

An increasingly active area of investigation is the search for tutorial strategies that address the complementary cognitive and affective concerns that shape the tutoring process [16]. Porayska-Pomsta and Pain [8] use dialogue analysis to classify cognitive and affective feedback[1] in terms of the degree to which each addresses a student's need for both autonomy and approval. Forbes-Riley and Litman (e.g., [18]) employ bigram analysis at the dialogue act level to extract tutorial strategies for responding to student uncertainty. Corpus analysis techniques have also informed work by Marineau *et al.* [9] on the classification of tutorial acts, as well as work by Rosé *et al.* (e.g., [19]) and Ohlsson *et al.* [10] on modeling the effectiveness of tutorial strategies.

Developing a clear understanding of the tradeoffs between cognitive and affective feedback is an important next step in tutorial dialogue research. Prior investigations

[1] We use *feedback* to refer to "information communicated to the learner that is intended to modify the learner's thinking or behavior for the purpose of improving learning" [17].

of tutorial feedback have established a foundational understanding of cognitive feedback in terms of how and when it is delivered (e.g., [20]). Jackson and Graesser [1] found the presence of cognitive feedback, as opposed to motivational "progress" feedback, was responsible for higher learning gains in experimental versions of AutoTutor; on the other hand, the presence of cognitive feedback lowered students' motivational ratings. A consistent finding observed by Tan and Biswas [2] was that students working with modified versions of Betty's Brain were able to learn better when given cognitive rather than affective feedback. Kelly and Weibelzahl [21] investigated a motivational strategy in which a student was progressively shown more of a hidden image after each successful step through the learning task. Students in the motivational treatment group showed larger increases in confidence levels compared with those in the control group, while there was no significant difference in learning gain. Finally, Wang *et al.* [22] found that tutors who gave polite feedback facilitated higher student self-efficacy gains, while learning was nearly unaffected.

Beyond these broadly observable tradeoffs, investigators have also found that tutorial strategies may impact student subgroups (e.g., low ability vs. high ability students) in different ways. Rebolledo-Mendez *et al.* [4] explored the effect of enhancing a tutoring system with motivational scaffolding. In M-Ecolab, initially unmotivated students were found to perform better with motivational adaptation and feedback, while students who were already motivated did not benefit from the motivational support. In a study of perceived politeness (a motivational aspect of tutorial utterances), Mayer *et al.* [23] found students who were experienced with computers were less bothered by direct commands from a machine, while inexperienced students were more apt to appreciate politeness.

3 Corpus Study

To determine the effect of specific tutorial strategies on learning and self-efficacy, a human-human tutoring corpus study was conducted. With a focus on tutorial strategies for addressing questionable student problem-solving actions, the study investigated student-tutor interactions in the domain of introductory computer science. The corpus consists of three types of events: tutor utterances, student utterances, and student problem-solving actions (in this case, programming events in which students create statements in Java programs).

3.1 Experimental Design

Subjects. Forty-three volunteers from a university-level introductory computer programming class attended a single tutoring session each. Subjects were not compensated for their participation; the indirect reward was that participants were not required to attend their weekly computer science laboratory class because they fulfilled the attendance requirement through study participation.

Procedure. At the beginning of each tutoring session, subjects completed a questionnaire containing items designed to gauge self-efficacy as it relates to completing the

computer science task [24]. Subjects also completed a pre-test that measured conceptual knowledge related to the learning task. The tutor and student were in separate rooms during the 55-minute tutoring session, working through an extended version of a software package developed to facilitate remote collaborative programming [25]. This software allowed tutors to observe student problem-solving actions in real time while carrying on conversations through a textual dialogue interface. Students were not aware of any tutor characteristics (e.g., name, gender). Upon completion of the session, students filled out a post-questionnaire and a post-test containing questions that were analogues to the pre-tutoring versions.

Tutors. Fourteen volunteer tutors were paired blindly with students based solely on scheduling availability. Tutors were in separate rooms from students and were not made aware of any student characteristics (e.g., self-efficacy rating, gender, pre-test score). All tutors were themselves students in a Department of Computer Science. Two were advanced undergraduates, and the remaining twelve were graduate students. All fourteen tutors reported experience as a peer tutor, and ten tutors had also served as teaching assistants in a university computing course for one or more semesters Of these ten, three had also served as primary instructors in a university-level introductory programming course. Four of the tutors were female. While these and other tutor characteristics may be useful for predicting what tutors *will do* (and perhaps even *why*), this paper begins with what tutors *did do* and goes on to draw conclusions on the relative effectiveness of various strategies [10].

3.2 Corpus Characteristics

The raw corpus contains 4,864 dialogue moves: 1,528 student utterances and 3,336 tutor utterances. As a chronology of tutorial dialogue interleaved with student problem-solving (programming) actions that took place during the tutoring sessions, the corpus contains 29,996 programming keystrokes and 1,277 periods of scrolling – all performed by students. Other problem-solving actions, such as opening and closing files or running the program, were sparse and were therefore omitted here.

Of the 3,336 tutor utterances, 1,243 occur directly after "questionable" student problem-solving action. (The notion of "questionable" is defined below.) This subset of tutorial utterances serves as the basis for the tutorial strategy comparison.

3.3 Problem-Solving Act Tagging

Student problem-solving actions were logged throughout the tutoring sessions. The two actions under consideration for this analysis are: typing in the programming interface and scrolling in the program editor window. To interpret the raw logged student problem-solving actions, these events were automatically tagged using a heuristic measure for correctness. This heuristic represents just a first step toward automatically classifying student actions in the problem-solving environment: if a programming keystroke (character) survived until the end of the session, this event was tagged *promising*. This heuristic is based on the observation that the subtasks in this learning

task were accomplished in a linear fashion, with tutors not allowing students to move forward until the previously implemented steps were judged to be correct. Conversely, if a programming keystroke (character) did not survive until the end of the session, the problem-solving act was tagged *questionable*. The rationale for this rule is that non-surviving characters were subsequently displaced or removed for some reason, meaning they were plausibly incorrect to start with. Finally, periods of consecutive scrolling were marked *questionable* because in this context, scrolling was almost uniformly undertaken by a student who was confused and looking for answers in irrelevant task scaffolding.

3.4 Dialogue Act Annotation

Because utterances communicate through two orthogonal channels, a cognitive channel and a motivational channel, each utterance was annotated with both a cognitive dialogue tag and a motivational dialogue tag. The dialogue act tag set, which consists of sixteen cognitive acts plus six motivational/affective acts, is an extension of the tag set presented in [26]. Table 1 displays the subset of this dialogue act tagging scheme relevant to the current study.

The entire corpus was tagged by a single human annotator, with a second tagger marking 1,418 of the original 4,864 utterances. The resulting kappa statistics were 0.76 in the cognitive channel and 0.64 in the motivational/affect channel.[2]

4 Analysis and Results

Overall, the tutoring sessions were effective: they yielded learning gains (mean 5.9%, median 7.9%), which were statistically significant (p=0.038), and they produced self-efficacy gains (mean 12.1%, median 12.5%), which were also statistically significant (p<0.0001).

Analyses revealed that statistically significant relationships hold between tutorial strategy and learning, as well as between tutorial strategy and self-efficacy gains. First, the values of learning gain and self-efficacy gain were grouped into binary categories ("Low", "High") based on the median value. Multiple logistic regression was then applied with the gain category as the predicted value and tutorial strategy, incoming self-efficacy rating, and pre-test score as predictors.[3] Multiple logistic regression was chosen over multiple linear regression because the learning instruments (10 items each) yielded few distinct values of learning gain. Logistic regression computes the odds of a particular outcome over another (e.g., "Having high learning gain versus low learning gain") given one value of the predictor variable over another (e.g., "The tutorial strategy was positive feedback instead of praise").

[2] This kappa was computed on all student and tutor utterances using the full tagging scheme.

[3] To control for the possibility that student outcomes were predicted entirely by incoming student characteristics rather than by any tutorial action, pre-test score and incoming self-efficacy rating were treated as predictors in all models.

Table 1. Relevant Tutorial Dialogue Acts

	Dialogue Act	Description	Example	Frequency*
Cognitive	*Positive* Cognitive Feedback	Indicates student correctness	"Right."	184
	Lukewarm Cognitive Feedback	Indicates partial student correctness	"Sort of." "Almost."	40
	Negative Cognitive Feedback	Indicates complete lack of student correctness	"That's not right."	24
	Neutral Cognitive Feedback (Tutorial Statement)	Informational statement containing no *explicit* indication of student correctness	"That variable should be declared outside the loop."	828
	Question	Question regarding task or general subject knowledge	"How many elements does the array need to hold?"	101
Motivational	Praise	Motivational statement intended to emphasize successes	"Great job!" "That's perfect."	89
	Reassurance	Motivational statement intended to minimize failure	"Don't worry about it." "That was a tricky part."	15
	Neutral	Statement with no clear motivational component.		1090

** The frequency column indicates a simple frequency out of 1,243 tutor utterances that followed questionable student problem-solving actions in the corpus. These dialogue acts are not mutually exclusive and are not exhaustive since this table displays only a subset of all dialogue acts.*

4.1 Presence of Tutorial Encouragement

We first consider two categories of corrective tutorial utterances: those *with* and those *without* explicit encouragement (i.e., praise or reassurance). Both these categories may, but need not, contain cognitive feedback components. (We restrict the analysis to only cognitive feedback in the next subsection, and later omit all such feedback to consider standalone tutorial encouragement.) A logistic regression model quantified the significant relationships between tutorial encouragement and learning gain, revealing that after accounting for the effects of pre-test score and incoming self-efficacy rating (both of which were significant in the model with $p<0.001$), observations containing tutorial encouragement were 56%[4] less likely to result in high learning gain than observations without explicit tutorial encouragement ($p=0.001$). On the other hand, tutorial encouragement was weakly linked to self-efficacy gains, with explicit encouragement being 57% more likely to result in high self-efficacy gain than tutorial responses that had no explicit praise or reassurance ($p=0.054$). These

[4] This value and its counterparts throughout the paper represent logistic regression point estimates of odds ratio (analogous to the regression coefficient in multiple linear regression). The accompanying p-value indicates the level at which the predictor variable was significant in the model.

models suggest that *the presence of tutorial encouragement in response to questionable student problem-solving action is weakly linked to self-efficacy gain but may detract from learning gain.*

4.2 Adding Encouragement to Cognitive Feedback

We now consider only corrective tutorial acts that were tagged as cognitive feedback and compare the relative impact of those *with* and *without* explicit tutorial praise or reassurance. Because the co-occurrence of cognitive feedback with reassurance was very low (n=2), we omit this strategy from consideration and compare the two strategies of *purely cognitive feedback* and *cognitive feedback plus praise.* A logistic regression model built as described above revealed that observations in which the tutor used cognitive feedback plus praise were associated with 40% lower likelihood of high learning gain than observations in which the tutor used purely cognitive feedback. No impact was observed on self-efficacy gain. These results suggest that in response to questionable student problem-solving action, *to achieve learning gains*, *purely cognitive feedback is preferred over cognitive feedback plus praise, while self-efficacy gain does not appear to be impacted either way*.

4.3 Standalone Tutorial Encouragement

In this corpus, tutorial encouragement is sometimes encountered with no cognitive feedback component; that is, the tutorial utterance is in no way aimed at giving substantive task-related feedback, but instead, is aimed at the student's motivational or affective state through explicit praise or reassurance. We now consider this tutorial strategy of *standalone motivational acts*. Unlike the previous results that had a consistent (or no statistically significant) impact on student sub-groups and were therefore reported only for the general student population, purely motivational statements appear to affect low and high self-efficacy students differently. A separate logistic regression was run for the low initial self-efficacy and high initial self-efficacy student groups. Among students with low incoming self-efficacy, observations in which the tutor employed a standalone motivational act were 300% as likely to be in the high self-efficacy gain group as observations in which the tutor employed a purely cognitive statement or a cognitive statement combined with encouragement (p=0.039). In contrast, among students with high initial self-efficacy, a purely motivational tactic resulted in 90% lower odds of being in the high self-efficacy gain group. Standalone motivational acts showed no statistically different impact on learning gain compared to other tutorial acts (p=0.268). This relationship held for both the low self-efficacy (p=0.216) and high self-efficacy subgroups (p=0.441) with regard to impact on learning gain. These results suggest that *standalone praise or reassurance may be useful for increasing self-efficacy gain among low initial self-efficacy students, but may decrease self-efficacy gain in high initial self-efficacy students*. In addition, *standalone praise or reassurance does not appear helpful for learning gains.*

4.4 Superiority of Positive Cognitive Feedback

We have seen evidence thus far that explicit tutor encouragement in the form of praise or reassurance has mixed effects on learning and self-efficacy gains. We now consider the class of purely cognitive tutorial moves, i.e., all tutorial acts that have no explicit

encouragement attached. As presented in Section 3.4, the strategies under consideration here are *positive*, *lukewarm*, *negative*, and *neutral* cognitive feedback plus *tutorial questions*. Because positive cognitive feedback related similarly to each of the other types of cognitive moves, we forego pairwise comparisons and instead contrast positive cognitive feedback against the group of all other purely cognitive strategies. Chi-square analysis reveals positive cognitive feedback had a significantly different impact on self-efficacy than other strategies (p=0.0028). A logistic regression refined the relationship, revealing positive feedback resulted in 190% increased odds of high student self-efficacy gain compared to the other cognitive strategies (p=0.0057). Positive cognitive feedback did not differ significantly from other types of cognitive strategies in a Chi-square comparison with respect to learning gains (p=0.390). The models thus suggest when dealing with questionable student problem-solving action, *positive cognitive feedback is preferable to other types of cognitive feedback for eliciting self-efficacy gains, but this type of feedback is not found to be better or worse than other cognitive feedback for effecting learning gains.*

5 Discussion

The study found that the presence of direct tutorial praise or encouragement in response to questionable student problem-solving action increased the odds that the student exhibited high self-efficacy gain, while lowering the odds of high learning gain. The study also found that purely cognitive feedback was preferable for learning gains compared to cognitive feedback with an explicitly motivational component. These empirical findings are consistent with theories of Lepper [7] who observed that some cognitive and affective goals in tutoring are "at odds." The results also echo quantitative results from other domains such as qualitative physics [1] and river ecosystems [2] that, in general, overt motivational feedback contributes to motivation but cognitive feedback matters more for learning.

In this study, standalone motivational utterances in response to questionable problem-solving action increased the likelihood of high self-efficacy gain among low incoming self-efficacy students. This motivational tactic, however, reduced the likelihood of self-efficacy gain in students with initially high self-efficacy. These results confirm empirical findings from the domain of ecology [4] in which only unmotivated students benefited from extra motivational scaffolding. While it is true that students with initially high confidence have less room for self-efficacy gain in the first place, it is also likely the case that students with high confidence or high ability may be less prone to need or appreciate motivational tactics such as politeness [23].

Of the tutorial strategies that occurred in the corpus, positive cognitive feedback emerged as an attractive approach for responding to plausibly incorrect student problem-solving actions. Responding positively (e.g., "Right") to questionable student actions is an example of indirect correction, which is recognized as a polite strategy (e.g., [8, 22]). As such, the positive feedback approach seems to have an implicit, yet perceptible, motivational component while retaining its usefulness as cognitive feedback. Qualitative inspection of the corpus indicates positive cognitive feedback in response to plausibly incorrect problem-solving actions was usually followed by neutral cognitive feedback that served to more informatively, yet indirectly, point out

student errors. Thus the benefits of positive cognitive feedback may also indicate (by proxy) the effectiveness of Lepper's indirect feedback acts [7].

6 Conclusions and Future Work

Balancing cognitive and motivational scaffolding has emerged as a key problem in tutorial dialogue. To investigate these issues, a corpus study of naturalistic human-human tutorial dialogue in the domain of computer science was conducted to determine the most effective use of motivation in the context of feedback for problematic student actions. The results suggest that positive cognitive feedback may prove an appropriate strategy for responding to questionable student problem-solving action in task-oriented tutorial situations because of its potential for addressing the sometimes competing cognitive and affective needs of students. For low self-efficacy students, however, it was found that direct standalone encouragement can be used to bolster self-efficacy, but the same standalone encouragement may not be helpful for high self-efficacy students.

In this work we have considered a limited set of motivational dialogue acts, namely praise and reassurance. Important future work will target an expanded set of affective dialogue acts to facilitate continued exploration of motivational and affective phenomena in this context. Also important will be expanding the window of consideration to pairs that include both student and tutor utterances, along with tuples that include three acts or more (e.g., problem-solving action, tutor utterance, student utterance) to model the effects of higher-level tutorial strategies. Finally, the current results reflect human-human tutoring strategies that proved to be effective; however, it remains to be seen whether these same strategies can be successfully employed in tutorial dialogue systems. Continuing to identify and empirically compare the effectiveness of alternative tutorial strategies will build a solid foundation for choosing tutorial strategies that balance the cognitive and affective concerns surrounding the complex processes of teaching and learning through tutoring.

Acknowledgments. The authors wish to thank Scott McQuiggan and the members of the Intellimedia Center for Intelligent Systems for their ongoing intellectual contributions, and the Realsearch Group at NC State University for extensive project development support. This work was supported in part by the National Science Foundation through Grant REC-0632450, an NSF Graduate Research Fellowship, and the STARS Alliance Grant CNS-0540523. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. Support was also provided by North Carolina State University through the Department of Computer Science and the Office of the Dean of the College of Engineering.

References

[1] Jackson, G.T., Graesser, A.C.: Content matters: An investigation of feedback categories within an ITS. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Proceedings of AIED 2007, vol. 158, pp. 127–134. IOS Press (2007)

[2] Tan, J., Biswas, G.: The role of feedback in preparation for future learning: A case study in learning by teaching environments. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 370–381. Springer, Heidelberg (2006)
[3] Forbes-Riley, K., Litman, D., Huettner, A., Ward, A.: Dialogue-learning correlations in spoken dialogue tutoring. In: Looi, C.-K., Mccalla, G., Bredeweg, B., Breuker, J. (eds.) Proceedings of AIED, pp. 225–232. IOS Press (2005)
[4] Rebolledo-Mendez, G., du Boulay, B., Luckin, R.: Motivating the learner: An empirical evaluation. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 545–554. Springer, Heidelberg (2006)
[5] del Soldato, T., du Boulay, B.: Implementation of motivational tactics in tutoring systems. Journal of Artificial Intelligence in Education 6(4), 337–378 (1995)
[6] Graesser, A.C., Person, N.K., Magliano., J.P.: Collaborative dialogue patterns in naturalistic one-to-one tutoring. Applied Cognitive Psychology 9(6), 495–522 (1995)
[7] Lepper, M.R., Woolverton, M., Mumme, D.L., Gurtner, J.: Motivational techniques of expert human tutors: Lessons for the design of computer-based tutors. In: Lajoie, S.P., Derry, S.J. (eds.) Computers as Cognitive Tools, pp. 75–105. Lawrence Erlbaum Associates, Inc., Hillsdale (1993)
[8] Porayska-Pomsta, K., Pain, H.: Providing cognitive and affective scaffolding through teaching strategies: Applying linguistic politeness to the educational context. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 77–86. Springer, Heidelberg (2004)
[9] Marineau, J., Wiemer-Hastings., P., Harter, D., Olde, B., Chipman, P., Karnavat, A., Pomeroy, V., Rajan, S., Graesser, A., Tutoring Research Group,: Classification of speech acts in tutorial dialog. In: Proceedings of the ITS 2000 Workshop on Modeling Human Teaching Tactics and Strategies, pp. 65–71 (2000)
[10] Ohlsson, S., Di Eugenio., B., Chow, B., Fossati, D., Lu, X., Kershaw., T.C.: Beyond the code-and-count analysis of tutoring dialogues. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Proceedings of AIED 2007, vol. 158, pp. 349–356. IOS Press (2007)
[11] Person, N.K., Kreuz, R.J., Zwaan, R.A., Graesser, A.C.: Pragmatics and pedagogy: Conversational rules and politeness strategies may inhibit effective tutoring. In: Cognition and Instruction, vol. 13(2), pp. 161–188. Lawrence Erlbaum Associates, Inc., Hillsdale (1995)
[12] Keller, J.M.: Motivational design of instruction. In: Reigeluth, C.M. (ed.) Instructional-Design Theories and Models: An Overview of Their Current Status, pp. 383–429. Lawrence Erlbaum Associates, Inc., Hillsdale (1983)
[13] Cameron, J., Pierce, W.D.: Reinforcement, reward, and intrinsic motivation: A meta-analysis. Review of Educational Research 64(3), 363–423 (1994)
[14] Deci, E.L., Koestner, R., Ryan, R.M.: Extrinsic rewards and intrinsic motivation in education: Reconsidered once again. Review of Educational Research 71(1), 1–27 (2001)
[15] Bandura, A.: Self-Efficacy: The Exercise of Control. W.H. Freeman and Company, New York (1997)
[16] Lehman, B., Matthews, M., D'Mello, S., Person, N.: What are you feeling? Investigating student affective states during expert human tutoring sessions. In: Proceedings of ITS 2008. Springer, Berlin (to appear, 2008) (in press)
[17] Shute, V.J.: Focus on formative feedback. Technical Report RR-07-11, Educational Testing Service, Princeton, NJ (2007)
[18] Forbes-Riley, K., Litman, D.: Using bigrams to identify relationships between student certainness states and tutor responses in a spoken dialogue corpus. In: Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue, Lisbon, Portugal (2005)

[19] Rosé, C.P., Bhembe, D., Siler, S., Srivastava, R., VanLehn, K.: The role of why questions in effective human tutoring. In: Hoppe, U., Verdejo, F., Kay, J. (eds.) Proceedings of AIED 2003, pp. 55–62. IOS Press (2003)

[20] Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A.: Intelligent tutoring goes to school in the big city. International Journal of Artificial Intelligence in Education 8, 30–43 (1997)

[21] Kelly, D., Weibelzahl, S.: Raising confidence levels using motivational contingency design techniques. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 535–544. Springer, Heidelberg (2006)

[22] Wang, N., Johnson, L., Rizzo, P., Shaw, E., Mayer, R.E.: Experimental evaluation of polite interaction tactics for pedagogical agents. In: Proceedings of the 10th International Conference on Intelligent User Interfaces, San Diego, pp. 12–19 (2005)

[23] Mayer, R.E., Johnson, W.L., Shaw, E., Sandhu, S.: Constructing computer-based tutors that are socially sensitive: Politeness in educational software. International Journal of Human-Computer Studies 64(1), 36–42 (2006)

[24] Bandura, A.: Guide for constructing self-efficacy scales. In: Urdan, T., Pajares, F. (eds.) Self-Efficacy Beliefs of Adolescents, pp. 307–337. Information Age Publishing, Greenwich (2006)

[25] Ho, C.-W., Raha, S., Gehringer, E., Williams, L.: Sangam – A Distributed pair programming plug-in for eclipse. In: Proceedings of the 2004 OOPSLA Workshop on Eclipse Technology eXchange, pp. 73–77. Association for Computing Machinery Press, New York (2004)

[26] Boyer, K.E., Vouk, M.A., Lester, J.C.: The influence of learner characteristics in task-oriented tutorial dialogue. In: Luckin, R., Koedinger, K., Greer, J. (eds.) Proceedings of AIED 2007, pp. 127–134. IOS Press (2007)

Assessing the Impact of Positive Feedback in Constraint-Based Tutors

Devon Barrow[1], Antonija Mitrovic[1], Stellan Ohlsson[2], and Michael Grimley[1]

[1] University of Canterbury, Private Bag 4800
Christchurch 8140, New Zealand
dba46@student.canterbury.ac.nz
{tanja.mitrovic,michael.grimley}@canterbury.ac.nz
[2] Department of Psychology, University of Illinois, Chicago
stellan@uic.edu

Abstract. Most existing Intelligent Tutoring Systems (ITSs) are built around cognitive learning theories, such as Ohlsson's theory of learning from performance errors and Anderson's ACT theories of skill acquisition, which focus primarily on providing negative feedback, facilitating learning by correcting errors. Research into the behavior of expert tutors suggest that experienced tutors use positive feedback quite extensively and successfully. This paper investigates positive feedback; learning by capturing and responding to correct behavior, supported by cognitive learning theories. Our aim is to develop and implement a systematic approach to delivering positive feedback in ITSs. We report on an evaluation study done in the context of SQL-Tutor, in which the control group used the original version of the system giving only negative feedback, while the experimental group received both negative and positive feedback. Results show that the experimental group students needed significantly less time to solve the same number of problems, in fewer attempts compared to those in the control group. Students in the experimental group also learn approximately the same number of concepts as students in the control group, but in much less time. This indicates that positive feedback facilitates learning and improves the effectiveness of learning in ITSs.

1 Introduction

Intelligent Tutoring Systems (ITSs) continue to grow in popularity and application. The use of computers to support education and learning has more than doubled between 1984 (36.2%) and 1997 (84%) [3]. ITSs are at the forefront of these technologies. There is however still considerable work to be done in making such systems more effective. Current ITSs provide one-on-one tutoring at relatively low cost and the added flexibility with regard timing, location and amount of the tutoring experience. Evaluation of a LISP tutor showed that students in the experimental group completed problems in one third the time of those under control conditions with improvement in learning of 1 standard deviation [4]. ANDES, a tutoring system for teaching Physics, improves performance by 0.9 standard deviations [5], while SQL-Tutor improves performance by 0.65 standard deviations in just two hours of interaction with the system [6].

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 250–259, 2008.

However, more research is still needed in order to develop a theory of tutoring which would link the various learning mechanisms to the types of information the learner needs and hence what the tutor (ITS) should provide. Current hypotheses propose that student learning is correlated with the tutor's use of knowledge-construction activities e.g. dynamic plan scaffolding, and refection/generalization techniques [5, 7]. There is a move to incorporate these techniques into ITSs through modeling of tutorial actions and strategies as observed with expert human tutoring [7]. Positive feedback is one of the teaching strategies which continue to surface throughout tutoring protocols. Work being done with tutoring protocols at the University of Illinois at Chicago [8] shows extensive and effective use of positive feedback by experienced human tutors. A logical step for ITSs therefore seems to be extending the proposed model of tutoring to incorporate positive feedback.

In this paper, we describe a study investigating the effect of positive feedback performed in the context of SQL-Tutor. We start by briefly describing this ITS in Section 2, and then describe our approach to providing positive feedback in Section 3. Section 4 presents the study and the results obtained, while the conclusions are given in the final section.

2 SQL-Tutor

SQL-Tutor assists university-level students in acquiring the knowledge and skills necessary to create SQL queries. SQL is the dominant database query language, which students find hard to learn. SQL-Tutor is a mature ITS, designed as a practice environment with the prerequisite that students be previously exposed to the SQL concepts in lectures. In the web-enabled version of SQL-Tutor, the architecture of which is shown in Figure 1, each student is assigned a unique web session. Students submit solutions which are sent to the student modeller for analysis. The student modeller identifies any errors or mistakes and updates the student model accordingly to reflect student progress within the domain. For more details of SQL-Tutor, please see [6].

To check the correctness of the student's solution, SQL-Tutor compares it to the correct solution, using domain knowledge, represented as a set of 700 constraints. A constraint consists of a relevance condition C_r, which checks whether the constraint is appropriate for a particular student's solution, and a satisfaction condition, C_s. A solution is correct if it satisfies the satisfaction conditions of all relevant constraints.

Once the student's solution is evaluated, the student model passes information to the pedagogical module which generates the appropriate feedback. If any constraints are violated, SQL-tutor will provide feedback on them. In the case where the solution is correct or the student requires a new problem to work on, the pedagogical module uses the information from the student model to select an appropriate problem.

SQL-Tutor provides feedback on demand only, when the student submits the solution. The system offers six levels of feedback, differing in the amount of detail provided to the student. On the first attempt, the system only informs the student whether the solution is correct or not. All other feedback levels provide negative feedback, i.e. feedback on errors. The second level (*Error Flag*) points the part of the solution that is incorrect. The third level (*Hint*) provides a description of one error, pointing out

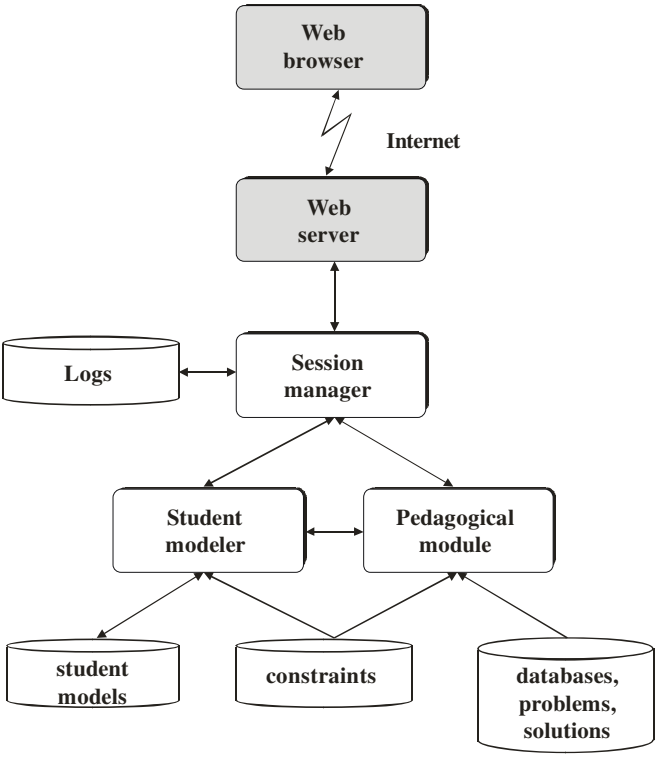


Fig. 1. Architecture of SQL-Tutor

where exactly the error is, what constitutes the error (performing blame allocation) and referring the student to the underlying domain principle that is violated (revising student's knowledge). The hint message comes directly from the violated constraint. The automatic progression of feedback levels ends at the hint level; to obtain higher levels of feedback, the student needs to explicitly require them. For example, the student can ask for the hint message for all violated constraints (All Errors), a partial solution (showing the correct version of one part of the solution that is wrong), or a complete solution for the problem.

3 Providing Positive Feedback

Consider a situation when a student is attempting a problem but is not entirely sure of what to do. That student makes a tentative attempt at the next step and surprisingly it turns out to be correct. The student is likely to store that piece of 'correct' knowledge if they are aware that their action is indeed correct. Therefore there must be some feedback mechanism which confirms to the student that their action is correct. Experienced human tutors support the learning process by providing feedback to confirm the correctness of the student's action. This is only one of the many situations where positive feedback is used quite effectively to support student learning. It therefore

appears that one of the ways in which positive feedback works is by reducing the number of tentative steps made by a student, creating and storing new knowledge chunks in cases where the student was lacking, or strengthening in situations where there was uncertainty and doubt.

Many student steps are tentative; the student is guessing, or at least rather uncertain as to what to do. In either case, if such a move happens to be correct then providing assurance to the student of its accuracy helps to reduce uncertainty and apprehension. We hypothesize that positive feedback works by helping to reduce the amount of uncertainty associated with student actions and reinforce existing knowledge; in some cases creating new knowledge and by so doing reduces both the time taken by students to solve problems and the number of errors made.

We developed a systematic approach to delivering positive feedback. This approach addresses three main issues: timing, content and presentation of positive feedback messages. Regarding timing of positive feedback, we considered two questions: when and how often should positive feedback be given. We proposed that positive feedback be given when the student submits a solution, as is currently the case with negative feedback in SQL-Tutor. However, the system should not give positive feedback on each correctly used constraint, as the amount of such feedback would be overwhelming. Instead, we identified events which add positive feedback only on selected constraints to the other messages given to the student. The positive feedback provided to a particular student will depend not only on the submitted solution, but also on the student's knowledge (as captured by the student model) and the state of interaction. We developed four general cases when positive feedback should be given:

a) *When the student is expressing uncertainty but nevertheless does the right* thing. This situation occurs when the student has previously made errors in a similar situation, but has made a correct attempt at the current problem. In this case, positive feedback reinforces the newly learnt domain principles.
b) *When the student is too paralyzed to do anything at all*. In such situations the student is unable to proceed without additional aid. The student can only move forward if the tutor provides a direct hint on what the solution should be. This hint can be requested by the student, or provided automatically by the system after a period of inactivity. If the student uses the hint successfully, then positive feedback should be given. Based on our hypothesis, positive feedback should increase student confidence, decrease uncertainty over the student's existing knowledge, decrease uncertainty over the knowledge given within the hint statement which was once lacking and strengthen the connection between these two pieces of knowledge chunks.
c) *When the student has overcome aspects of the domain commonly agreed upon as being difficult and challenging*. Certain domain principles are difficult and challenging increasing cognitive load and the amount of active information processing required from the student. We propose giving positive feedback when the student correctly and adequately deals with such situations. This requires cognitive task analysis to identify such domain principles.
d) *When the problem or task has been successfully completed and at other major goals within the tutoring session.* Solving a problem represents a significant achievement. If a student gets a very difficult problem correct on the first attempt

or satisfies a difficult constraint the first time it is relevant, then it potentially signifies that the student has mastered those aspects of the domain. In this case we give positive feedback to indicate to the student the magnitude of their accomplishment and to reinforce that correct response.

Based on these cases, we developed a set of rules for providing positive feedback. Rule 1 generates positive feedback for each constraint satisfied for the first time (i.e. all previous attempts on that constraint were incorrect). Rule 2 provides positive feedback when the student was given a hint on how to solve the problem by SQL-Tutor after a period of inactivity, and has managed to apply the hint successfully (i.e. the student satisfied the constraint for the first time in its history). In that case, the student will be given positive feedback on the newly satisfied constraint. Rule 3 is similar to the previous rule, but covers the situation when the student requested a hint, rather than being provided with one automatically. Rule 4 provides positive feedback on a constraint that a student used correctly earlier in the session, but then kept violating it. When the student uses that constraint correctly again, rule 4 would generate reinforcing positive feedback on that constraint. Rule 5 covers a situation when a student was given a hint, and has consequently satisfied a constraint which was previously violated. Rule 6 covers a similar situation, but differs from rule 5 in the fact that the

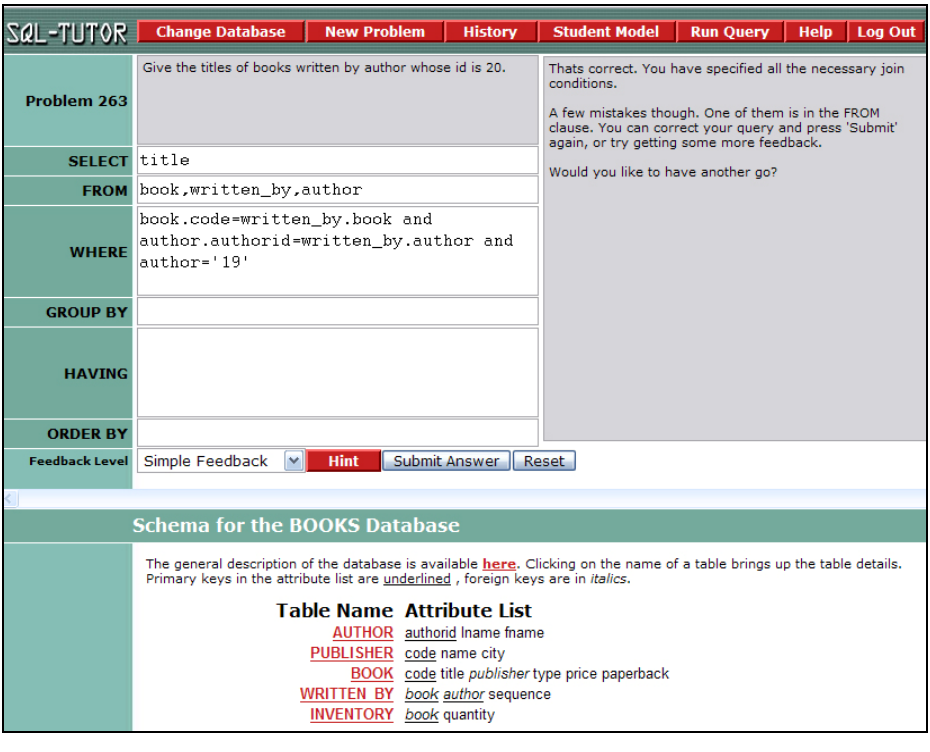


Fig. 2. Screenshot of SQL-Tutor providing both positive and negative feedback

student requested a hint. Rule 7 generates positive feedback in the case of a difficult problem being solved correctly, while rule 8 provides positive feedback after satisfying a difficult constraint. The rules are applied in this order.

The content of positive feedback is determined by the underlying learning theory SQL-Tutor is based on [9]. A positive feedback message is similar to the negative one: it points to a part of the solution which is relevant, and explains the domain principle involved. A negative feedback message in addition discusses how the student's solution violates the domain principle. Regarding presentation, we decided to provide positive and negative feedback simultaneously. If a student's solution was incorrect, but still matched some of the rules presented, positive feedback was presented first, followed by negative feedback. Figure 2 illustrates a situation in which the student received some positive feedback about join conditions, and then negative feedback, on the *Error Flag* level, pointing the student to a mistake in the FROM clause.

4 Evaluation Study

An evaluation study was conducted at the University of Canterbury with students enrolled in an introductory database course, from May 9, 2007 to June 6, 2007. The participants logged in to SQL-Tutor for the first time during scheduled labs, but could use it later at any time before the end of the study. The participants were randomly assigned to either the control group or the experimental group. The control group students received only negative feedback, while experimental group received both negative and positive feedback, as explained in the previous section. Out of 79 students enrolled in the course, 55 logged into SQL-Tutor at least once, with 51 completing the (online) pre-test. The maximum mark on the pre-test was 4. There was no significant difference between the mean scores on the pre-test for the two groups. All student actions were recorded in the logs. Some students used the system for a very short time, and we have eliminated logs of those students who used SQL-Tutor for less than 10 minutes, as we assumed that this was insufficient time to produce learning effects. After eliminating these students, we were left with 41 participants who attempted the pre-test. Table 1 shows some statistics from the study.

The experimental group students attempted and solved almost the same number of problems in significantly less time than the students in the control group, and also made fewer attempts at problems. The students from both groups also acquired the same amount of knowledge, as measured by the number of constraints learned. The number of learned constraints is an internal measure based on a very simple heuristic. The student model in SQL-Tutor stores the history of usage of each individual constraint. To see whether a student knows a constraint at the beginning of the interaction, we take the first five attempts on it. If the student satisfied that constraint more than seventy percent (70%) of times when it was relevant, then we assume the student knows the constraint. Otherwise, we assume that the student is yet to learn that constraint. At the end of the interaction with SQL-Tutor, we observe the last five attempts taken and perform the same calculation. If the student did not know the constraint initially, but has now satisfied that constraint more than 70% of the time, then the constraint has been learned. The measure therefore only applies to newly acquired constraints, learned as a result of using the system.

Table 1. Summary of students' interaction with SQL-Tutor

	Control	Experimental	p
Participants	23	18	
Pre-test mean (sd)	1.7 (0.8)	2.1 (1.3)	
Constraints learned	10 (6.1)	9.3 (6.8)	
Time (min)	193.8 (198.7)	92.3 (44.7)	0.012
Problems Attempted	28 (25)	26 (15)	
Problems Solved	25 (24)	22 (15)	
Total Attempts	119 (99)	98 (66)	
Time per Solved Problem	9.8 (7.9)	5.8 (4.8)	0.024
Time per Attempted Problem	7.5 (4.5)	4.1 (2.0)	0.002
Lab-test mean (%)	57 (26.5)	59.3 (24.3)	

We analysed the difference in means for the two groups for each of the measures reported in Table 1 using the t-test. The only significant differences are for the time spent with SQL-Tutor, average time per problem solved and the average time per attempted problem. The mean time for the experimental group is only half the amount of time used by the control group. While students in the experimental group spent on average 101.5 minutes less than students in the control group, they were able to solve almost the same number of problems; an average of 22 solved problems in the experimental group compared to an average of 25 by students in the control group. We also compared students' performance on a lab test, which was an assessment item for the course, performed after the SQL-Tutor study. The experimental group's mean performance on the lab-test was slightly higher, although the difference is not significant.

We also performed an ANCOVA analysis with time as the dependent variable, pre-test score as the co-variate and mode (group membership), number of negative feedback messages seen, number of solved problems and total attempts as explanatory variables. All factors together accounted for 86% (R^2=0.862) of the variability in total time and an interaction was noticed between mode and total time spent in the system (p=0.005). Also significant was the total number of attempts (p<0.0001). It is worth noting that prior knowledge as measured by pre-test was not a significant factor (p=0.392).

To further consider the impact of all factors on learning, we performed two multiple regression analyses. Of particular interest was the students' prior knowledge as reflected in the pre-test score, the total time spent receiving tutoring and the number of feedback messages seen. The last factor consisted of the number of negative feedback messages seen by control group, while for the experimental group, we used the two factors: the number of negative and the number of positive messages. The number of learned constraints was used as a measure of learning. The results are shown in Table 2. The number of participants was low for this kind of analysis; however, we plan to repeat the study in 2008 and have a larger data set. Therefore the following results should be taken with caution. For the experimental group, the model we obtained including all four predictors accounted for 77% of the variance, with time contributing most. The number of positive feedback messages is the only (marginally) significant predictor, and accounts for 6% of the variance. For the control group, the

model with three predictors explains 71% of the variance, and the number of negative feedback messages is the only significant predictor.

These results suggest that the students' prior knowledge did not affect the average number of constraints learned and did not explain a significant portion of the variance in the learning for any of the groups. We see that the strongest predictor of learning in the case of the experimental group is the number of positive feedback messages seen, while in the case of the control group, the strongest predictor turns out to be the number of negative feedback messages seen. It is not surprising that negative feedback is the strongest predictor of learning, for this is a rather common finding in educational research; however it is interesting that positive feedback turns out to be the strongest predictor for the experimental group. It shows that student's learning is influenced considerably by positive feedback messages received and that the more positive feedback messages a student receives, the more they are likely to learn (β=0.625). This evidence supports directly the hypothesis of this research; that positive feedback works by reducing uncertainty in student knowledge, decreasing the number of future errors by strengthening weak knowledge and in some cases creating new knowledge. The high correlation between learnt constraints and time also supports this, r=0.74 for the experimental and r=0.68 for the control group.

Table 2. Multiple regression results for constraints learned

		R^2 change	**β**
Control (R^2=0.714)	Pre-test	0.015 (ns)	0.083 (ns)
	Time	0.458 (p<0.001)	-0.285 (ns)
	Negative feedback	0.241 (p=0.001)	1.080 (p=0.001)
Experimental (R^2=0.766)	Pre-test	0.000 (ns)	-0.036 (ns)
	Time	0.560 (p=0.001)	0.075 (ns)
	Negative feedback	0.143 (p=0.021)	0.203 (ns)
	Positive feedback	0.063 (p=0.083)	0.625 (p=0.083)

We also performed a multiple regression of the lab-test scores using the same factors as a basis for comparison (Table 3). The models (containing all predictors) obtained account for 19% and 76% of the variance for the control and experimental group respectively. Unlike our previous findings with learned constraints, we found that student prior knowledge marginally significantly predicts lab-test scores for both groups. In the case of the control group, prior knowledge accounts for 13% of the variance. In the experimental group prior knowledge accounts for 57% of the variance but interestingly positive feedback is also marginally significant, accounting for 12% of the variance. Given the very low correlation between learned constraints and lab-test scores, and the findings of the learned constraint multiple regression analysis, these findings are not at all surprising. They reveal that students sitting the lab-test are influenced strongly by other sources of knowledge outside of that taught by SQL-Tutor. Despite this however, positive feedback still appeared to be a significant factor again supporting our hypothesis that positive feedback increases learning.

Table 3. Multiple regression results for lab-test score

		R^2 change	β
Control (R^2=0.185)	Pre-test	0.134 (p=0.094)	0.375 (p=0.096)
	Time	0.002 (ns)	-0.402 (ns)
	Negative feedback	0.050 (ns)	0.502 (ns)
Experimental (R^2=0.763)	Pre-test	0.569 (p<0.001)	0.546 (p=0.096)
	Time	0.052 (ns)	0.394 (ns)
	Negative feedback	0.024 (ns)	-1.042 (ns)
	Positive feedback	0.118 (p=0.031)	0.885 (p=0.096)

Figure 3 shows the learning curves for both groups. These curves show the error rate averaged over all constraints and all students, for each occasion when constraints were used. The actual data are well approximated by the power curves. There is no significant difference between the learning rates of the two groups, which is consistent with the previous finding about the number of constraints learned. Please note that the learning curves show how students learn constraints as a function of the number of attempts they used the constraints, not as a function of time elapsed between attempts. The control group students simply needed more time to learn the same amount of knowledge.

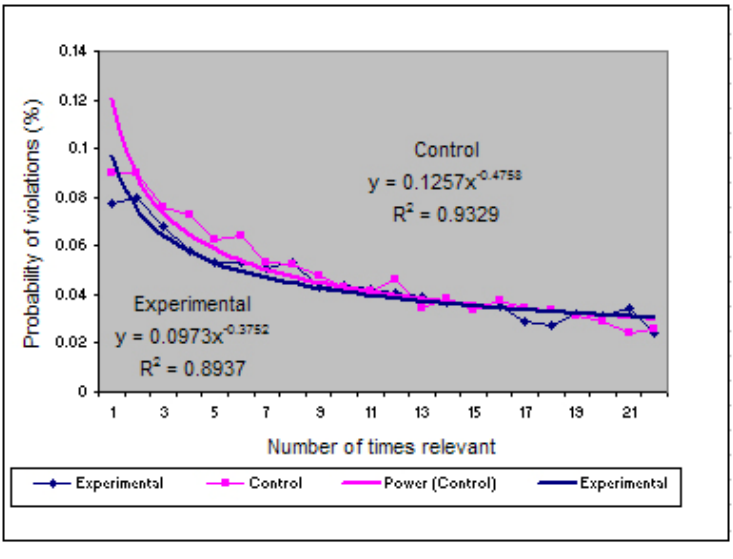


Fig. 3. Learning curves for all constraints

5 Conclusions

Starting from the observation that expert human tutors use positive feedback often, we modified SQL-Tutor, an ITS that teaches SQL querying, to give positive feedback.

The modified version of the system provides positive feedback to students in several situations: when they are unsure of their actions, after they successfully use a hint provided by SQL-Tutor (either requested or provided by the system), after learning difficult domain principles or completing problems correctly. We implemented a set of rules that take into account the student's solution, the student model and the current state of interaction, and generate positive feedback.

An evaluation study performed in an introductory database course showed that positive feedback does affect learning significantly. Although we have not observed a significant difference in the amount of knowledge learned while interacting with SQL-Tutor, the students who received positive feedback solved the same number of problems and learnt the same amount of knowledge as the students in the control group but in half the time of the control group. The difference in time is significant, thus proving the importance of positive feedback in ITSs.

Acknowledgements

The project presented in this paper has been supported by a NZ Commonwealth MSc Scholarship to the first author. We thank all members of ICTG for their support and help.

References

1. Ohlsson, S.: Learning from Performance Errors. Psychological Review 103(2), 241–262 (1996)
2. Anderson, J.R.: Rules of the Mind. Lawrence Erlbaum (1993)
3. The Condition of Education, Technical Report 2007064, Institute of Educational Sciences, National Center for Education Statistics, Washington, DC, http://nces.ed.gov/pubs2007/2007064.pdf
4. Koedinger, K.R., Anderson, J.R.: Intelligent Tutoring Goes to School in the Big City. I.J. AIED 8, 30–43 (1997)
5. Gertner, A., VanLehn, K.: Andes: A Coached Problem Solving Environments for Physics. In: Gauthier, G., VanLehn, K., Frasson, C. (eds.) ITS 2000. LNCS, vol. 1839, pp. 131–142. Springer, Heidelberg (2000)
6. Mitrovic, A., Ohlsson, S.: Evaluation of a Constraint-Based Tutor for a Database Language. Int. J. Artificial Intelligence in Education 10(3-4), 238–256 (1999)
7. Heffernan, N.T., Koedinger, K.: An Intelligent Tutoring System Incorporating a Model of an Experienced Human Tutor. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 596–608. Springer, Heidelberg (2002)
8. Ohlsson, S., Di Eugenio, B., Chow, B., Fossati, D., Lu, X., Kershaw, T.C.: Beyond code-and-count analysis of tutoring dialogues. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work, pp. 349–356. IOS Press, Amsterdam (2007)
9. Zakharov, K., Mitrovic, A., Ohlsson, S.: Feedback Micro-engineering in EER-Tutor. In: Looi, C.-K., McCalla, G., Bredeweg, B., Breuker, J. (eds.) Proc. Artificial Intelligence in Education AIED 2005, pp. 718–725. IOS Press (2005)

The Dynamics of Self-regulatory Processes within Self-and Externally Regulated Learning Episodes During Complex Science Learning with Hypermedia

Amy M. Witherspoon[1,3], Roger Azevedo[1,3], and Sidney D'Mello[2,3]

The University of Memphis
[1] Department of Psychology
[2] Department of Computer Science
[3] Institute for Intelligent Systems
Memphis, TN 38152

Abstract. This paper examines the dynamics of college students' self-regulatory processes within self-regulated learning (SRL) and externally-regulated learning (ERL) episodes during hypermedia learning. We re-analyzed and extended the results from an original study recently conducted by Azevedo and colleagues [1] to address four questions related to adaptivity, based on the temporal and dynamic deployment of self-regulatory processes by learners and human tutors in fostering complex science learning with hypermedia. Our questions include: (1) How does access to a human tutor affect the deployment of various SRL processes during learning?; (2) Which transitions between self-regulatory processes are more likely to occur within SRL and ERL?; (3) Which transitions between SRL classes are more likely or less likely to occur with SRL and ERL learning episodes?; (4) Are there significant correlations between learners' observed likelihood of transitions (between SRL processes) and learning outcomes? Lastly, we discuss implications for the design of MetaTutor, an adaptive hypermedia learning environment.

Keywords: adaptivity, self-regulated learning, externally-regulated learning, human tutoring, dynamic processes, cognition, metacognition, complex science learning.

1 Introduction

Traditional computer-based learning environments (CBLEs) such as ITSs are effective to the extent that they can adapt to the needs of individual students by systematically and dynamically scaffolding key learning processes [2-4]. The ability of these environments to provide adaptive, individualized scaffolding is based on an understanding of how learner characteristics, system features, and the mediating learning processes interact during learning in particular contexts. A critical aspect of providing individualized instruction is scaffolding, or instructional support in the form of guides, strategies, and tools, which are used during learning to support a level of understanding that would be impossible to attain if students learned on their own [5,6].

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 260–269, 2008.

Despite our ability to provide adaptive scaffolding to students learning about well-structured tasks with traditional CBLEs [e.g., 2], providing adaptive scaffolding to students' learning about conceptually-challenging domains such as the circulatory system remains a challenge for hypermedia instruction [3,7-10]. We argue that harnessing the full power of hypermedia learning environments will require empirical research aimed at understanding what kinds of scaffolds are effective in facilitating individualized instruction, and when they are best deployed [8,10-11].

In this paper we present a re-analysis of a human tutoring study conducted by Azevedo, Greene, and Moos [1]. The original study examined the effectiveness of self-regulated learning (SRL) and externally regulated learning (ERL) on college students' learning about a science topic with hypermedia during a 40-minute session. A total of 82 college students with little knowledge of the topic were randomly assigned either to the SRL or ERL condition. Students in the SRL condition regulated their own learning, while students in the ERL condition had access to a human tutor facilitating their SRL. The researchers converged product (pretest-posttest declarative knowledge and qualitative shifts in participants' mental models) with process (think-aloud) data to examine the effectiveness of SRL versus ERL. Analysis of declarative knowledge measures showed that the ERL condition group mean was significantly higher than the SRL condition group mean on both a labeling (heart components) and blood flow diagram task. There were no statistically significant differences between groups on a matching (circulatory system components to definitions) task, but both groups showed statistically significant increases in performance. Further analyses showed that the odds of being in a higher mental model (as measured by an essay task; see [1]) posttest group were decreased by 65% for the SRL group as compared to the ERL group. In terms of SRL behavior, participants in the SRL condition engaged in more use of selecting new information sources, re-reading, summarizing, free searching, and enacting control over the context of their learning. In comparison, the ERL participants engaged in more activation of prior knowledge, use of feeling of knowing and judgment of learning, monitoring progress toward goals, drawing, hypothesizing, coordination of information sources, and expressing task difficulty.

While this study contributes to several fields by demonstrating the effectiveness of human tutors as effective external regulating agents in facilitating college students' learning about complex and challenging science topics, its process data can be further analyzed to examine and predict the types of SRL processes that are used by learners while learning alone (SRL condition) and with a human tutor (ERL condition). As such, this papers aims to address questions: (1) How does access to a human tutor affect the deployment of various SRL processes during learning?; (2) Which transitions between SRL processes are more likely to occur within SRL and ERL? (3) Which transitions between SRL classes are more likely or less likely to occur with SRL and ERL learning episodes? (4) Are there significant correlations between learners' observed likelihood of transitions between SRL processes and learning outcomes? The foundations for these questions and analyses are based on the following theoretical [12-15] and empirically-based [11,14,16-17] assumptions. First, the key to understanding individualized instruction is through examining the temporal and dynamic deployment of SRL processes during learning both by the learners and human tutors [see 5-6,18]. Second, the deployment of SRL processes involves several (a) macro-level classes of processes related to planning, monitoring, learning strategies,

and methods of handling task difficulties and demands; and that each class can be further sub-divided into (b) micro-level processes (e.g., planning involves activating prior knowledge, creating and coordinating sub-goals, and recycling goals in working memory [WM] [1,7,14,19-20]. This multi-level hierarchical analysis of the temporal and dynamic unfolding of SRL processes is key to understanding adaptivity during learning. Third, the deployment of these processes can be detected, and traced, modeled using a variety of cognitive on-line trace methodologies (e,.g., think-alouds, emote-alouds, log file traces) and subsequently modeled using AI and computational linguistics algorithms to build adaptive learning environments capable of scaffolding learners' SRL and complex learning [18,21]. Fourth, both learners and human tutors deploy self-regulatory processes that reflect the learning situation and may be driven using particular scaffolding methods, tutoring scripts, etc. [7,11].

2 Method

2.1 Participants

Participants were 82 non-biology college majors from a large public mid-Atlantic university in the United States who took part in a recent study published study by Azevedo and colleagues [1]. The mean age of the participants was 21 years.

2.2 Paper and Pencil Materials

Paper and pencil materials for the experiment included a participant informed consent form, participant demographic questionnaire, and identical circulatory system pretest and posttest. The pre/posttest were identical to those used by Azevedo and colleagues [1] and included a matching task, a labeling task, and a blood flow diagram task.

2.3 Procedure

Participants were randomly assigned to either the SRL (n = 45) or ERL condition (n = 37). Participants were given 20 minutes to complete all of the circulatory system pretest measures and then immediately given the learning task by the experimenter, which involved learning with a commercially available hypermedia environment to learn about the circulatory system for 40 minutes. Participants in both the SRL group and the ERL group received the learning task instruction verbally from the experimenter, as well as in writing on a sheet of paper that was available throughout the learning session (See [1, p. 75])

Participants in the ERL condition, in addition to receiving this instruction, had access to a human tutor who scaffolded student's SRL by prompting participants to: (1) activate their prior knowledge (PKA); (2) create plans and goals for their learning and to monitor the progress they were making toward the goals; and (3) deploy several key SRL strategies, including summarizing, coordination of informational sources, hypothesizing, drawing, and using mnemonics.

A tutoring script was used by the human tutor in the ERL condition to guide decision making in when prompts should be used and what kind of prompts to implement,

given the current learning situation. This script was created based on previous literature on human tutoring [5,6,18] and recent empirical findings from studies on SRL and hypermedia [7,12]. For more information about the tutoring script, please see [1, p. 72]. After the learning task, each participant had 20 minutes to complete the posttest measures.

2.4 Coding and Scoring of Product and Process Data

Pretest and Posttest scoring procedure. The procedure for scoring the matching, labeling, and blood flow tasks was identical to that used in [1].

Learners' think-aloud protocols and regulatory behavior. The raw data collected from this study consisted of 54.1 hours of audio and video recordings from 82 participants. A modified coding scheme from Azevedo and colleagues [8,12] was used to code participants' verbalizations in this study. Several recent theoretical models of SRL [13,15,20] informed the coding scheme's use of the following five classes of variables: *planning, monitoring, strategy use, handling task difficulty and demands,* and *interest*. *Planning* involves creating approaches to learning and setting goals, and activating knowledge of the task and contextual conditions. *Monitoring* involves metacognitive awareness of the dynamic conditions of the task, context, and self. *Strategy use* involves active deployment of learning strategies in pursuit of a learning goal. *Handling task difficulty and demands* includes efforts to control and regulate different aspects of the task and context. Finally, *interest* includes statements about learner interest in the task or content domain. Descriptions of each SRL variable, the class that each variable belongs to, and examples of each variable can be found in [16, p. 67-69].

Azevedo & colleagues' SRL model was used to segment the think-aloud data for coding corresponding SRL variables to each segment. This resulted in 11,567 segments (M = 141.1 per participant; SD = .97) to be coded in the next phase. A research assistant trained on Azevedo and colleagues' coding scheme then assigned a single SRL variable to each coded segment. There was agreement on 6,784 out of 6,941 segments (58% randomly sampled from the 11,567 segments), yielding an inter-rater agreement of .98. Any remaining disagreements were resolved through discussion between the two independent coders.

3 Results and Discussion

3.1 Question 1. How Does Access to a Human Tutor Affect the Deployment of Various SRL Processes During Learning?

Descriptive statistics for the proportional usage of each SRL process, by condition is presented in Table 1. The table only displays the top ten most frequently used SRL processes for each learning condition. The SRL group most frequently summarized, controlled the learning context by using features of the hypermedia environment to enhance reading of text, inspection of the diagrams, and manipulate the embedded animations, and took notes. By contrast, those in the ERL group most often expressed feelings of knowing (FOK), activated prior knowledge, and sought help from the

human regulating agent. It stands to reason that learners in the ERL group would demonstrate greater use of help-seeking than those in the SRL group, due to the presence of the regulating agent. As described earlier, part of the tutoring script the tutor adhered to included prompts to activate prior knowledge and monitor their progress throughout the learning session, which could explain these learners' use of feeling of knowing and prior knowledge activation to regulate their learning.

Table 1. Conditional probabilities of the most common SRL processes

SRL Behavior	SRL Group		ERL Group	
	M	*SD*	*M*	*SD*
Summarization	0.16	0.11	0.08	0.05
Take notes	0.13	0.15	0.08	0.07
Control of context	0.13	0.1	-	-
Feeling of knowing	0.08	0.06	0.16	0.07
Re-reading	0.08	0.07	0.03	0.04
Judgment of learning	0.05	0.05	0.09	0.04
Select new information source	0.05	0.04	-	-
Prior knowledge activation	0.04	0.04	0.14	0.09
Content evaluation	0.03	0.05	-	-
Subgoal generation	0.03	0.04	-	-
Help seeking behavior	-	-	0.1	0.07
Coordination of information sources	-	-	0.05	0.03
Read notes	-	-	0.03	0.04
Drawing	-	-	0.03	0.03

3.2 Question 2. Which Transitions between SRL Processes Are More or Less Likely to Occur within SRL and ERL Episodes?

We utilized the likelihood metric (***L***) developed by D'Mello, Taylor, and Graesser [22] as an analytical measure to compute the probability of any SRL behavior transitioning into another SRL behavior.

L explicitly accounts for the base rate biases of the destination SRL behavior in assessing the likelihood of a transition from a source SRL behavior to a destination SRL behavior. ***L*** is computed as:

$$L = \frac{\Pr(NEXT \mid PREV) - \Pr(NEXT)}{(1 - \Pr(Next))} \quad (1)$$

A value of 1 means that the transition will always occur; a value of 0 means that the transition's likelihood is exactly what it would be given only the base frequency of the destination state. Values above 0 signify that the transition is more likely than it could be expected, and values under 0 signify that the transition is less likely.

For this and subsequent analyses, only transitions between the most frequent SRL processes were considered. Using the ten processes most often employed in the SRL group and the ten most often used in the ERL group, this resulted in 100 possible transitions per group. Analyses were run separately on the SRL data and the ERL data to determine which transitions within each group were more (or less) likely to occur than base rate of that group only.

Specifically, for each participant, we computed the likelihood (***L***) value for each of the 100 SRL transitions. A one-sample, two-tailed t-test was then utilized to assess

whether the mean ***L*** was statistically significantly higher or lower than a hypothesized mean of 0 (chance level). To correct for Type I errors, Bonferroni correction was used, resulting in a *p*-value threshold of .0005 (.05/100). This analysis revealed 15 significant transitional likelihood metrics for the ERL group's SRL behavior, and four significant transitional metrics for the SRL group (See Table 2). For example, within the SRL class *monitoring*, it was significantly more likely that, in the ERL condition, feeling of knowing would follow judgment of learning. The remaining positive (or more likely) transitions within the ERL condition are all for transitions where the participant remains in a given state (e.g. prior knowledge activation → prior knowledge activation). Participants in the ERL condition appear to persist in a certain learning strategy (or planning process) more than learners in the SRL condition.

Table 2. Likelihood metrics for each significant transition between SRL codes, by learning condition

SRL Class and Transition	ERL Likelihood Mean(SD)	SRL Likelihood Mean(SD)
Planning		
Prior Knowledge Activation (PKA) → PKA	0.142(.157)	-
Sub-goal generation → Sub-goal generation	-	-0.049(.040)
Monitoring		
Judgment of Learning → Feeling of Knowing	0.132(.206)	-
Learning Strategies		
Coordination of Info→ Sources → Read Notes	-0.030(.039)	-
Draw → Read Notes	-0.036(.041)	-
Draw → Summarize	-0.086(.056)	-
Draw → Take Notes	-0.072(.095)	-
Read Notes → Draw	-0.044(.039)	-
Read Notes → Help Seeking Behavior	-0.083(.078)	-
Read Notes → Read Notes	0.279(.283)	-
Read Notes → Summarize	-0.095(.061)	-
Re-read → Read Notes	-0.028(.035)	-
Re-read → Sub-goal generation	-	-0.037(.042)
Summarize → Prior Knowledge Activation	-0.093(.119)	-
Summarize → Summarize	0.139(.193)	-
Take Notes → Prior Knowledge Activation	-	-0.030(.032)
Take Notes → Read Notes	-0.029(.037)	-
Take Notes → Take Notes	0.173(.236)	-
Handling Task Difficulty & Demands		
Control of Context → Take Notes	-	-0.079(.127)

* Empty cells indicate this transition was not significant within this learning condition.

3.3 Question 3. Which Transitions between SRL Classes Are More or Less Likely to Occur within SRL and ERL Episodes?

For the analyses on transitions of SRL classes (See section 2.5), all possible transitions between the five classes of SRL behavior (planning, monitoring, learning strategies, handling task difficulty and demands, interest) were considered, resulting in a total of 25 possible transitions. Analyses were run separately on the SRL data and the ERL data to determine which transitions within each group were more (or less) likely

to occur than base rate of that group only. Separate one-sample t-tests were run on the likelihood metrics (described in the coding and scoring section) to determine which transitions had likelihoods which were significantly different from zero. Due to the number of tests run in each analysis, the Bonferroni correction was used, resulting in a *p*-value threshold of .002 (.05/25). This analysis revealed five significant transitional likelihood metrics for the ERL group's SRL behavior, and one significant transitional metric for the SRL group (See Table 3). As with the SRL codes, the ERL participants evidenced more likelihood of remaining in the same SRL class (i.e. Planning → Planning, Learning strategies → Learning strategies). This indicates that access to a human tutor influences participants to continue using a similar SRL process (within the same class of SRL), whether through explicit prompts or positive feedback from the human tutor, or due to the sheer presence of the tutor.

Table 3. Likelihood metrics for each significant transition between SRL classes, by learning condition

Class Transition	ERL Likelihood Mean(SD)	SRL Likelihood Mean(SD)
Planning → Planning	0.124(.150)	-
Planning → Learning Strategies	-0.196(.240)	-
Monitoring → Learning Strategies	-0.107(.184)	-
Learning Strategies → Task Difficulty and Demands	-	-0.041(.080)
Learning Strategies → Learning Strategies	0.162(.132)	-
Learning Strategies → Planning	-0.080(.076)	-

* Empty cells indicate this transition was not significantly different from zero within this learning condition.

3.4 Question 4. Are There Significant Correlations between Participants' Observed Likelihood of Transitions (between SRL Processes) and Learning Outcomes?

This section describes a set of Spearman's Correlation coefficients which were assessed to determine if there was a relationship between the transition likelihood metrics obtained from the *ERL* participants and performance on the three learning measures. We restricted the correlational analyses to include only those transitions which were significant in the initial testing of the process transitions within the ERL condition (See Table 2). Each correlation is between one of the learning measure's gain score (posttest minus pretest) and the likelihood metric of one transition from one SRL process to the next (See section 2.5). We obtained four significant correlations between performance on learning measures and likelihood of transitions between SRL processes. First, there was a significant negative correlation ($\rho = -.40$, $p = .04$) between the likelihood of occurrence of drawing → summarization and performance on the matching task, which entailed matching 13 circulatory system components with their corresponding definitions. Also, the likelihood of reading notes → drawing was significantly negatively correlated with both the labeling gain scores ($\rho = -.58$, $p = .01$), in which participants labeled 14 parts of the heart on a diagram without a word bank, and the blood flow gain scores ($\rho = -.54$, $p = .01$), in which participants filled in the order of components of the circulatory system in blood flow,

using a word bank. Finally, there was a significant negative correlation between the likelihood of the occurrence of summarization → prior knowledge activation and the labeling gain scores (ρ = -.40, p = .02).

4 Implications for the Design of MetaTutor: An Adaptive Hypermedia Learning Environment for Learning about Science

MetaTutor is an adaptive hypermedia learning environment that is currently being developed by Azevedo and his research team (Cai, Graesser, Lewis, Lintean, McNamara, Rus, Smith, and Witherspoon). It focuses on teaching students self-regulated learning (SRL) processes in the context of learning about science topics such as the human circulatory system. Theoretically, it is based on models and frameworks of SRL [3,8,12-14,19-20]. The underlying assumption of MetaTutor is that students should regulate key cognitive, metacognitive, motivational, social, and affective processes in order to learn about complex and challenging science topics.

The purpose of MetaTutor (see Figure 1 above) is to examine the effectiveness of animated pedagogical agents as external regulatory agents used to detect, trace, model, and foster students' self-regulatory processes during learning about complex science topics. The design of MetaTutor is based on Azevedo and colleagues' extensive empirical results with hundreds of adolescents and college students that show that students' learning about a challenging science topic with hypermedia can be facilitated if the students are provided with adaptive human scaffolding that addresses both the content of the domain and the processes of SRL.

The results from these analyses of the dynamics of SRL behavior within SRL and ERL indicate that access to a human tutor affects the micro-level deployment of SRL processes throughout a learning session. Overall, results indicate that learning strategies are the most commonly used self-regulatory processes used during hypermedia learning followed by monitoring processes. In addition, results also show differences in the deployment of commonly used self-regulatory processes between conditions. Learners in the SRL condition tend to use fewer classes (i.e., strategy and methods of handling task difficulties and demands) while those in the ERL condition tended to use all five classes of SRL processes. These findings align with SRL frameworks and models [e.g., 13,15,20] and augment these models by providing evidence of the exact nature and classification of these processes. The transitional analyses indicate that learners in the ERL condition tended to remain in a certain state (persist with a certain SRL process or class) beyond base rates for these processes, indicating that these learners either chose, or were prompted, to continue employing a certain learning strategy at various times throughout the learning task. None of the transitions in the SRL condition demonstrated higher likelihood of occurrence than base rates. This

seems to suggest a certain amount of uncertainty to these learners' self-regulation. Information processing models (i.e. [20]) hypothesize that SRL is a constant interchange between monitoring processes and the control processes (related to learning strategies). Lack of evidence of a strong likelihood of transition from *monitoring* to *learning strategies* (or vice versa) within this data set suggests that these learners are not following the pattern of SRL which is prescribed within these models. Lastly, the negative correlations in this analsis seem to indicate that there are certain learning strategies such as drawing, reading notes, and summarization that should not be preceded by other strategies (summarization, drawing) and the activation of prior knowledge. This emerging evidence suggests that adaptive hypermedia environments should detect and trace these specific transitions during learning and offer student feedback to select other SRL processes.

Taking into account the previous findings [1] as well these more in-depth analyses, we assert that an adaptive hypermedia learning environment should simulate this human tutor as closely as possible. This will involve generating a learner model adaptively, tracing learners' paths within the hypermedia environment, comparing student-set learning goals (as well as learning goals provided by the system) with the content in various sections of the hypermedia environment, and prompting learners to monitor cognitive, task, and affective conditions and use effective learning strategies.

Acknowledgments. This research was supported by funding from the National Science Foundation (Early Career Grant ROLE 0133346, REESE 0633918, and ROLE 0731828) awarded to the second author. The authors thank Drs. Greene, Moos who co-authored the original study, Shanna Smith, Emily Siler, Andrew Trousdale, Jennifer Scott, Sarah Leonard, and Evangeline Poulos for preparing the process data.

References

1. Azevedo, R., Greene, J.A., Moos, D.C.: The effect of a human agent's external regulation upon college students' hypermedia learning. Metacognition and Learning 2(2/3), 67–87 (2007)
2. Koedinger, K., Corbett, A.: Cognitive tutors: Technology bringing learning sciences to the classroom. In: Sawyer, R.K. (ed.) The Cambridge handbook of the learning sciences, pp. 61–77. Cambridge University Press, NY (2006)
3. Azevedo, R.: Using hypermedia as a metacognitive tool for enhancing student learning? The role of self-regulated learning. Educational Psychologist 40(4), 199–209 (2005)
4. Lajoie, S.P., Azevedo, R.: Teaching and learning in technology-rich environments. In: Alexander, P., Winne, P. (eds.) Handbook of educational psychology, 2nd edn., pp. 803–821. Erlbaum, Mahwah (2006)
5. Chi, M.T.H., Siler, S., Jeong, H.: Can tutors monitor students' understanding accurately? Cognition and Instruction 22, 363–387 (2004)
6. Chi, M.T.H., Siler, S., Jeong, H., Yamauchi, T., Hausmann, R.: Learning from human tutoring. Cognitive Science 25, 471–534 (2001)
7. Azevedo, R., Jacobson, M.: Advances in scaffolding learning with hypertext and hypermedia: A summary and critical analysis. Educational Technology Research & Development 56(1), 93–100 (2008)

8. Azevedo, R., Witherspoon, A.M.: Self-regulated use of hypermedia. In: Graesser, A., Dunlosky, J., Hacker, D. (eds.) Handbook of metacognition in education, Erlbaum, Mahwah (in press)
9. Brusilovsky, P.: Adaptive navigation support in educational hypermedia: The role of student knowledge level and the case for meta-adaptation. British Journal of Educational Technology 34(4), 487–497 (2004)
10. Jacobson, M.J., Azevedo, R.: Advances in scaffolding learning with hypertext and hypermedia: Theoretical, empirical, and design issues. Educational Technology Research and Development 56(1), 1–3 (2008)
11. Azevedo, R., Hadwin, A.F.: Scaffolding self-regulated learning and metacognition: Implications for the design of computer-based scaffolds. Instructional Science 33, 367–379 (2005)
12. Azevedo, R.: The role of self-regulation in learning about science with hypermedia. In: Robinson, D., Schraw, G. (eds.) Current perspectives on cognition, learning, and instruction (in press)
13. Pintrich, P.R.: The role of goal orientation in self-regulated learning. In: Boekaerts, M., Pintrich, P., Zeidner, M. (eds.) Handbook of self-regulation, pp. 451–502. Academic Press, San Diego (2000)
14. Witherspoon, A., Azevedo, R., Greene, J.A., Moos, D.C., Baker, S.: The dynamic nature of self-regulatory behavior in self-regulated learning and externally-regulated learning episodes. In: Luckin, R., Koedinger, K., Greer, J. (eds.) Artificial intelligence in education: Building technology rich learning contexts that work, pp. 179–186 (2007)
15. Zimmerman, B.: Theories of self-regulated learning and academic achievement: An overview and analysis. In: Zimmerman, B., Schunk, D. (eds.) Self-regulated learning and academic achievement: Theoretical perspectives, pp. 1–37. Erlbaum, Mahwah (2001)
16. Azevedo, R., Moos, D.C., Greene, J.A., Winters, F.I., Cromley, J.C.: Why is externally-regulated learning more effective than self-regulated learning with hypermedia? Educational Technology Research and Development 56(1), 45–72 (2008)
17. D'Mello, S.K., Craig, S.D., Sullins, J., Graesser, A.C.: Predicting affective states expressed through an emote-aloud procedure from AutoTutor's mixed-initiative dialogue. International Journal of Artificial Intelligence in Education 16, 3–28 (2006)
18. Graesser, A.C., Bowers, C.A., Hacker, D.J., Person, N.K.: An anatomy of naturalistic tutoring. In: Hogan, K., Pressley, M. (eds.) Effective scaffolding of instruction, Brookline Books (1997)
19. Azevedo, R.: Understanding the complex nature of self-regulated learning processes in learning with computer-based learning environments: An introduction. Metacognition and Learning 2(2/3), 57–65 (2007)
20. Winne, P.H.: Self-regulated learning viewed from models of information processing. In: Zimmerman, B., Schunk, D. (eds.) Self-regulated learning and academic achievement: Theoretical perspectives, pp. 153–189. Erlbaum, Mawah (2001)
21. Biswas, G., Leelawong, K., Schwartz, D., Vye, N.: The Teachable Agents Group at Vanderbilt. Learning By Teaching: A new agent paradigm for educational software. Applied Artificial Intelligence 19, 363–392 (2005)
22. D'Mello, S.K., Taylor, R., Graesser, A.C.: Monitoring affective trajectories during complex learning. In: McNamara, D.S., Trafton, J.G. (eds.) Proceedings of the 29th Annual Cognitive Science Society, pp. 203–208. Cognitive Science Society, Austin (2007)

The Politeness Effect in an Intelligent Foreign Language Tutoring System

Ning Wang[1] and W. Lewis Johnson[2]

[1] USC Institute for Creative Technologies
13247 Fiji Way, Marina del Rey, CA 90292 USA
nwang@ict.usc.edu
[2] Alelo TLT, LLC
11965 Venice Bl., Suite 402, Los Angeles, CA 90066 USA
ljohnson@tacticallanguage.com

Abstract. A previous study showed that pedagogical agents that offer feedback with appropriate politeness strategies can help students learn better [21]. This work studied the Politeness Effect in a foreign language intelligent tutoring system, and provided further evidence that tutorial feedback with socially intelligent strategies can influence motivation and learning outcomes.

Keywords: Politeness Effect, Pedagogical Agents, Intelligent Tutoring System, Affective Computing, Second Language Acquisition, Motivation.

1 Introduction

Reeves and Nass have proposed the Media Equation hypothesis [20], which states that people respond to media, including computer-based media, as they do to other people. They argued that designers of computer systems should take this similarity into account. Researchers in intelligent learning environments have then begun to investigate how the Media Equation might apply to educational software. Lester et al. [15] conducted a study in which an animated pedagogical agent named Herman the Bug facilitated learning in an intelligent learning environment named Design-a-Plant. They posited the Persona Effect, that an animated pedagogical agent with a life-like persona and expressed affect could facilitate learning.

A number of pedagogical agent investigations have been conducted, seeking to understand the Persona Effect in more detail, and replicate it in a range of learning domains [9][18]. The results of these studies have been mixed. André et al. [1] demonstrated that an animated agent could help reduce the perceived difficulty of instructional material, and Bickmore [3] reported that subjects liked an animated agent that responded socially to them. But in neither study the agent yield differences in learning gains. Further studies [2][8][16] suggested that it was the voice of the animated agent that influenced learning, not the animated persona at all.

The animated persona itself may not be the primary cause of the learning effects of animated agents. Rather, if as Reeves and Nass suggest learners respond to pedagogical agents as if they were social actors, the agents' effectiveness should depend upon whether or not they behave like social actors.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 270–280, 2008.

Human tutors make extensive use of social intelligence when they try to satisfy learner's cognitive needs while motivating and supporting learners [14]. Porayska-Pomsta (2004) observed that expert human tutors use a wide range of strategies to phrase criticism so that it can indirectly "get the message through" without "hurting learner's motivation". She linked the "indirectness" in the feedback to the notion of politeness and tried to use the politeness theory [4] to explain the various strategies used by the tutor to phrase the tutorial feedback.

Politeness theory holds that people in all cultures have face wants: a desire for positive face (the desire to be approved of by others) and for negative face (the desire to be unimpeded by others). Many interactions between people, such as requests or instructions, potentially threaten face, and so people employ a range of politeness strategies to mitigate face threat and lessen an utterance's impact on positive or negative face. A series of studies of how human tutors interact with learners [10][11] found that human tutors use a range of tactics to address students' face.

Although politeness theory describes tutorial interaction at the tactical level, there is much more to tutorial interaction than face threat mitigation. Human tutors phrase their feedback comments to avoid negative impact on learner face and actively seek to influence learners' underlying motivational and affective states.

Researchers in motivation in learning e.g., [13], have identified several factors that promote learner motivation, including the so-called 4 Cs: confidence, curiosity, challenge, and control. Lepper & Woolverton [14] studied highly effective tutors in remedial mathematics education, and found that they employed motivational tactics to promote and optimize the 4 Cs. There is a close correspondence between the face wants identified by Brown and Levinson and some of these motivational factors. Negative face is related to control, so tactics that address learner negative face may also influence learner sense of control. Positive face is related to self-confidence; if learners have a sense that others approve of their performance, they are more likely to be more confident of their own performance. Thus, the politeness tactics that we observed in our studies of human tutors can be viewed as part of the tactical repertoire that tutors can employ to promote learner motivation.

Wang et al [21] applied the media equation hypothesis to the socially intelligent behavior of the pedagogical agent, in particular the use of politeness strategies. The study showed learners who received polite tutorial feedback achieved better learning results than those who received direct feedback. This effect is termed the Politeness Effect. Later, McLaren [17] applied the politeness strategies to intelligent tutoring systems in real classroom environment and failed to replicate the Politeness Effect.

In this paper, we present our investigation of the use of politeness in the context of a foreign language intelligent learning environment - Tactical Iraqi[1] [12]. The main question we want to address is: "How does politeness influence learning?" Our hypothesis is that pedagogical agents with proper politeness strategies can improve student's learning result by promoting student's motivation.

[1] Tactical Iraqi™ is a registered trademark of Alelo Inc. in the United States and other countries.

2 Tactical Iraqi

Tactical Iraqi is one of several game-based courses developed by Alelo TLT LLC., based on earlier prototypes developed at the Information Science Institute of University of Southern California. It is a training system that supports individualized language learning and helps military service members who may have no knowledge of foreign language and culture quickly acquire functional communication skills.

Tactical Iraqi includes three modules: the Skill Builder, the Mission Game and the Arcade Game. The Skill Builder consists of interactive lessons and exercises, and interactive game experiences. Learners use headset microphones to interact with the software, along with a keyboard and mouse. Lessons, exercises, and game experiences all involve speaking in the target language; speech recognition software is used to interpret the learner's speech.

Learners are introduced to concepts of language and culture in the Skill Builder lessons, and practice and apply them in the Arcade Game and Mission Game. The current study focuses on Skill Builder only. More information on the Arcade Game and Mission Game could be found in [12].

2.1 Feedback in the Skill Builder

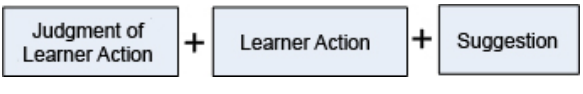
To study tutor feedback in Tactical Iraqi, we videotaped sessions of human tutoring in the context of Tactical Iraqi. Analysis of the videos revealed six different types of tutoring feedback:

- Acknowledgement/Criticize: acknowledge that the learner action is correct or incorrect.
- Elaborate: explains a language fact relates to learner's action.
- Suggest Action: offers hints to the student for the next step.
- Recast: when learner makes a mistake, instead of explicitly criticizing the action, tutor simply demonstrates the correct action.
- Encourage Effort: feedback aims to elicit more effort from learner.
- Consolation: consoles the student by saying his errors are expected.

We decided to implement acknowledgement/criticize, elaborate and suggest action in two types of Skill Builder pages: vocabulary pages and exercise pages. We designed recast as part of Suggestion – suggesting learner to listen to tutor speech again. Tutor strategies purely for motivational purpose (Encourage Effort and Consolation) are not included in the study.

2.1.2 Feedback on Vocabulary Pages

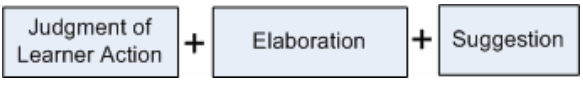
User interaction on a pronunciation page consists of listening to the tutor's phrase, recording learner's own speech, playing back the learner's speech and receiving feedback about the recorded phrase. We designed the feedback on vocabulary pages with the following structure:



The learner's speech is first processed by a speech recognizer. The feedback model receives the recognized phrase and compares it to tutor's phrase. If it matches then the Judgment of Learner Action is "correct", otherwise it's "incorrect". The second component of the feedback – Learner Action – displays the phrase recognized by the speech recognizer, e.g. "It sounds like you said 'as-salaamu 9aleykum (Hello)'." The third component of the feedback offers the learner a suggestion on what to do next: practicing the utterance more, listening to the tutor speech or moving on for now. An example of the complete feedback could be "Your pronunciation is incorrect. It sounds like you said 'as-salaamu 9aleykum (Hello)'. Try again."

2.1.3 Feedback on Exercise Pages

There are three types of exercise pages: utterance formation pages, multiple-choice pages and match-item page. User interaction on an utterance formation page consists of recording a response in the foreign language to a question and receiving feedback regarding the answer. Multiple-choice page consists of a multiple-choice question. Match-item page presents learner with match-item questions. Learner matches a list of phrases in Iraqi Arabic to their translations in English. The structure of feedback on utterance formation pages is shown below.



The first and third components are similar to the ones for pronunciation feedback. The second component – elaboration – presents analysis of learner's answer. The lesson XML, which defines the pages in the Skill Builder, also includes possible correct and likely incorrect answers to questions on an exercise Page. The feedback model retrieves the analysis from the lesson XML based on the answer recognized. An example of the complete feedback would be "Incorrect. 'li sh-sharaf' is used to formally accept an invitation, and not to respond to a new acquaintance. Try again."

Feedbacks like the example above can create threats to learner's face. Judging learner action, especially in the case of criticism, can threaten the learner's positive face. Suggesting action, e.g. "Try again" on the other hand, can threaten learner's negative face. To mitigate the face threat, we designed a series of politeness strategies for the feedbacks based on Brown & Levinson's politeness theory, as shown in Table 1. Examples of these politeness strategies are listed in Table 2.

Table 1. Politeness strategies in Skill Builder feedback

Feedback Component	Politeness Strategies
Judgment of Learner Action	Exaggerate, Common Ground, Conventionally Indirect, Be Vague, Understate, Impersonalize
Learner Action or Elaboration	Impersonalize
Suggestion	Common Ground, Tautology, Question, Impersonalize

Table 2. Examples of politeness strategies

Politeness Strategy	Example
Exaggerate	Great job!
Common Ground	Let's practice this a little bit more before we move on.
Be Vague	Looks like someone got the right answer!
Understate	This utterance needs a little bit more work.
Question	How about we listen to the tutor's speech again?
Tautology	Practice makes perfect.
Impersonalize	It might be helpful to listen to the tutor's speech again.
Conventionally Indirect	This utterance requires more practice.

Fig. 1. Feedback with appropriate politeness strategies delivered by an avatar

To apply the politeness strategies, a database containing phrase templates for each feedback components using different politeness strategies is created. The feedback model queries the database with feedback component type and politeness value. The database finds the all matches within a politeness value range, selects one at random and returns it to the feedback module. The feedback model combines the query results and delivers the feedback to the learner by an avatar (Figure 1). The avatar is not animated and no speech synthesized speech is used. The feedback is delivered in text.

3 Method

Sixty-one volunteers (59% women, 41% men) from the greater Los Angeles area participated in the study. They were recruited by responding to recruitment posters on Craigslist.com and were compensated $40 for three hours of their participation. On average, the participants were 38.4 years old (min=21, max=63, std=11.5), with 1.6% of them with high school diploma, 21.3% with some college education, 50.8% with college degree, 8.2% with some graduate education and 18% with graduate degree.

3.1 Design

To investigate the effect of politeness strategies in tutorial feedback, we created two types of feedbacks: a polite feedback which is phrased using various politeness strategies and a direct feedback which is phrased without any politeness strategies. An example of direct feedback is "No, that means 'This is a sergeant.' Try again." An example of polite feedback is "It's usually hard to get answers to this question right, but that means 'This is a sergeant.' How about we try it again?" The study was designed as a between-subjects experiment with two conditions: Polite (n=31) and Direct (n=30), to which participants were randomly assigned.

3.2 Procedure

Participants first watched a video about the Tactical Iraqi. Then participants filled out the pre-questionnaire packet. The experimenter then gave a brief introduction on how to use the Skill Builder of Tactical Iraqi. The experimenter informed the subjects to take the lessons in the Skill Builder in order and not to take any quizzes. Next, participants started training in the Skill Builder in Tactical Iraqi. Participants in Polite condition received polite feedback while participants in Direct condition received direct feedback. Experimenter turned on the camcorder and left the room. One hour later, experimenter returned to the laboratory and ended session 1.

The next day, participants came back and completed another hour of training. They were then asked to write down the name of the lessons they took in Skill Builder. Later participants filled out the post-questionnaire packets. Finally, they took the quizzes from the lessons they took in Skill Builder.

3.3 Apparatus

Two Dell laptop computers installed with Tactical Iraqi were setup in two separate rooms. A headset was connected to each laptop computers. A camcorder was setup in front of each laptop computers to record participants' behavior.

3.4 Measures

3.4.1 Learning Gains

Learning Gains were measured using quizzes at the end of each lesson in the Skill Builder. The quizzes contain three types of questions. First type of question is *Utterance-Formation* question, where participants answer the question by recording their

own speech. The second type of question is *Multiple-Choice* question. The third type of question is the *Match-Item* question, where participants match phrases in Iraqi Arabic to translations in English. Each correct answer scored 1 point. Participants took quizzes from all the lessons that they took during the 2 hour training.

3.4.2 Motivation

Two indices of motivation were measured: self-efficacy and perceived autonomy. Self-efficacy measured both in the pre-training questionnaire (α=.829) and the post-training questionnaire (α=.713). The difference between pre-training and post-training results will allow interpretation of how self-efficacy increased or decreased due to the training. Items from self-efficacy measure include items such as "Compare to others, I think I'm pretty good at learning Iraqi Arabic." Sense of autonomy (α=.885) was measured only in the post-training questionnaire. Example items from autonomy measure include "I feel the system was deciding what I should do next for me."

3.4.3 Individual Difference

Individual characteristics were measured in an attempt to test their possible interaction with the Politeness Effect. These individual characteristics include extraversion [5], openness [7], conscientiousness [5], preference for indirect help (α=.286) and attitudes toward language learning [6].

4 Results

Data from seven sessions were excluded. Two sessions were excluded because computer crashes and speech recognizer malfunctions. One session was excluded because participant's hearing and speech impairment. Four other sessions were excluded because participants "cheated" on the post-test. In Tactical Iraqi Skill Builder, lessons and quizzes are always accessible to the user. At the beginning of each experiment session, participants were instructed to not to take any quizzes. Immediately before the post-test (quizzes), participants were instructed not to review the lessons before or during the quizzes. Log data from Tactical Iraqi showed that participants from the four excluded sessions either took the quizzes before the post-test, or reviewed the lessons during the quizzes. As a result, data from 50 participants was included in the analysis, 25 from the polite condition and 25 from the direct condition. Student T-test was used to compare results from polite and direct group.

4.1 Learning Results

Overall, we did not find significant difference between polite and direct group on overall quiz scores (p=.626, M_{polite}=7.08, SD_{polite}=4.00, M_{direct}=6.56, SD_{direct}=3.48). We then broke down the comparison of learning performance to different types of quiz questions. On utterance-formation questions, there is significant difference between polite and direct group (p=.037, M_{polite}=5.08, SD_{polite}=2.66, M_{direct}=3.64, SD_{direct}=2.06). On other two types of questions, we did not find significant differences: multiple choice questions (p=.180, M_{polite}=1.92, SD_{polite}=1.78, M_{direct}=2.68, SD_{direct}=2.15), Match-item questions (p=.183, M_{polite}=.08, SD_{polite}=.28, M_{direct}=.24, SD_{direct}=.52).

Match-item question scores are extremely low because there were only a couple of them in the quizzes. And they only appear in quizzes of later lessons. Few participants encountered them in the post-training learning test.

4.2 Motivation Results

We compared change of self-efficacy to learn Iraqi Arabic from before two learning session. We found significant results on self-efficacy change (p=.045, M_{polite}=.848, SD_{polite}=.856, M_{direct}=.366, SD_{direct}=.803). On autonomy, we did not find significant differences between two groups. (p=.838, M_{polite}=3.04, SD_{polite}=1.04, M_{direct}=3.11, SD_{direct}=1.45).

4.3 Individual Differences

Two-way between groups analysis of variance were conducted to explore the impact of individual differences and polite treatment on learning and motivation. Medium splits were conducted on the individual differences variables to divide each variable into two categories.

4.3.1 Motivation to Learn Foreign Language

We found a marginally significant main effect of motivation to learn foreign language on change of self-efficacy to learn foreign language (F(1, 46)=3.93, p=.053, M_{group1}=.49, SD_{group1}=.80, M_{group2}=.84, SD_{group2}=.90, η^2 =.108). This means that self-efficacy of participants with higher motivation to learn foreign language increased more than those with lower motivation to learn foreign language. The interaction effect (F(2, 46)=3.58, p=.065) did not reach statistical significance. The main and interaction effects for motivation to learn foreign language on other autonomy and learning were not statistically significant.

4.3.2 Preference for Indirect Help

The main effect of preference for indirect help on learning outcomes and motivation were not statistically significant. The interaction effect of preference for indirect help and polite treatment on learning outcomes and motivation were not statistically significant.

4.3.3 Extroversion

The main effect of extroversion on learning outcomes and motivation were not statistically significant. The interaction effect of extroversion and polite treatment on learning outcomes and motivation were not statistically significant.

4.3.4 Intellect

There was a statistically significant main effect for intellect on the overall quiz score (F(1, 46)=11.28, p=.002, η2=.197), Utterance-formation question quiz score (F(1, 46)=6.11, p=.017, η2=.117) and Multiple-Choice quiz score (F(1, 46)=9.15, p=.004, η2=.166). This means that participants with higher self-reported intellect performed better on the learning test than participants with lower self-reported intellect. The main effects of intellect and self-efficacy and sense of autonomy were not significant.

The interaction effects of intellect and polite treatment on any of the quiz score measures were not significant. The interaction effects of intellect and polite treatment on self-efficacy and sense of autonomy were not significant.

4.3.5 Education

The analysis of variance show that the effect of education on overall quiz scores was significant ($F(1, 46)=5.53$, $p=.023$, $M_{group1}=6.17$, $SD_{group1}=3.22$, $M_{group2}=8.50$, $SD_{group2}=4.45$) and the effect size was moderate ($\eta^2=.107$). The interaction between education and politeness on overall quiz score was also statistically significant ($F(2, 46)=4.41$, $p=.041$), but the effect size was small ($\eta^2=.087$). For the participants with lower education, those who received polite treatment did not differ those who received direct treatment on their overall quiz score ($M_{Polite}=5.89$, $SD_{Polite}=2.87$, $M_{Direct}=6.47$, $SD_{Direct}=3.66$). For participants with higher education, the difference between those who received polite treatment and those who received direct treatment did not reach statistical significance either ($M_{Polite}=10.83$, $SD_{Polite}=5.00$, $M_{Direct}=6.75$, $SD_{Direct}=3.29$). We did not find any significant effect of education on self-efficacy and sense of autonomy. The interactions between education and politeness on self-efficacy and sense of autonomy were not statistically significant.

5 Discussion

This paper has presented an investigation of the Politeness Effect in a foreign language tutoring system. Our results showed that on utterance-formation questions in the quizzes, those who received polite treatment did significant better than those who received direct tutorial feedback.

In Tactical Iraqi Skill Builder, not all exercises are alike. The exercises in Tactical Iraqi are designed to progressively prepare learners to apply their skills in conversation. The focus of Tactical Iraqi curricula is to develop spoken communication skills. It provides learners with a progression of exercises that start with basic recognition and recall, and progress toward spoken conversation. The utterance-formation quiz questions are the ones that require learner to answer the question by recording their own speech. They are much more difficult and complex than multiple-choice and match-item questions, and are closer to real-life language use. The results shown here are similar to the study on Persona Effect [15] in that the polite agent helped learner perform better on only complex problems.

Our results also show that participants who received polite tutorial feedback increased their self-efficacy more than those who received direct tutorial feedback. This is consistent with our hypothesis. However, we did not observe significant difference between polite and direct group on self-report of sense of autonomy. This is likely to be because the study was carried out in only in the Skill Builder of Tactical Iraqi. The Politeness Effect, if it exists, is not likely to apply identically to all learners in all learning environments. In the Skill Builder, there is relatively little scope for learners to exercise their autonomy. They either speak the language correctly or they do not. And they can either move on or continue practicing. Politeness tactics that focus on learner autonomy may therefore have limited effect.

Several individual differences showed influence on the learning result, although very few showed interactions with the polite and direct experiment manipulation. It was not surprising that participants with higher motivation to learn foreign language showed higher increase of self-efficacy to learn Iraqi Arabic than those who with lower motivation. The influences of intellect and education are quite interesting, indicating that participants who consider themselves highly intellectual or received higher education learned better. However, there were small to none interaction with the experiment manipulation. Contrary to the findings in [21], we did not find preference for indirect help to have much influence on either learning or motivation. This may be because the instrument to measure preference for indirect help has low inter-item reliability (α=.286).

There are several limitations to current study. The learning gains were measured right after the training. Even though the utterance-formation quizzes are close to real-life conversations, measures in a role-playing interview maybe a more comprehensive measure of communication skills. The version of Tactical Iraqi used in the study was in its early development stage. Speech recognizer error could potentially reduce the credibility of the feedback.

In conclusion, designers of educational software should consider carefully how politeness strategies apply to their particular application. In interactive applications that provide feedback, there are typically many opportunities to employ politeness tactics. Conversely, system developers and content authors who neglect politeness issues may unintentionally introduce messages that threaten learner face. Attention to politeness issues may result in improved learner performance, as well as improved learner attitudes and motivation.

Acknowledgments. We'd like to thank Dr. Yigal Arens, Dr. Stacy Marsella and Dr. Jonathan Gratch for help seeking funding for this study.

References

1. André, E., Rist, T., Müller, J.: Guiding the user through dynamically generated hypermedia presentations with a life-like character. In: Proceedings of the 1998 International Conference on Intelligent User Interfaces, pp. 21–28. ACM Press, New York (1998)
2. Atkinson, R.K., Mayer, R.E., Merrill, M.M.: Fostering social agency in multimedia learning: Examining the impact of an animated agent's voice. Contemporary Educational Psychology 30, 117–139 (2005)
3. Bickmore, T.: Relational Agents: Effecting Change through Human-Computer Relationships. Unpublished PhD thesis. MIT, Cambridge, (2003)
4. Brown, P., Levinson, S.C.: Politeness: Some universals in language use. Cambridge University Press, New York (1987)
5. Donnellan, M.B., Oswald, F.L., Baird, B.M., Lucas, R.E.: The mini-IPIP scales: Tiny-yeteffective measures of the Big Five factors of personality. Psychological Assessment 18, 192–203 (2006)
6. Gardner, R.C., Lambert, W.E.: Attitudes and motivation in second language learning. Newbury House, Rowley, Mass (1972)

7. Goldberg, L.R.: International Personality Item Pool: A Scientific Collaboratory for the Development of Advanced Measures of Personality and Other Individual differences (1999), http://ipip.ori.org/ipip/
8. Graesser, A.C., Moreno, K., Marineau, J., Adcock, A., Olney, A., Person, N.: AutoTutor improves deep learning of computer literacy: Is it the dialog or the talking head? In: Hoppe, U., Verdejo, F., Kay, J. (eds.) Proceedings of International Conference on Artificial Intelligence in Education, pp. 47–54. IOS Press, Amsterdam (2003)
9. Johnson, W.L., Rickel, J.W., Lester, J.C.: Animated pedagogical agents: Face-to-face interaction in interactive learning environments. IJAIED 11, 47–78 (2000)
10. Johnson, W.L., Kole, S., Shaw, E., Pain, H.: Socially Intelligent Learner-Agent Interaction Tactics. In: Proceedings of AIED 2003. IOS Press, Amsterdam (2003)
11. Johnson, W.L., Wu, S., Nouhi, Y.: Socially intelligent pronunciation feedback for second language learning. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220. Springer, Heidelberg (2004)
12. Johnson, W.L.: Serious use of a serious game for language learning. In: Luckin, R., et al. (eds.) Artificial Intelligence in Education, pp. 67–74. IOS Press, Amsterdam (2007)
13. Lepper, M.R., Hodell, M.: Intrinsic motivation in the classroom. In: Ames, C., Ames, R. (eds.) Research on motivation in education, 3rd edn., pp. 73–105. Academic Press, San Diego (1989)
14. Lepper, M.R., Woolverton, M.: The wisdom of practice: Lessons learned from the study of highly effective tutors. In: Aronson, J. (ed.) Improving academic achievement: Impact of psychological factors on education., pp. 135–158. Academic Press, Orlando (2002)
15. Lester, J.C., Converse, S.A., Kahler, S.E., Barlow, S.T., Stone, B.A., Bhogal, R.S.: The persona effect: Affective impact of animated pedagogical agents. In: Pemberton, S. (ed.) Proceedings of CHI 1997, pp. 3539–3666. ACM Press, New York (1997)
16. Mayer, R.E., Sobko, K., Mautone, P.D.: Social cues in multimedia learning: Role of speaker's voice. Journal of Educational Psychology 95, 419–425 (2003)
17. McLaren, B.M., Lim, S., Yaron, D., Koedinger, K.R.: Can a Polite Intelligent Tutoring System Lead to Improved Learning Outside of the Lab? In: The Proceedings of AIED 2007, pp. 443–440. IOS Press (2007)
18. Moreno, R., Mayer, R.E., Spires, H., Lester, J.C.: The case for social agency in computer-based teaching: Do students learn more deeply when they interact with animated pedagogical agents? Cognition and Instruction 19, 177–213 (2001)
19. Porayska-Pomsta, K.: Influence of Situational Context on Language Production. Unpublished Ph.D. thesis, University of Edinburgh, Edinburgh, United Kingdom (2004)
20. Reeves, B., Nass, C.: The media equation. Cambridge University Press, New York (1996)
21. Wang, N., Johnson, W.L., Rizzo, P., Shaw, E., Mayer, R.E.: Experimental Evaluation of Polite Interaction Tactics for Pedagogical Agents. In: Proceedings of the 2005 International Conference on Intelligent User Interfaces, pp. 12–19. ACM Press, New York (2005)

Investigating the Relationship between Spatial Ability and Feedback Style in ITSs

Nancy Milik[1], Antonija Mitrovic[1], and Michael Grimley[2]

[1] Intelligent Computer Tutoring Group, Department of Computer Science
[2] School of Educational Studies and Human Development
University of Canterbury, Christchurch, New Zealand
Nancy.milik@gmail.com
{tanja.mitrovic,michael.grimley}@canterbury.ac.nz

Abstract. Rapid and widespread development of computerised learning tools have proven the need for further exploration of the learners' personal characteristics in order to maximise the use of the current technology. In particular, this paper looks at the potential of accounting for spatial ability in ERM-Tutor; a constraint-based tutor that teaches logical database design. Our evaluation study shows no conclusive results to support a difference in effectiveness of the textual versus multimedia feedback presentation modes with respect to the students' spatial ability. However, we observed a number of trends indicating that matching the instruction presentation mode towards the students' spatial ability influences their perception of the system and motivation to use it, more than their learning gain.

1 Introduction

Intelligent Tutoring Systems (ITSs) are effective learning tools due to the adaptive pedagogical assistance they provide. They make decisions about the timing and content of teaching actions and feedback to each student based on their individual state. Students differ in their strategies, approaches, and capabilities for learning and processing cognitive information. Although it is evident that such personal characteristics play a vital role in the learning process, only a small number of studies have investigated the effects of accounting for them in ITSs. For example, Conati et al. [1] use the Five Factor personality traits (openness to experience, conscientiousness, extraversion, agreeableness and neuroticism) in representing different personality types and goal priority in a Dynamic Bayesian Network. This network is then used to maintain an assessment of the student's current emotional state. In contrast, EDUCE [2] uses the Multiple Intelligence learning characteristics (logical/mathematical, verbal/linguistic, visual/spatial and musical/rhythmic) in order to provide a customized learning path.

In this paper, we describe a project which focuses on the spatial ability, a psychometric construct [3] essential to activities related to spatial reasoning, such as the ability to manipulate images or spatial patterns into other arrangements [4]. Learners with high spatial abilities perform better with graphic or spatially-oriented content

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 281–290, 2008.

than those with low spatial ability. It is worth noting, however, that a low spatial ability score is not a deficit; there is evidence that it can be improved through training and practice [5, 6]. Nevertheless, changing ITSs to accommodate low spatial ability learners could be more practical and beneficial for the system/domain's problem-solving task. That is, learners with different spatial abilities should receive different types of content.

This paper presents an approach to support learners' spatial ability in ERM-Tutor [7], a constraint-based ITS [8] that teaches logical database design (i.e. the algorithm for mapping conceptual to logical database schemas). We start by presenting some relevant work in Section 2. Section 3 gives an overview of the tests we used to measure spatial ability, while the following section presents ERM-Tutor and the modifications made in this project. We then describe the preliminary study and the results obtained in Section 5, followed by conclusions and future work in the final section.

2 Related Work

Personal characteristics are a major factor in learning. Many theories exist regarding how individuals process and encode information differently, such as Richard Mayer's theory of multimedia learning [9-13]. Mayer defines multimedia as the presentation of material using both words and pictures, and proposes that presenting verbal explanations alone in instructional situations is less conducive to learning for some students than presenting verbal explanations in conjunction with pictures [9]. Subsequently, he defines a multimedia instructional message as communication that makes use of our dual learning channel [14, 15] which is intended to foster learning.

Figure 1 shows a representation of the dual channel theory. One channel is dedicated to processing words, whether printed or spoken, and the other is for processing pictorial forms. Based on this assumption, along with the assumptions that each channel has a limited capacity and require active processing, Mayer defines the *Cognitive Theory of Multimedia Learning* [10]. The theory states that learning occurs when learners attend to relevant incoming information (sensory memory), select and organise important information and integrate it with their prior knowledge (working memory) into mental representations (long-term memory). Mayer argues that making use of both visual and auditory channels when presenting learning instructions aids in deep, or meaningful, learning, indicated by good retention and transfer performance. His rationale is that when presenting a message combining an image and text, the information is effectively being perceived and processed twice (once through each channel). Moreover, the words and pictures complement each other, aiding the learner to mentally encode and integrate the information.

It is evident, however, that learners have different cognitive styles and abilities. Some people learn better with visual methods of instruction, whereas others learn better with verbal methods of instruction. The question that arises is whether presenting the same instructional information, based the multimedia theory, is beneficial for both groups of people? Or does it overload the mental processing of some people or even confuse them? More importantly, if learners process information differently, then how can an instructional environment be tailored to better suit their individual needs? Is it actually beneficial to customise digital instructional environments?

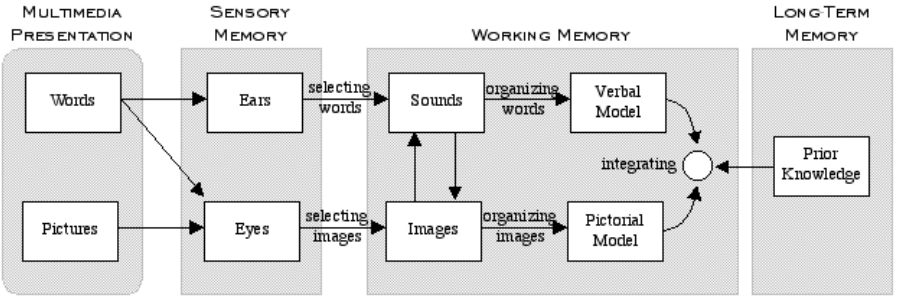


Fig. 1. Information processing via dual learning channels (Figure 3.2 from [9])

These are also some of the questions that Mayer considered. As the result, he documented a number of principles for designers of instructional environments to follow in order to make the maximum use of the learners' dual channels. For example, the *coherence principle* states that students learn better when extraneous words, pictures and sounds are excluded rather than included (i.e., presentations should be as clear and concise as possible to minimise mental processing overload). The principle that is of most interest to us however, is the *individual differences* principle, which states that "[multimedia] *design effects are stronger for low-knowledge learners than for high-knowledge learners and for high spatial learners rather than from low spatial learners*" (p. 161) [10]. This is because high-knowledge learners are able to use their prior knowledge to compensate for the cognitive processing needed to integrate the information received by the dual-channel. On the other hand, low-spatial learners must devote so much cognitive capacity to mentally integrate the information. Therefore, it is the combination of the learners' spatial ability and level of knowledge that influences their meaningful/deep learning.

3 Measuring Spatial Ability

Spatial ability is important in multimedia learning [10], as the learner needs to encode spatial information from sensory memory, maintain an internal representation in working memory, and perform spatial transformations in order to integrate the information in long term memory. There has been an interest in finding a correlation between individuals' spatial ability level and their gender and age. Studies investigating such correlation, for example testing spatial memory and spatial navigation through a novel environment, showed a male advantage for spatial performance, suggesting that spatial ability is one of the most reliable of all cognitive gender differences in humans [16], as well as an age related decline in performance [17].

The learners' spatial ability and the type of content representation directly affect the learners' cognitive load, level of concentration and motivation. Steinke et al. [18] investigated the usage of 3D models in a hypermedia learning system on plant and animal cell biology. They found that participants with high spatial ability levels spent more time on task-relevant content, whereas those with low spatial abilities spent

more time with the 3D models. Low spatial ability participants experience more difficulties in using 3D models and are easier distracted from task-relevant content. More interestingly, high spatial ability participants had a more positive attitude towards 3D content, thus confirming that a high subjective involvement results in a positive influence on the knowledge gain.

Psychometric tests (such as [19-21]) used for determining spatial ability typically consist of paper-and-pencil tasks requiring inspecting, imagining or mentally transforming shapes or objects at the *figural* scale of space [22]. These tests do not provide a discrete value on the spatial ability scale, but rather a relative position within a sample group that determines high or low classifications. We explored short versions of two tests from the battery of cognitive tests developed by Ekstrom, French, and Harman [19]: a ten-item Paper Folding Test intended to evaluate a component of spatial ability called visualisation, and an eighty-item mental Card Rotation Test which evaluates spatial orientation. Each test has a three-minute time limit and is suitable for ages 13-18.

4 ERM-Tutor

Constraint-based tutors enhance learning in a variety of domains, such as database querying (SQL-Tutor [23]), database design (ER-Tutor [24]) and data normalization (NORMIT [25]). ERM-Tutor [7] is a tutor in which students practice the algorithm for mapping conceptual database schemas (i.e. ER diagrams) into relational schemas. Each step in the algorithm maps one ER concept by either creating a new relation or altering previously created relations by adding foreign keys and attributes [26].

The interface (Figure 2) enables students to view problems, work on their solutions and receive feedback. The problem-solving area is the main part of the page, and its general layout is the same for all steps. First, there is a short description of the student's task for that step. For example, for step two the task text reads "*Map all the weak entity types*". This is basically to remind the student what is required in this step, rather than be educational material in its own right. The problem is presented to the student as an ER diagram, but the student also has an option of seeing a textual description of the database, by clicking the *Problem Text* button. Underneath the diagram, brief instructions on what is expected in this step and how to use the input boxes to create or alter a table are presented. At any time, the student can view the solution developed so far by clicking the *Completed Tables* button. This pops up a window containing all the relations defined by the student.

The student creates or alters one relation at a time. Each step of the algorithm is broken into subtasks. For example, in step one, the student maps one regular entity type at a time, and the system checks the resulting relation before moving on to the next entity type. Figure 2 illustrates a situation when the student has mapped the MEETING weak entity type, and has specified a relation (with the same name) with three attributes (*timing, id* and *description*). For each attribute, the student can specify whether it is a primary and/or foreign key. When the student completes the relation, he/she can request the system to check the solution. If there are any mistakes in the solution, ERM-Tutor provides feedback. In Figure 2, the system informs the student that there are some missing attributes representing foreign keys from the owners of

the MEETING weak entity type. If the solution is correct, the student can move on to the next entity type, or to the following step of the algorithm.

The feedback/help area occupies the top right side of the screen. A help page is displayed by default when a task page is first displayed. This provides a textual description of how to use the interface. When the student submits a solution, the help page is replaced by the appropriate feedback. Based on the feedback level chosen (*hint*, *explanation*, *list all errors*, or *full solution*), the system informs the student whether their solution was correct or not, and provides hints for correcting the errors. The help page can be redisplayed any time by clicking on the *Help* button.

Influenced by Mayer's work, we created a new version of the system. The original ERM-Tutor only provides text-based feedback. Following the multimedia learning theory, we decided to incorporate a pictorial aspect in the messages; for each feedback message, we created a graphically annotated version.

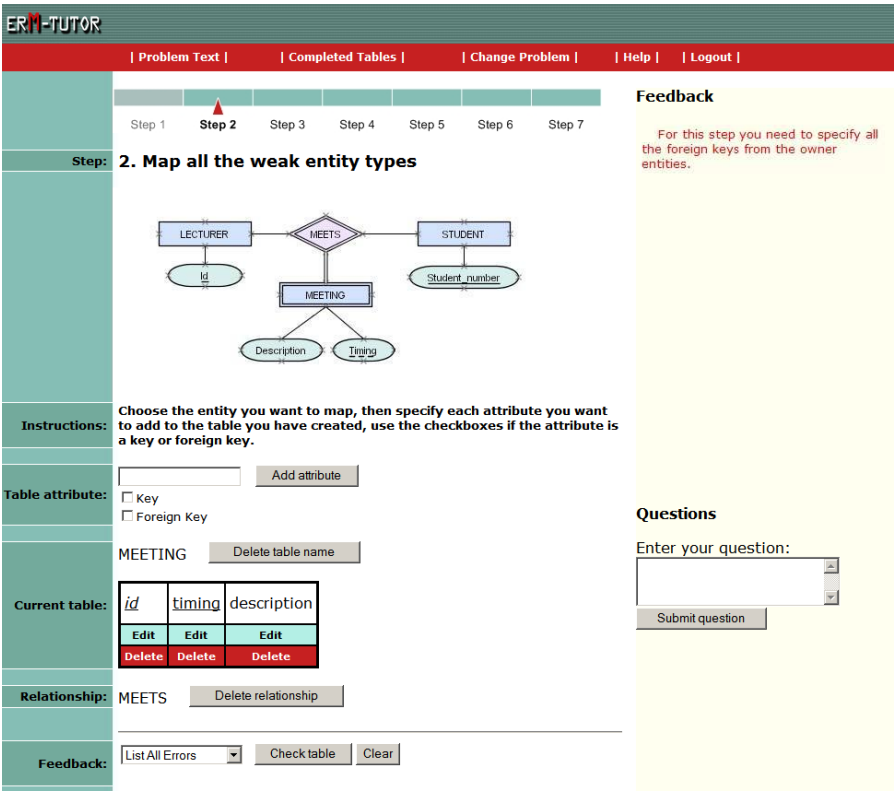


Fig. 2. Screenshot of ERM-Tutor

Being a constraint-based tutor, each feedback message in ERM-Tutor is associated with a constraint. In other words, each constraint has a feedback message which is displayed when the constraint is violated. Consequently, each message provides a hint on how to satisfy its particular constraint. To make the original and the newly created

messages comparable, we kept the text identical in both versions. The only difference is the addition of a pictorial representation in the new version. Figure 3 shows the multimedia (text and picture) version of the second feedback message given in Figure 2. A total of 112 images were created, each corresponding to a single feedback message. In addition, ERM-Tutor was modified to cater for both versions of feedback and prepared for an evaluation study described in the following section.

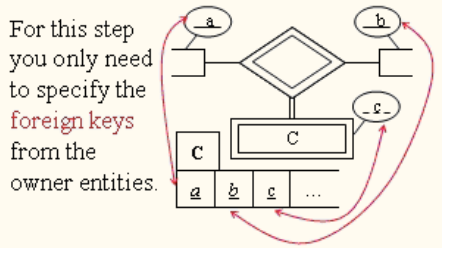


Fig. 3. An example feedback message in the multimedia form

5 Evaluation Study

We preformed an evaluation study with students enrolled in an introductory database course at the University of Canterbury in March 2007. Our hypothesis is that students with a high spatial ability level will benefit more from multimedia feedback than students with a low spatial ability, given the same background knowledge. As each student's spatial ability level (either high or low, as opposed to the actual value) is determined relatively to the sample group, we conducted a preliminary study in the previous year to calculate the students' spatial ability median score and we made the assumption that the students' samples from both years will be comparable. This score was used as the threshold in classifying students as having high/low spatial abilities.

Each participant was allocated to one of the versions of the system, providing either textual or multimedia feedback. The experiment allows for a $2x2$ comparison: textual messages for high (TH) and low spatial ability students (TL), and multimedia messages for high (MH) and low spatial ability students (ML).

The study was conducted in two sessions of scheduled labs on ER mapping, straight after students had attended lectures on the topic. Each participant attended one of the sessions, and worked with ERM-Tutor individually, solving problems at their own pace. At the start of a session, the students were given an information sheet describing the study, a consent form, and a pre-test on paper (with a maximal score of eight) consisting of four multichoice questions and a mapping question. To make the results of the pre-test and post-test comparable, two tests were used; students in the first session used version A as the pre-test and version B as the post-test and students in the second session used the reverse.

When a student logged onto the system, they were presented with a set of instructions explaining the two spatial ability tests, and a sample problem. Additionally, for each test, they were asked to rate own ability on a scale of 1 to 5 before sitting the tests. They had three minutes to solve the problems in each test. Once the spatial tests

were completed, or their time was up, the students were allocated to the appropriate version of the system and were asked to use it, solving as many problems as they would like. At the end of the session, students were asked to fill in a post-test and a questionnaire about the system. Finally, the students were encouraged to use the system at any time until the end of the course.

A total of 43 students submitted both pre and post tests. The mean score for all students on the pre-test was 4.3 (sd=2.2) and post-test was 5.2 (sd=2.2), resulting in a significant improvement in their performance ($t=3.4$, $p<0.001$). The scores for the different groupings of students are given in Table 1. The analysis indicated that there was statistically significant difference between the students' performance in the pre- and post- tests by those in the HT ($t=-3.4$, $p<0.005$), LM ($t=-2.0$, $p<0.05$) and HM ($t=-1.8$, $p=0.0553$) groups. However, there was no significant difference for the LT ($t=-0.2$, $p=0.4365$) group. A closer look at the LT group shows that its students have a higher pre-test score, with a mean of 5.8 (sd =1.1), and hence they improved the least in comparison with the other groups, scoring means of HT: 4.2 (2.4), LM: 3.6 (2.1) and HM: 4.2 (2.3). Although we hoped for an ideal setting of comparable groups, this imbalance in prior knowledge between the four groups was unavoidable.

Furthermore, ANOVA analyses across the pre and post- tests of the four groups did not yield any significant difference, indicating that all groups improved in a similar manner regardless of the feedback mode presented or their spatial ability. This suggests that presentation modes have similar influence on performance regardless of the spatial ability; that is, all students, whether low or high spatial, improved in performance regardless of the feedback mode they were given. We suspect however, that although ANOVA analysis on the pre-test did not indicate significant difference, the higher pre-test in the LT group has an influence on these statistical tests.

Table 1. Mean (*sd*) pre-test and post-test scores for all classifications of students

Classification	t-test: Paired Two Sample for Means				
	No.	**Pre-test**	**Post-test**	**t-stat**	**P-value**
LT	8	5.8 (1.1)	5.9 (2.5)	-0.2	0.4365
HT	12	4.2 (2.4)	5.7 (1.8)	-3.4	0.0029
LM	12	3.6 (2.1)	4.5 (2.1)	-2.0	0.0385
HM	11	4.2 (2.3)	5.1 (2.5)	-1.8	0.0553
Textual (LT HT)	20	4.8 (2.2)	5.8 (2.0)	-2.2	0.0185
Multimedia (LM HM)	23	3.9 (2.2)	4.9 (2.2)	-2.7	0.0070
Low (LT LM)	20	4.5 (2.1)	5.0 (2.2)	-1.4	0.0840
High (HT HM)	23	4.2 (2.3)	5.4 (2.2)	-3.6	0.0008
Matched (LT HM)	19	4.8 (2.2)	5.4 (2.5)	-1.4	0.0954
Unmatched (LM HT)	23	3.9 (2.2)	5.1 (2.0)	-3.8	0.0005

The tests for the rest of the groupings show either a statistically or marginally significant difference in students' performance between the pre- and post- tests scores. The p-values produced show that unmatched groupings scored a higher significant confidence than the matched groupings. A closer look at the figures show that the matched groupings had a higher pre-test mean score of 4.8 (sd = 2.2) than the

unmatched groupings (mean=3.9,sd=2.2). This difference was verified as marginally significant using a two-sample assuming unequal variances t-test (t=1.5, p=0.0759). A further test within the matched groupings, comparing the pre-test scores between the LT and HM groups indicated a significant difference (t =2.0, p<0.05).

We did find, however, an interesting trend in the data after analysing the students' log files and their interaction with the system. We looked at the total time interacting with the system, number of attempted problems, number of solved problems, percentage of solved problems and the total number of attempts/student solutions submitted (Table 2). We found that the HM group had a consistently higher mean for all these types of interactions, followed by the HT group, then the LT group and lastly the LM group with the lowest mean. The four groups came out in the same order for all the types of interactions we examined. Although the difference in numbers is quite small and statistically insignificant, this trend is in line with Mayer's theory that high spatial students will benefit more from multimedia presentation. In other words, the high spatial students appreciated and used the system more when they received multimedia feedback messages, whereas the low spatial students were less inclined to use the system when they received the multimedia feedback messages.

Table 2. Summary of means (*sd*) of system interaction results

	LT	**HT**	**LM**	**HM**
Total time (min)	62 (27.3)	72.7 (37.3)	57.7 (29.8)	72.2 (31.8)
Attempted problems	6.9 (6.3)	8.8 (5.0)	4.8 (3.0)	9.1 (5.0)
Solved problems	4.4 (5.0)	6.5 (4.6)	3.2 (3.8)	7.2 (4.5)
% solved problems	61.2 (36.8)	69.4 (26.1)	53.4 (43.2)	72.9 (21.3)
Total attempts	118.4 (97.7)	130.1 (73.2)	69.7 (46.6)	148.5 (90.7)

The same trend is reflected by the students' perception of the system indicated by their subjective results. Table 3 shows the mean responses to the 1 to 5 Likert scale questions of the four groups, where 1 represents the most negative response and 5 the most positive response. Again although the difference is not statistically significant, it seems that the LM group consistently reported the lowest ratings for the system, finding it more difficult and less interesting than the other groups. An interpretation of this could be that because the LM group spent more cognitive effort processing the feedback messages and hence enjoyed ERM-Tutor the least.

Table 3. Summary of means (*sd*) of subjective results for ERM-Tutor

	LT	**HT**	**LM**	**HM**
Overall quality	3.8 (0.7)	3.7 (0.7)	3.0 (1.1)	3.7 (0.8)
Terrible-Wonderful	3.4 (0.9)	3.5 (0.9)	2.9 (0.8)	3.6 (0.7)
Difficult-Easy	3.3 (1.0)	3.4 (0.8)	2.8 (1.1)	3.0 (0.8)
Boring-Fun	3.2 (1.0)	3.4 (0.9)	3.3 (0.8)	3.2 (0.8)

6 Conclusions

This paper looks at the potential of incorporating a multimedia representation of ERM-Tutor's feedback messages and evaluating the impact of various styles of feedback messages on learning in respect to the students' spatial ability levels. In the evaluation study, we presented students with one of the two feedback presentation modes, either textual or multimedia. We analysed the students' performance with respect to their spatial ability level and feedback mode they received. The results indicate that all students improved in their domain knowledge after interacting with ERM-Tutor. However, we did not find statistically significant difference between the pre- and post- tests scores across the four groups (LT, HT, LM and HM). Although we allocated similar number of students to each group, we were unable to control for the students' prior existing knowledge that influence their gain scores between their pre- and post- tests.

We observed a number of trends in the collected data. In particular, there was a tendency for students with high spatial ability who received multimedia feedback to interact the most with the system, and students with low spatial ability to interact the least with the system. Moreover, there was no noticeable difference between students receiving textual feedback regardless of their spatial ability. These findings indicate that, in terms of interactions with the system, the textual feedback had the same effect on students, whereas the multimedia messages had a greater effect on the high spatial students than on the low spatial students. We also note that students in the matched groups enjoyed interacting with the system more and were more motivated than those who were not matched. On the other hand, students in the unmatched groups had a higher learning gain than those who received matched type of feedback, based on the post-test scores. Although our contributions towards accounting for the students' spatial ability lack statistically significant measures, there is evidence that matching the presentation of instruction towards the students spatial ability has an influence on their perception of the system and motivation to use it, more than their learning gain.

Our analyses suggest that although students have a range of spatial ability skills, their preferences could be different than their ability levels. It is therefore, worth further investigating whether students have a differing preference to their capabilities. If this is evident, then we suspect that allowing the students to choose their preferred feedback presentation mode would increase their motivation and influence a positive affective state.

Acknowledgements

The project presented in this paper has been supported by an ICTG MSc Scholarship to the first author. We thank all members of ICTG for their support and help.

References

1. Conati, C., Maclare, H.: Evaluating a Probabilistic Model of Student Affect. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 55–66. Springer, Heidelberg (2004)

2. Kelly, D., Tangney, B.: Predicting Learning Characteristics in a Multiple Intelligence Based Tutoring System. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 678–688. Springer, Heidelberg (2004)
3. Jensen, A.R.: The G Factor: the Science of Mental Ability. Psycoloquy 10, 23 (1999)
4. Carroll, J.: Human cognitive abilities. Cambridge University Press (1993)
5. Baenninger, M., Newcombe, N.: The role of experience in spatial test performance: A meta-analysis. Sex Roles 20(5), 327–344 (1989)
6. Vicente, K.J., Williges, R.C.: Accommodating individual differences in searching a hierarchical file system. Man-Machine Studies 29(6), 647–668 (1988)
7. Milik, N., Marshall, M., Mitrovic, A.: Responding to free-form student questions in ERM-Tutor. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 707–709. Springer, Heidelberg (2006)
8. Ohlsson, S.: Constraint-based Student Modeling. In: Student Modeling: the Key to Individualized Knowledge-based Instruction. Springer, Berlin (1994)
9. Mayer, R.: Multimedia learning: Are we asking the right questions? Educational Psychologist 32, 1–19 (1997)
10. Mayer, R.: Multi-media Learning. University of California (2001)
11. Mayer, R., Moreno, R.: Nine ways to reduce cognitive load in multimedia learning. Educational Psychologist 38(1), 43–52 (2003)
12. Mayer, R., Sobko, K., Mautone, P.: Social cues in multimedia learning: Role of speaker's voice. Journal of Educational Psychology 95(2), 419–425 (2003)
13. Moreno, R., Mayer, R.: Cognitive principles of multimedia learning: The role of modality and contiguity. Journal of Educational Psychology 91(2), 358–368 (1999)
14. Baddeley, A.D.: Working Memory. Oxford University Press (1986)
15. Paivio, A.: Mental Representations: A Dual Coding Approach. Oxford University Press (1986)
16. Moffat, S.D., Hampson, E., Hatzipantelis, M.: Navigation in a "virtual" maze: Sex differences and correlation with psychometric measures of spatial ability in humans. Evolution and Human Behavior 19(2), 73–87 (1998)
17. Moffat, S.D., Zonderman, A.B., Resnick, S.M.: Age differences in spatial memory in a virtual environment navigation task. Neurobiology of Aging 22(5), 787–796 (2001)
18. Steinke, M., Huk, T., Floto, C.: The Influence of Cognitive Abilities and the Presence of 3D Models on the Use of Task Relevant Content in Hypermedia Learning Systems (2004)
19. Ekstrom, R., French, J., Harman, H.: Manual for kit of factor referenced cognitive tests. Princeton (1997)
20. Bodner, G.M., Guay, R.B.: The Purdue Visualization of Rotations Test. The Chemical Educator 2(4), 1–17 (1997)
21. Guay, R.B.: Purdue Spatial Visualization Test: Rotations. Purdue Research Foundation, West Lafayette, IN (1977)
22. Hegarty, M., et al.: Spatial abilities at different scales: Individual differences in aptitude-test performance and spatial-layout learning. Intelligence 34, 151–176 (2006)
23. Mitrovic, A.: A Knowledge-Based Teaching System for SQL. In: ED-MEDIA 1998, pp. 1027–1032 (1998)
24. Suraweera, P.: An Intelligent Tutoring System for Entity Relationship Modelling. Artificial Intelligence in Education 14(3), 375–417 (2004)
25. Mitrovic, A.: NORMIT, a Web-enabled tutor for database normalization. In: Proc. ICCE 2002, pp. 1276–1280 (2002)
26. Elmasri, R., Navathe, S.B.: Fundamentals of database systems. Addison-Wesley, Reading (2000)

Individualizing Tutoring with Learning Style Based Feedback

Shahida M. Parvez and Glenn D. Blank

Computer Science and Engineering Department
Lehigh University
Bethlehem, PA 18015 USA
sparvez@rcn.com, glennblank@gmail.com

Abstract. To approximate more closely effective human tutors, intelligent tutoring systems should adapt not only to a student's knowledge but also her learning style. We introduce a pedagogical framework that incorporates the Felder-Silverman learning style model and validated instrument for assessing individual learning style. The framework provides a feedback infrastructure based on the learning style model dimensions (such as visual, verbal, intuitive, sensor, etc.). It has been implemented as part of the DesignFirst-ITS, helping novices learn how to design a class in UML from a problem description. The system has been evaluated with high-school students and results show that learning style based feedback helps students realize higher learning gains.

Keywords: Learning styles, pedagogical module, feedback, user model, intelligent tutoring system, CS1, unified modeling language.

1 Introduction

Though ITSs are quite successful in helping students learn, they still fall short of the ability of effective human tutors to consider individual characteristics and preferences in order to customize both the tutoring content and process. The individual characteristics and preferences of the student are dubbed individual learning style. For the ITSs to match the success of good human tutors, ITSs need to adapt not only to the knowledge level but also to the learning style of the student to maximize learning.

Learning style refers to individual skills and preferences that affect how a student perceives, gathers, and processes learning materials [14]. People learn more when the instruction is matched to their individual learning styles [4, 6]. As a result, a number of adaptive educational systems have been developed that are based on learning style research: CS383 [3], Arthur [13], iWeaver [30], EDUCE [15]. These systems maintain a learning style profile for each student and use this profile to adapt the presentation and navigation of instructional content to each student.

Developing e-learning systems that adapt to student learning styles is not a trivial task. Design and development challenges include selecting the appropriate learning style model and instrument, creating course content consistent with the various learning styles, and determining the level and degree of adaptation of domain content. It is even more challenging to design an ITS that adapts to individual learning style

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 291–301, 2008.

because the ITS focuses more on student modeling and understanding of the domain knowledge rather than just the presentation mode and delivery as in adaptive hypermedia systems.

This paper presents a pedagogical framework that generates multidimensional feedback based not only on the knowledge level of the student but also on the individual learning style of the student. The pedagogical framework is based on the Felder-Silverman learning style model and is implemented in DesignFirst-ITS (once known as CIMEL-ITS), an ITS that helps novices learn object-oriented design by creating UML class diagrams. Evaluation with high-school students shows that students made significant learning gains after using the ITS. A GUI feedback maintenance tool makes it possible for teachers to add and update feedback in the ITS without any programming or assistance from the ITS developer.

The rest of this paper is organized as follows: section 2 describes related work; section 3 gives a brief overview of the Felder-Silverman learning style model; section 4 describes the pedagogical framework; section 5 describes DesignFirst-ITS; section 6 describes evaluation results; and section 7 presents the conclusion and future work.

2 Related Work

Learning style research became very active in the 1970's and has resulted in over 71 different learning style models and theories. Some of the most cited theories are Myers-Briggs Type Indicator [23], Kolb's learning style theory [17], Gardner's Multiple Intelligences Theory [11] and Felder-Silverman Learning Style Theory [7, 8]. Learning style research has been used in various settings and at different levels. In industry, corporations are using learning style research to create supportive work environments that foster communication and productivity. In academia, learning style research is being used for different purposes: to provide learning support to K-12 children who are either struggling or are gifted; to help college students maximize their learning gain by providing them insight into how they learn; and to help instructors design courses that appeal to students of various learning styles.

Learning style has also been integrated in adaptive e-learning environments based on learning style research. Adaptive e-learning systems are ideal for creating learning style based instructional material as they do not face the same limitations as human instructors who are unable to cater to individual students due to lack of resources [14]. Some of the adaptive systems that incorporate learning style are CS383 [3], ACE [25], AES-CS [27] and Flexi-OLM[18].

All these systems are based on different learning style models and use different methods to obtain a student's learning style. One method is to have the user fill out a learning style questionnaire. Another method is to infer student preferences from her interaction with the system, such as pages the student visits and links that she follows. After obtaining the student learning style, these systems use that information to adapt the sequence and/or presentation form of the instructional material to the student.

CS383 [3], an adaptive educational hypermedia system for a computer systems course (CS383), modifies content presentation using the Felder-Silverman learning style model. Learners fill out the Index of Learning Style questionnaire (ILS) that categorizes them as sensor/intuitive, verbal/visual and sequential/global. For example,

sensor learners like facts while intuitive learners prefer concepts; visual learners like pictures and graphics while verbal learners like written explanations; sequential learners prefer step by step presentations while global learners like to see the big picture first. In CS383, the presentation form of the content matches the student's learning style. For example, visual students are presented information in a graphical form while the verbal students receive the information in text form, etc.

The Flexi-OLM system [18] models a learner's understanding of basic C programming based on her answers to multiple-choice and short-answer questions. The system supports an open learner model that enables the learner to view information about her skill level, knowledge and misconceptions in a choice of seven formats, designed according to the Felder-Silverman learning style model.

Formal and informal evaluation studies of CS383, ACE and AES-CS suggest that students learn more when the system adapts to individual learning style. However, not all adaptive systems that incorporate learning style support the hypothesis that learning style adaptation results in increased gains. For example, evaluation studies of EDUCE [15] suggest that students learn more when they receive instruction that is mismatched to their learning style. One reason for these inconsistent evaluation results is that different systems are based on different learning style models and all these models have a different perspective of which individual characteristics affect the learning process. Another reason is that there are no set guidelines or standards that designers can use to create learning style based systems. Lack of standard methodologies also makes it difficult to determine the effectiveness of these systems. Yet another reason could be that some adaptive hypermedia systems use learner navigation data to keep an updated learning style profile of the learner and the learners do not necessarily only browse the information format that would be considered the best match for their learning style.

3 Felder-Silverman Learning Style Model

The Felder-Silverman Learning Style Model [7] categorizes a student's learning style on a sliding scale of four dimensions; *sensor-intuitive, visual-verbal, active-reflective* and *sequential-global.* Table 1 summarizes learning environment preferences of typical learners from each of these four dimensions of the Felder-Silverman model.

The Index of Learning Styles (ILS) instrument supports the Felder-Silverman learning style model by categorizing individual learning style preferences along four different dimensions of the model [10]. The ILS is a questionnaire containing 44 questions, 11 questions corresponding to each of the four dimensions of the learning style model. Each question is designed to determine if a respondent tends to belong to one category or another on that dimension. It does so by asking the respondent to choose only one of two options where each option represents a category. Since there are 11 questions for each dimension, a respondent is always classifiable along each dimension. The range of data for each dimension is from 0 to 11. Since there are four dimensions and each dimension has two poles there are 16 possible combinations, i.e. types of learner, in this model. However, the learning style dimensions of this model are continuous and not discrete categories at each pole. The learner's preference on a given scale does not necessarily belong to only one of the poles and may be strong, mild, or almost non-existent.

Table 1. Felder-Silverman learning style model dimensions

Active	Tries things out, works within a group, discusses and explains to others
Reflective	Thinks before doing something, works alone
Sensor	Learns from and memorizes facts, solves problems by well-established methods, patient with details, works slower
Intuitive	Discovers possibilities and relationships, innovative, easily grasps new concepts, abstractions and mathematical formulation, works faster
Visual	Learns from pictures, diagrams, flow charts, time lines, films, multimedia content and demonstrations
Verbal	Learns form written and spoken explanations
Sequential	Learns and thinks in linear/sequential steps
Global	Learns in large leaps, absorbing material almost randomly

The Felder-Silverman model was chosen for several reasons: ease of use; the Index of Learning Styles has been validated and provides a convenient way to assess student learning style [10, 19, 31, 28]; the limited number of dimensions of the model make it easier to incorporate it into an educational system; it has been used by educators at various institutions to improve education [7, 26]; many adaptive educational hypermedia systems, such as CS383 [3], TANGOW [24], and WHURLE [2], use this model to adapt the course presentation and/or sequence to individual learners.

4 Pedagogical Framework

Our pedagogical framework is designed to provide feedback that addresses multiple dimensions of the Felder-Silverman model. This framework consists of two parts: a *feedback infrastructure* that contains the feedback components and a *feedback generation process* that dynamically chooses these components to create coherent feedback based on students' learning style and students' erroneous actions. This pedagogical framework supports multiple levels of feedback, from a gentle reminder to a detailed explanation of the concept. This multiple hint strategy is called "hint sequencing" [12] and it refers to a sequence of hint templates that are used to generate feedback. The first hint is usually very general. As the student continues to need help about a given concept, the hints keep on getting more and more specific. Many of the successful tutors such as PAT [16], an algebra tutor, and LISPITS [5], a tutor for LISP, use this strategy. This strategy is also used by successful human tutors who offer multiple levels of feedback: they tend to start with a general hint and proceed to more specific hints related to students' erroneous actions [21].

Unlike other systems, our system does not end up providing the solution (since a design does not have only one solution), but instead offers more help in the form of pop-up hints or an extended tutorial. The tutorial mode (http://designfirstui.cse.lehigh.edu:8080/servlets-examples/servlet/GetLoginID) gives a student a detailed explanation of the concept with examples. The tutorial mode can also help students who just want to learn a concept before working on their solution. The student learning style information is obtained using the Index of Learning Style instrument (ILS) [9] which the students fill out prior to using the system for the first time. We now examine each part of our framework in more detail.

4.1 Feedback Infrastructure

The feedback architecture consists of the following components that contain information in the form that is suitable for different dimensions of the Felder-Silverman learning style model.

1. **Definition** – This component verbally introduces definitions of domain concepts. This particular component is useful for many learning style dimensions such as verbal, sensor, intuitive. An example of this component would be "Attributes are characteristics of an object that persist through the life of that object."
2. **Example** – This component illustrates a given concept. It can be used for almost any learning style, especially the sensor style which prefers a practical approach to concepts. An example of feedback in this component might be "Attributes of a car might be its color, model, make, etc."
3. **Question** – This component contains questions that could serve as hints during the interactive mode. There are two different types of questions: closed-ended questions that require a learner to simply answer yes/no or just provide a factual answer, and open-ended questions that require a student to think about her problem solving behavior. An example of a closed-ended question in this component might be "Is the correct data type to represent money a double?" while an example of open-ended question would be "Why did you set the data type for money to string?" Open-ended questions encourage the student to reflect about her reasoning process. This component is important for a "reflective" type learner as it gently nudges her to reflect on her action. It can also be useful for intuitive, global, and sequential learners as the open-ended questions can lead them to think about the relationships between different steps/things, about the big picture and about the steps involved in creating the solution.
4. **Scaffold** – This component nudges a learner who might be lost towards a correct solution by pointing her in the right direction. Often it is not enough to tell a novice that her action is incorrect; she needs guidance about where to learn more. For example, "Use the tutorial to learn about 'datatypes." This component is useful for global, intuitive, and sensor learners.
5. **Picture** – This component contains images, animation, or video that visually explains a concept. For example, when teaching the data type concept, one could create an image consisting of transparent containers marked as int, long, double, string, etc. These containers could have things such as a dollar sign in the double container, age in the int container, name in the string container, etc. Aimed at visual learners, this component also helps global learners see the big picture.
6. **Relationships** – This component contains information that helps a learner understand how a concept fits into the overall problem solving activity. Often learners understand a concept but have a difficult time understanding how it fits into the context of the problem. For example, a student might understand what attributes and methods are but might not know the relationship between the two in the context of the problem. This component is mainly for global learners.
7. **Application** – This component contains information about a concept that extends beyond the concept definition by showing an application. For example, a student might know the definition of a constructor but might not know that a class could have multiple constructors. This component is mostly suitable for sensor learners.

8. **Exercise –** This component supports active learning through hands-on activities or by applying a concept. It occurs in tutorial mode rather than hint mode, for all learning styles.

Each of these components has the following attributes that are used by the assembly algorithm to create feedback to be presented to the student.

Concept: unique concept in curriculum associated with the student's error;
Related_concept: relationship concept that the student may not understand;
Level: indicates the feedback level for which the component is designed;
Type: component feedback type (definition, question, etc.);
Category: component dimension (visual/verbal, active/ reflective, etc.);
Content: name of the visual / animation file;
Text: feedback text string;
Times_used: how many times this component has been used;
Status: active / inactive;
Presentation_mode: textual / graphical.

4.2 Feedback Generation Process

The feedback generation process uses the feedback infrastructure, domain knowledge and certain inputs to generate learning style based feedback. Feedback generation first *selects* feedback components based on inputs, then *assembles* a feedback message from the selected components. Inputs to the selection process include the student feedback history, learning style profile, student model information, and current student problem solving action packet. The student feedback history contains all the feedback that the student has received for each concept in the past and the feedback components that were used to generate the feedback. The learning style profile categorizes the preferred learning style of the student along the dimensions active/reflective, sensor/intuitive, sequential/global or verbal/visual. The student model specifies probabilistically how well the student understands each concept. The current student problem solving action packet provides the system with the action that the student performed, the error that was generated and the concept for which the student needs feedback.

The selection process uses these inputs to create a selection criterion which is then used to select feedback components from the feedback infrastructure. The selection criterion specifies information such as the concept the student needs help with, the presentation style of the feedback, current level of feedback and applicable feedback component types. Components whose attributes satisfy the selection criteria are chosen and put on a selected list of components.

The assembly process uses the components from the selected component list to create a feedback message. A feedback message first reiterates the action the student performed, second tells the student if the action was correct or incorrect, and third gives feedback about the given concept. The assembly process applies rules to subselect components from selected component list. One of the rules is that only a certain number of components can be used for each feedback level. Another assembly rule is that only one visual component is allowed in each feedback message, because each component explains a given concept in its entirety. After selecting components based

on the assembly rules, the assembly process creates a feedback message by putting all three parts of the feedback message together.

Generating learning style based feedback is a complex task, since the feedback must address the knowledge gap of the student and it must also be adapted to the student's learning preference. Besides its content, presentation of feedback also impacts its effectiveness.

Our pedagogical framework uses various dimensions of the Felder-Silverman learning style model to customize different aspects of the feedback. For example, the verbal / visual dimension is used to individualize presentation style. Verbal learners receive feedback in the form of written words while visual learners receive feedback that also emphasizes images, pictures and multimedia. The active / reflective dimension integrates hands-on activities for active learners and open-ended questions for reflective learners. The sequential / global dimension helps to determine if the learner should only receive feedback for the given concept itself or other information about how this concept relates to other concepts. The intuitive / sensor dimension determines whether to explain the concept abstractly or with concrete facts.

The system maintains a cross-reference of dimensions of the Felder-Silverman model and components in the feedback infrastructure. The system uses this cross-reference to choose components to generate learning style based feedback.

5 DesignFirst-ITS

DesignFirst-ITS is an intelligent tutoring system that provides one-on-one tutoring to help beginners in a CS1 course learn object-oriented analysis and design, using elements of UML [1]. DesignFirst-ITS is based on a "design-first" curriculum that teaches students to design a solution and the objects that comprise it before coding [22]. This curriculum enables students to understand and comprehend the problem without getting bogged down with programming language syntax.

The Curriculum Information Network (CIN) consists of domain knowledge which is object-oriented design concepts. These concepts are linked together through various relationships such as prerequisite and equivalence and are assigned a measure of learning difficulty. For example, Prerequisite (class: object) shows that the concept "object" is a prerequisite of "class." In other words, the student must understand what an object is before he can create a class.

The Expert Evaluator (EE) interfaces with a student through the LehighUML plug-in, created for the Eclipse Integrated Development Environment (IDE). Eclipse IDE is a Java development environment that can be extended by integrating plug-ins (software modules) to provide additional functionality. The LehighUML plug-in allows the student to create UML class diagrams. (A stand-alone version of LehighUML has also been created, for use outside of the complex Eclipse environment.) As the student designs a solution for a given problem in the plug-in environment, LehighUML reports each student action in a database on a server. The EE evaluates each of the student's steps in the background by comparing it with its own solution and generates an information packet for a correct student action and an error packet for an incorrect action. The Student Model (SM) analyzes these packets to determine the knowledge level of the student for each concept and attempts to find reasons for student errors [29]. The SM uses this information to update the student profile and passes the

original packets along with the reason packet that contains possible reasons for the student error to the Pedagogical Advisor (PA).

The PA is based on the learning style based pedagogical infrastructure described above. Taking into account the curriculum information network (CIN), the EE's analysis of the student's actions and the SM's analysis of the student's understanding of concepts in the CIN, the PA determines what feedback to provide to the student.

6 Evaluation Experiment and Results

DesignFirst-ITS was evaluated with 42 high-school students in the spring and summer of 2007 during multiple studies with identical materials. (Over 80% of the subjects were from underrepresented minorities and over a third of them were women.) The data from these studies were compiled and analyzed to determine if the learning style feedback resulted in bigger gains. The students who participated in the studies were novices to object-oriented design and programming. The evaluation procedure began by setting up three groups: a non-feedback group which did not receive any feedback at all; a textual-feedback group which received feedback in the form of plain text; and a learning-style-feedback group which received feedback that matched their learning style. The students filled out the Index of Learning Style (ILS) questionnaire, which the PA of DesignFirst-ITS used. The students were to learn the basic concepts of objects and classes and how to manipulate them in the Eclipse environment through a multimedia lesson. Then an instructor explained a step-by-step procedure for creating an object-oriented design solution for a problem description in English to generate the primary class, its attributes, and methods. As an assignment, the students were given a problem description of a movie ticket vending machine. The students followed the procedure to generate a solution from the problem description and to create a class diagram using the LehighUML plug-in and the DesignFirst-ITS.

Prior to using the system, the students took a pre-test to measure their prior knowledge and to give a baseline on which to compare the post-test. The pre-test and post-test consisted of fifteen multiple choice questions pertaining to definition, identification and application of object oriented concepts such as class, attribute, method, data type, etc. Then the students logged into DesignFirst-ITS and started to input their design in LehighUML. The students who belonged to the no-feedback group did not receive any feedback as their work was not supervised by the system. The textual-feedback group and the learning-style-feedback group both received feedback from the system as their designs were analyzed as they input them into the system. After the students had completed their designs, they were given the post-test.

The paired t-test suggests that there is no significant difference ($p>.05$) between the pre-test and post-test for the no-feedback group and the textual-feedback groups. However, the t-test shows a significant difference ($p<.001$) between the pre-test and the post-test for the learning-style-feedback group. The fact that there is no significant difference in pre-test and post-test for the no-feedback group makes sense because this group did not receive any help between the tests. On the other hand, the textual-feedback group could have shown some improvement between their test scores but interestingly did not. Many factors could have contributed to this lack of improvement: small sample size, students did not read the feedback, students did not understand the feedback, etc. One of the most likely reasons is that, in general, high school students do not like to read, and the

students in the evaluation study did confirm that by voicing their dislike about reading when they were asked to read the handouts and feedback carefully. For the learning-style-feedback group, the evaluation data shows that students who received the learning style feedback did realize learning gains after using the system. This result suggests that providing feedback in a student's preferred learning style is certainly worth the effort it takes to create it. It is also likely that students paid more attention to feedback in their preferred learning style.

In addition to the pre-test and post-test, the students who received any type of feedback were also given a Pedagogical Advisor evaluation questionnaire asking them specific questions about the feedback that they received from the ITS. The purpose of this questionnaire was to determine if the students found the feedback helpful in identifying their errors and in fixing these errors. Other reasons were to determine if the students liked the visual feedback and actually understood the information conveyed in the images / diagrams. The results of the survey showed that 90% of students read the advice, 72% found it helpful in identifying the error in their design, 71% found it helpful in correcting their errors, 65% liked the visual images and 69% understood the information conveyed in these images. Overall, 70% students liked the Pedagogical Advisor feedback.

7 Conclusions and Future Work

We have described a general framework to provide pedagogical advice tailored to an individual learning style, determined with a well-established learning style model and a validated instrument. We specified the overall architecture and feedback selection and assembly algorithm for a pedagogical advisor, which provides both popup hints and extended multimedia lessons. Evaluation results with high school students indicate that learning style based feedback helps students realize learning gains.

Given this general learning style framework, we develop a feedback maintenance tool that lets instructors add/delete feedback to this framework without programming or developer hand-holding (http://designfirstui.cse.lehigh.edu:8080/servlets-examples/servlet/GetAdminID). Two computer science instructors have used this tool successfully, validating its effectiveness.

Our most obvious future work is to demonstrate further the generality of the learning style framework and maintenance tool by applying them to other ITS domains. With more data, it may be also useful to determine which learning style dimension has the largest impact on student learning gains and to adapt feedback accordingly. It might also be useful to inform a student of her learning style to see if she appreciates the feedback or wishes to adapt the feedback so that she can learn along alternative learning style dimensions.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grants No. EIA-0087977 and 0231768 and the Pennsylvania Infrastructure Technology Association (PITA). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

1. Blank, G., Parvez, S., Wei, F., Moritz, S.: A Web-based ITS for OO Design. In: Proc. 12th International Conference on Artificial Intelligence in Education, Amsterdam (2005)
2. Brown, E.J., Brailsford, T.: Integration of learning style theory in an adaptive educational hypermedia (AEH) system. In: 11th International Conference of the Association for Learning Technology (ALT-C), Exeter, pp. 14–16 (2004)
3. Carver, C.A., Howard, R.A., Lane, W.D.: Enhancing Student Learning through Hypermedia Courseware and Incorporation of Learning Styles. IEEE Transactions on Education 42(1), 22–38 (1999)
4. Claxton, D.S., Murrell, P.: Learning styles: Implications for improving educational practices. ASHE-ERIC Higher Education Report No. 4. Washington: Association for the Study of Higher Education (1987)
5. Corbett, A., Koedinger, K., Anderson, J.: LISP Intelligent Tutoring System: Research in Skill Acquistion. In: Larkin, J.H., Chabay, R.W. (eds.) Computer-assisted Instruction and Intelligent Tutoring Systems: Shared Goals and Complementary Approaches, pp. 73–109. Erlbaum, Hillsdale (1992)
6. Dunn, R., Dunn, K.: Teaching students through their individual learning styles: A practical approach. Reston Publishing, Reston (1978)
7. Felder, R.M., Silverman, L.K.: Learning and Teaching Styles. Engineering Education, 674–681 (1988)
8. Felder, R.M.: Matters of Style. American Society of Engineering Education (ASEE) Prism 6(4), 18–23 (1996)
9. Felder, R.M., Solomon, B.A.: Learning styles and strategies (2001), `http://www.ncsu.edu/effective_teaching/ILSdir/styles.htm`
10. Felder, R.M., Spurlin, J.E.: Applications, Reliability, and Validity of the Index of Learning Styles. Intl. J. Engr. Education 21(1), 103–112 (2005)
11. Gardner, H.: Frames of Mind. Basic Books, New York (1983)
12. Gertner, A., VanLehn, K.: Andes: A Coached Problem Solving Environment for Physics. In: Gauthier, G., VanLehn, K., Frasson, C. (eds.) ITS 2000. LNCS, vol. 1839, pp. 133–142. Springer, Heidelberg (2000)
13. Gilbert, J.E., Han, C.Y.: Adapting Instruction in search of 'a significant difference'. J. Network and Computer Applications 22(3), 149–160 (1999)
14. Jonassen, D.H., Grabowski, B.L.: Handbook of Individual Differences, Learning and Instruction. Lawrence Erlbaum Associates, Hillsdale (1993)
15. Kelly, D., Tangney, B.: Incorporating Learning Characteristics into an Intelligent Tutor. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 729–738. Springer, Heidelberg (2002)
16. Koedinger, K.: Cognitive Tutors as Modeling Tools and Instructional Models. In: Forbus, K., Feltovich, P. (eds.) Smart Machines in Education, pp. 145–167. AAAI Press/MIT Press, Cambridge (2001)
17. Kolb, D.A.: Experiential learning: Experience as the source of learning and development. Prentice-Hall, Englewood Cliffs (1984)
18. Kyparisia, A., Papanikolaou, A.M., Bull, S., Grigoriadou, M.: Designing learner-controlled educational interactions based on learning/cognitive style and learner behaviour. Interacting with Computers 18(3), 356–384 (2006)
19. Litzinger, T.A., Lee, S.H., Wise, J.C., Felder, R.M.: A Study of the Reliability and Validity of the Felder-Soloman Index of Learning Styles. In: Proc. ASEE Annual Conference (2005)

20. Livesay, G., Dee, K., Felder, R.M., Hites, L., Nauman, E., O'Neal, E.: Statistical evaluation of the index of learning styles. In: Proc. ASEE Annual Conference (2002)
21. Merrill, D.C., Reiser, B.J., Ranney, M., Trafton, J.G.: Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. Journal of the Learning Sciences 3(2), 277–305 (1992)
22. Moritz, S., Blank, G.: A Design-First Curriculum for Teaching Java in a CS1 Course, SIGCSE Bulletin (inroads), pp.89–93 (June 2005)
23. Myers, I.B.: Introduction to Type. Gainsville, Fla. Center for the Application of Psychological Type (1976)
24. Paredes, P., Rodriguez, P.: Considering Learning Styles in Adaptive Web-based Education. In: Proc. 6th World Multiconference on Systemics, Cybernetics and Informatics, Orlando, Florida, pp. 481–485 (2002)
25. Specht, M., Oppermann, R.: ACE: Adaptive CourseWare Environment. New Review of HyperMedia and MultiMedia 4, 141–161 (1998)
26. Thomas, L., Ratcliffe, M., Woodbury, J., Jarman, E.: Learning styles and performance in the introductory programming sequence. In: Proc. 33rd SIGCSE Technical Symposium on Computer Science Education, pp. 33–37. ACM Press, Cincinnati (2002)
27. Triantafillou, E., Pomportsis, A., Demetriadis, S.: The design and the formative evaluation of an adaptive educational system based on cognitive styles. Computers and Education 41, 87–103 (2003)
28. Van Zwanenberg, N., Wilkinson, L., Anderson, A.: Felder and Silverman's Index of Learning Styles and Honey and Mumford's Learning Styles Questionnaire: How do they compare and do they predict academic performance? Educational Psychology 20(3), 365–381 (2000)
29. Wei, F., Blank, G.: Student Modeling with Atomic Bayesian Networks. In: Proc. 8th International Conference on Intelligent Tutoring Systems, Taiwan, pp. 491–502 (2006)
30. Wolf, C.: iWeaver: Towards 'Learning Style'-based e-Learning in Computer Science Education. In: Australasian Computing Education Conference (ACE 2003), Research and Practice in Information Technology, vol. 20, pp. 273–279 (2003)
31. Zywno, M.S.: A Contribution of Validation of Score Meaning for Felder-Soloman's Index of Learning Styles. In: Proc. Annual ASEE Conference (2003)

Use of Agent Prompts to Support Reflective Interaction in a Learning-by-Teaching Environment

Longkai Wu and Chee-Kit Looi

National Institute of Education
Nanyang Technological University, Singapore
longkai.wu@gmail.com, cheekit.looi@nie.edu.sg

Abstract. A learning-by-teaching environment (Biswas, Schwarz, Bransford et al., 2001), can be used to create a context in which student can play the role of tutor through teaching the agent tutee. Without meaningful feedback from the agent, there is no reason to expect student's engagement with the teaching interaction and growth in learning. This study tries to investigate the design of student-agent reflective interaction triggered by the agent prompts in a learning-by-teaching agent environment, Betty's Brain. A pilot study in using the prompts within the agent environment is undertaken. The result gives us some preliminary evidence that the agent prompt support on reflective interaction can be positive in enhancing student's learning when pursuing learning-by-teaching activities.

Keywords: Reflective Learning, Learning by Teaching, Teachable Agent.

1 Introduction

Recent research has shown the evidence of learning benefits to tutors from tutee's question prompts in the context of peer tutoring. Cohen, Kulik and Kulik (1982) demonstrated empirical evidence of learning gains for tutors compared to nontutors in the context of peer tutoring. King, Staffieri and Adelgais (1998) specially studied the tutor's explanations and questioning in the tutoring process as the sources for tutor's learning based on high-level question stems (i.e. questions prompting for comparisons, justifications, causes-and-effects, evaluations, etc.). Graesser, Person and Magliano (1994) showed that the tutee's occasional "deep" questions out of major "shallow" questions can stimulate the tutor's deeper response. Coleman, Brown and Rivkin (1997) demonstrated very similar findings in collaborative learning settings with students using high-level explanation prompts. Roscoe and Chi (2004) found that in a non-reciprocal and naturalistic (i.e. little or no training) tutoring context, the tutee's questions can motivate tutor explanations and meta-cognition, and thus have a significant and positive influence on the tutor's learning activities and opportunities.

Derived from the practice of peer tutoring, a computer-based learning-by-teaching environment (Biswas, Schwarz, Bransford et al., 2001), called Teachable Agent, is designed to offer specific advantages for research on cognitive effects of tutoring to tutors. In a learning-by-teaching environment, the student plays the role of a more capable tutor to teach a less knowledgeable computerized agent tutee by explicitly

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 302–311, 2008.

instructing (e.g. modeling) and observing the agent's independent problem solving activities to repair its mistakes. Biswas et al. (2005) found that students who taught the agent developed more integrated knowledge than those who did not. Roscoe and Chi (in press) remarked that, though it is not peer tutoring, the teachable agent environment is still impressive in increasing the student's learning gain by adding a peer-tutoring-like format to the intelligent tutoring system. The introduction of an agent tutee instead of human tutee allows researchers to concentrate on the tutor's learning opportunities inherent in the tutoring process. Some potential confounds, like spontaneous tutoring behaviors, can be obscured.

One of the major impediments in pursuing learning-by-teaching activities, especially for middle school students, is that they are likely to be novices in their disciplines and lack the autonomy to generate constructive schemes to fulfill their teaching tasks. Therefore, they need more guidance from the environment to facilitate their practice (Schwarz et al., 2005). Wagster, Tan, Biswas and Schwarz (2007) proposed the positive effects of metacognitive feedback in affecting learner's behavior in learning and transfer. Still few studies have examined the role of agent prompts, which can initiate learners' response and self-explanation in learning and teaching. In this paper, we focus on the design of generating agent prompts in a learning-by-teaching environment to initiate a reflective interaction between student and agent. We use this system to help students learn to monitor and improve their own learning behaviors, which involves student's responses to agent prompts and self-explanations.

The remainder of the paper is organized as follows. First, we present an overview of the research literature of reflective interaction between humans. Second, design issues of an agent prompts production schema are discussed. Third, a pilot study on how well these agent prompts deliver their proposed benefits to middle school students in learning-by-teaching environment is provided. Lastly, future directions for this work are described.

2 Reflective Human-Human Interaction

The reflective human-human interaction involves tutor's explanations and responses to tutee's questions. It is a crucial element of the tutoring process (Roscoe and Chi, in press). Graesser and McMahen (1993) found that the tutee's questions, which make the tutor notice his/her own contradiction or lack of knowledge, can promote confusion or curiosity in the tutor. Schwarz, Blair et al. (2005) further proposed that people learn best when an interaction co-mingles with ideas, which help them see their ideas reflected back to them as shaped by another person's thoughts, and this facilitates their ability to learn.

Grasser, Person et al. (1995) discussed the kinds of tutee questions that occur during tutoring, which can be divided into shallow and deeper questions. Shallow factual questions ("what" questions) ask definitions or simple calculations while deeper questions ("how" and "why" questions) ask causal relationships and underlying principles, requiring elaboration, inference and logical reasoning. Peverly and Wood (2001) indicated that deeper questions support learning more efficiently than shallow questions.

Davis (1998) proposed two types of reflection prompts, generic and directed prompts, which are "review" and "thinking" questions to guide the student in self monitoring, principle-based explanation and knowledge integration in middle school classroom. The students confronted to the reflection prompts were reported to learn more efficiently than unprompted ones.

In summary, the literature of human-human peer tutoring suggests that the deeper questions the tutee asks for elaboration and integration, the more productive explanations tutor responds with leading to critical thinking (Chi et al., 2001). However, no study has been undertaken on this relationship or has directly considered how different questions may facilitate human-agent reflective interaction.

3 Design of Agent Prompts Generation System

3.1 Overview

Our work is focusing on the generation and incorporation of meaningful agent prompts, which can arouse student-agent reflective interaction, in a learning-by-teaching environment. We build our work on an existing system, Betty's Brain (as shown Fig 1.), a learning-by-teaching environment originated from the Teachable Agent Group in Vanderbilt University. With the ability to learn what the students have taught by concept mapping, Betty's Brain is used to play the role of agent tutee in our research.

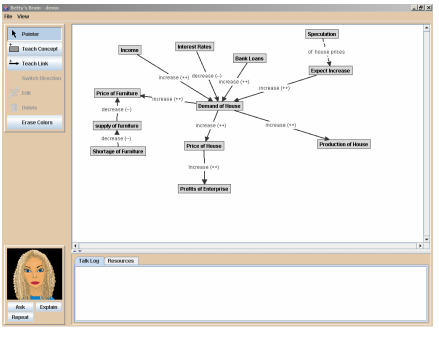


Fig. 1. A Learning-by-Teaching Environment: Betty's Brain ((Biswas, Schwarz, Bransford et al., 2001)

Within the agent environment, we create a built-in agent prompt generation system which can analyze, as well as compare student map with expert map tailored in the domain of basic economics. Our goal to adapt the current version of Betty's Brain is to let the agent tutee to raise meaningful question prompts, encourage student's responses and explanation to foster reflective interaction for better learning outcomes.

3.2 Design of Agent Prompt Generation System

In Figure 2, the architecture of Agent Prompts Generation System is depicted with four major components involved, namely the Agent Prompts Generator, Map Comparator and Map Analyzer and the Stage Detector.

The Agent Prompts Generator monitors the student's concept mapping activities (the student teaches Betty by producing a concept map in the agent environment) and plays the role of coordinator in the system. It mainly receives the results from the Map Analyzer and Map Comparator, selects proper prompts from the repository via Stage Detector and sends them to the Reflective Dialogue for students to respond to. Students tutor receives these prompts in an Reflective Dialogue and tries to respond while teaching the agent by modeling in a Concept Map Editor.

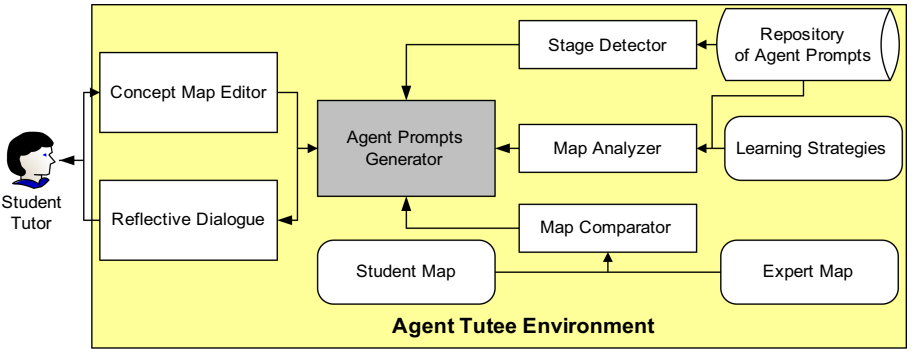


Fig. 2. Architecture of Agent Prompts Generation System

Map Comparator

Inspired by the work of Leelawong and et al. (2003), the map comparator can look for the following patterns: missing expert concepts, missing expert links, and incorrect expert links. Once a pattern is confirmed, the map comparator sends the message to the agent prompts generator and produces corresponding question prompts in the reflective dialogue component.

We adopt a new fuzzy integration and matching algorithm (Chen, Lin and Chang, 2001) to compare a student map (M') with the expert map (M). The fuzzy integration part allows one single expert map be integrated from works of several subject experts instead of only one, while the fuzzy matching part can take advantage of both propositional form and hierarchical structure in concept maps. It also has the potential to be better applied to students with higher performance and subjects with harder materials. Using this algorithm, the concept map is extended with attribute values which are associated with both nodes and links. The attribute values are used to reflect the relative significance of these concepts and relationships representing knowledge. A metric function is then applied to measure in terms of these attribute values to overlay parts of M and M'

$$D(M, M') = \sum_{i=1}^{n} |a_i - a'_i| + \sum_{j=1}^{m} |b_j - b'_j|$$

where a_i and a_i' are the attribute values of corresponding nodes n_i and n_i' belonging to M and M', respectively b_j and b_j' are the attribute values of the corresponding links l_j and l_j' in the respective maps.

Map Analyzer

We propose and design a Map Analyzer to analyze the insufficiency of student map and then generate some question prompts for student to reflect upon. Based on previous study on students' maps in our classroom experiments, the map analyzer, as a map structural information analyzing subsystem, synchronously traverses the semantic structure, collects the structural information, detects the common defects, and checks the repository for corresponding prompts and gives the real-time prompts. For example, the map analyzer can analyze limited or isolated nodes, wrong directional links, conflict routes of pair nodes (as shown in Figure 3) and monotonous structures (detecting limited link types).

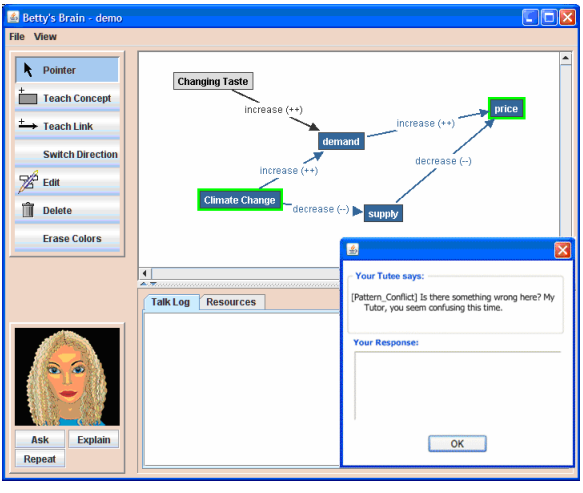


Fig. 3. Pattern of Conflict in the Learning-by-Teaching Agent Environment

For example, as shown in Figure 3, there is a conflict in the concept map, which means the reasoning following different paths engender different and inconsistent results. In this case, the increase of climate change will cause the price to increase following one path, while causing the price to decrease following the other path. Then the Pattern_Conflict pattern will be matched and triggers a corresponding question prompts asking the student to respond to the gap from the agent tutee.

Repository of Agent Prompts and Stage Detector

All the agent prompts are stored in a repository file in our system. Our current study seeks to compare two modalities of agent prompts, namely self-reflection prompts and agent tutee's question prompts, in the learning-by-teaching agent environment.

The self-reflection prompts are a series of metacognitive questions guiding student to reflect on learning and teaching, such as "Why should you teach?", "How can you use the tools to help your student understand what you teach?", "What do you learn from your student?", and et al. These prompts are context-independent and intended to remind student to assume the responsibility as teacher and teach the software better.

We also introduce another modality of prompts, the agent tutee's question prompts, in the learning-by-teaching environment, by trying to formulate reflective interaction between the student tutor and agent tutee. These question prompts, designed from the perspective of a tutee, are intended to make the agent behave in an inquisitive way and stimulate the student tutor in the process of understanding, monitoring, misunderstanding repairing and self-explanation when dealing with the complex and unfamiliar domains,

We employed a Stage Detector working with the repository to detect the student's tutoring stage, which includes before tutoring, during tutoring and after tutoring. It will trigger student in different cognitive and meta-cognitive aspects of when interacting with the agent tutee. Sample agent tutee's questions prompts adapted to three stages are as follows:

- Question Prompts Triggering General Thinking before Tutoring
  - [Introduction of the background story], *you are my tutor now, Can you teach me?*
  - *Do you feel you are prepared to teach me now?*
  - *What is the expectation of you for me?*
  - *How do you plan to teach me?*
- Question Prompts Triggering Domain Knowledge Thinking during Tutoring
  - *Can you explain some concepts you taught me just now?*
  - *Can you check is there any error concept/link in the map you teach me?*
  - *Can you check if there is any conflict link in the map you teach me?*
  - *I need more knowledge to finish my quiz task. Teach me more, please, my tutor.*
- Question Prompts Triggering Task-Specific Thinking during Tutoring
  - *Can you ask some questions related to the online resources? I am interested.*
  - *I haven't been asked quiz questions. Can you ask me those?*
  - *I have learned a lot from you. Can you send me to take a quiz?*
- Question Prompts Triggering Reflection Assessment after Tutoring
  - *Did I really learn from you? How will you evaluate your teaching work?*
  - *What is the most important thing you have taught to me?*
  - *Did you also learn something from me after you teach me?*

4 Pilot Study

4.1 Participants and Procedure

To assess the potential effectiveness and implications for designing agent prompts in a learning by teaching environment, we conducted a pilot study on 13 female students (five 12-year-old, three 13-year-old, four 14-year-old, and one 15-year-old) from a local secondary school. We chose the supply and demand in basic economics as the domain for students to pursue learning-by-teaching activities with Betty's Brain.

The 2-hour classroom pilot study was divided in 3 sessions. At the beginning of session 1, the students were told a background story, which requires students to help the Betty to pass economics exams to be prepared for assistance to her father in financial crisis. They were also introduced to basic features of the software. Then, the students prepared themselves to teach Betty by independently reading the available reference materials in basic economics.

In session 2, the 13 students were randomly divided into three groups: teach group, interaction group and reflection group. Each group was asked to teach different version of Betty's Brain. Teach group (3 members) used the version with no prompts. Interaction group (5 members) used the version with agent tutees's question prompts. Reflection group (5 members) used the version with self-reflection prompts. All members were told to teach their own agents by modeling while responding to the questions from the agent. They were encouraged to work independently without much communication among them.

In session 3, all students were asked to write down on an interview sheet their thoughts on their first learning-by-teaching experience.

4.2 Results and Discussions

The table below presents the mean number of valid causal links and concepts in the student maps, as well the standard deviation, together with the density ratio of valid concepts by valid causal links, in each group.

Table 1. Results of Average Valid Concepts and Links in Student's Map

Student map	Teach Group	Interaction Group	Reflection Group
Valid Concepts	13.3 (5.5)	7 (2.6)	13.4 (7.5)
Valid Causal Links	17 (6.1)	11 (6.9)	14.6 (7.7)
Density Ratio of Valid Concepts by Valid Causal Links	0.78	0.64	0.92

The analysis of the maps from the 3 groups in this study indicated that the maps of the teach group and reflection group tend to have richer and more complete structures than that of the "interaction" students. Excerpts of the interactive dialogue between student and agent from the interaction group are as follows.

[Before Tutoring]

① [Agent] Hi, nice to meet you, my tutor. I can't understand the relevant materials well. Please help to read them and teach me the way I can understand. How will you comment on the materials at the beginning?
[Your Response] Okay, please pay attention and don't be afraid to ask relevant questions.

[During Tutoring]

② [Agent when it detects that the several nodes are recently appended] It seems there are several concepts you have taught me. Can you choose some to explain to me?
[Your Response] Supply is the factor of time. This is because supply relies on how long a trend last and how long/much the supply. Supply and demand are a factor of disequilibrium. When supply and demand are not equal to each other, they contribute to disequilibrium.
….

③ [Agent when it detects that its answering behavior is just fulfilled] I just answered your questions. How will you evaluate my answer?
[Your Response] Poor. Try to paraphrase your answers
….

[Agent when it detects a conflict in the map] Is there something wrong you just teach me? My tutor. You seem confusing this time?
[Your Response] NIL
[Agent when it detects no response to previous prompt] Could you still check if there is any further conflict relationship in the map?
[Your Response] No, but it is a bit confusing.
….
[Agent when it detects the insufficiency of nodes/links in the map] I am always hungering for knowledge. Try to teach me more, please, my tutor.
[Your Response] Read the note.

[After Tutoring]

4 [Agent] Did I understand your meaning? How will you evaluate your teaching work?
[Your Response] My teacher and my past experience.
[Agent] How will you evaluate your teaching work?
[Your Response] I think not entirely. Maybe a little bit. It is my first time teaching. I will explore it in future.
[Agent] What was the most important thing you tried to teach me?
[Your Response] To let you understand what I mean
[Agent] Did you learn something for me?
[Your Response] Yes, you are inquisitive and ask questions when you are unsure of.

Comparatively, the following are other excerpts of the interaction between a student in the reflection group and self-reflection agent prompts.

[Before Tutoring]

1 [Agent] What is teaching and who is it for?
[Your Response] Teaching is a "Transference of knowledge" two way. It is for everyone.
[Agent] Who am I teaching?
[Your Response] Betty
[Agent] What should we be teaching?
[Your Response] Economics
[Agent] Why teach?
[Your Response] help Betty earn money
[Agent] Who teaches?
[Your Response] Anyone

[During Tutoring]

2 [Agent] How do you want students to interact with me?
[Your Response] Asking them questions. Getting them to explain topics
[Agent] What resources should be found to help me teach my student
[Your Response] websites, books, etc.
[Agent] How can you use the tools to help my student understand what you taught?
[Your Response] Concept maps are easy to "read" as it is something visual.
[Agent] Do I really understand your meaning?
[Your Response] No, you just follow the concept map as you don't explain the relationship.

3 [Agent] Can you give more elaboration on the concept I just taught to my students?
[Your Response] Yes, but I did not know how to give examples.
[Agent] Can you give some reasons on teaching the relationship between the concepts just you taught me?
[Your Response] No, I can't.

[After Tutoring]

4 [Agent] What was the most important thing you tried to teach me?
[Your Response] The relationships between demand, supply and price.
[Agent] What is/are the most important thing(s) my you will learn from me?
[Your Response] 1. How to teach using concept maps? 2. Basic concepts of demand or/and supply.
[Agent] What did you learn from me?
[Your Response] How to present my facts in a concept map?
[Agent] What was your thinking?
[Your Response] Focus on my goal to teach my student.

The students' self-explanations, from the reflective dialogues, give us some hints that a better student-agent interaction might be achieved by allowing the agent to pose questions to the student. These self-explanations also suggest that our designed agent prompts could be positive in enhancing their learning. Students are led to reflect on the knowledge in supply and demand, teaching tasks they are working on, past learning experiences and future learning plans.

On the interview sheet, one student chose "Feedback from someone who is less capable that challenge you" as the feedback that makes her most involved in the learning-by-teaching environment. She also commented that "If I am not sure about something, Betty can help me by setting me on the right track to find the answer by myself", but "there are some things that are hard to explain just by using the idea of increasing and decreasing.". She also appealed that "It would be more helpful if Betty could provide explanations in greater details". These feedbacks indicate that some more work can be done in extending the propositional representation scheme in Betty's Brain, such as accommodating more types of nodes and associating links.

5 Conclusion

From this pilot study, it is suggested that the support of agent prompts in a learning-by-teaching environment has the potential to engage students in a reflective interaction with the agent and help them promote reflection. Students require additional support, like agent prompts, from the context, in efficiently understanding the general metacognitive skills, task-specific skills and domain knowledge system for implementing learning. Feedback from the exploratory study in the secondary school class indicate that the idea of prompts could be successful in motivating student's thinking and getting them to thinking in depth when learning on complex and unfamiliar domains. More extensive studies will be conducted with a focus on how different modalities of question prompts can motivate students' learning in the environment.

Acknowledgements

We wish to thank Gautam Biswas for the Teachable Agents software. We have made some adaptations to the software. We are also grateful to the teachers (Michael Jalleh, Edwin Koh, Tan Hock Heng and Pertami Junia) of Raffles Girls' School (Secondary), Singapore, who helped us with the study.

References

Biswas, G., Schwartz, D.L., Bransford, J.: The Teachable Agents Group at Vanderbilt University. Technology Support for Complex Problem Solving: From SAD Environments to AI. In: Forbus,, Feltovich (eds.) Smart Machines in Education, pp. 71–98. AAAI Press, Menlo Park (2001)

Biswas, G., Schwartz, D.L., Leelawong, K., Vye, N., TAG-V.: Learning by teaching: A new paradigm for educational software. Applied Artificial Intelligence 19(3) (2005)

Chen, S.W., Lin, S.C., Chang, K.E.: Attributed concept maps: fuzzy integration and fuzzy matching. IEEE Transactions on Systems, Man, And Cybernetics 31(5) (2001)

Cohen, P.A., Kulik, J.A., Kulik, C.C.: Educational outcomes of tutoring: A meta-analysis of findings. American Educational Research Journal 19(2), 237–248 (1982); Coleman, E.B.: Using explanatory knowledge during collaborative problems solving in science. Journal of the Learning Sciences 7, 387–427 (1998)

Cohen, J.: Theoretical considerations of peer tutoring. Psychology in the Schools 23, 175–186 (1986)

Coleman, E.B., Brown, A.L., Rivkin, I.D.: The effect of instructional explanations on formal learning from scientific texts. The Journal of the Learning Sciences 6(4), 347–365 (1997)

Gartner, A., Kohler, M.C., Riessman, F.: Children teach children: Learning by teaching. Harper and Row, New York (1971)

Graesser, A.C., McMahen, C.: Anomalous information triggers questions when adults solve quantitative problems and comprehend stories. Journal of Educational Psychology 85(1), 136–151 (1993)

Graesser, A.C., Person, N.K., Magliano, J.P.: Collaborative Dialogue Patterns in Naturalistic One-to-One Tutoring. Applied Cognitive Psychology 9, 495–522 (1995)

King, A.: Transactive peer tutoring: Distributing cognition and metacognition. Educational Psychology Review 10(1), 57–74 (1998)

King, A., Staffieri, A., Adelgais, A.: Mutual peer tutoring: Effects of structuring tutorial interaction to scaffold peer learning. Journal of Educational Psychology 90(1), 134–152 (1998)

Leelawong, K., et al.: Teachable Agents: Learning by Teaching Environments for Science Domains. In: Proc. 15th Innovative Applications of Artificial Intelligence Conf., Acapulco, Mexico, pp. 109–116 (2003)

Peverly, S., Wood, R.: The effects of adjunct questions and feedback on improving the reading comprehension skills of learning disabled adolescents (2001)

Roscoe, R.D., Chi, M.: The influence of the tutee in learning by peer tutoring. In: Forbus, K., Gentner, D., Regier, T. (eds.) Proceedings the 26th Annual Meeting of the Cognitive Science Society, Chicago, pp. 1179–1184 (2004)

Roscoe, R.D., Chi, M.T.H.: The influence of tutor-tutee interactions on tutor learning. Instructional Science (in press)

Schwartz, D.L., Pilner, K.B., Biswas, G., Leelawong, K., Davis, J.: Animation of thoughts. In: Lowe, R., Schnotz, W. (eds.) Learning with Animation: Research and Implications for Design. Cambridge University Press, Cambridge (2005)

Silberman, M.: Active learning: 101 strategies to teach any subject. Allyn and Bacon, Boston (2002)

Wagster, J., Tan, J., Biswas, G., Schwartz, D.: How Metacognitive Feedback Affects Behavior in Learning and Transfer. In: The thirteenth International Conference on AI in Education, Marina del Rey, California (2007)

A Standard Method of Developing User Interfaces for a Generic ITS Framework

Mikaël Fortin, Jean-François Lebeau, Amir Abdessemed,
François Courtemanche, and André Mayers

Department of Computer Sciece, University of, Sherbrooke
{mikael.fortin,andre.mayers}@usherbrooke.ca

Abstract. This paper presents a new method to combine in the ASTUS framework the advantages of generic ITSs and their standards with the usability of their domain-specific counterparts when faced with the problem of user interface design. This method involves coupling the taught domain's semantics with the interface, defining scripts to explicit problem-solving step recognition and specifying how relevant data is gathered from that interface. The method is applied to a classic problem solving domain to highlight its potential.

Keywords: generic ITS, user interface, interface design, feedback, learning environments.

1 Introduction

From the perspective of the user interface designer, two forms of intelligent tutoring systems (ITS) can be distinguished. The first focuses on offering means to model new learning domains and their corresponding interfaces for use within a generic ITS at the cost of somewhat limited usability and sophistication. The second offer rich, usable interfaces but are domain-specific. Thus, providing generic ITSs with highly usable interfaces is a desirable end. This paper presents work aiming to achieve this through a new method. It is organized as follows. First, an overview of human-computer interaction in existing ITSs is presented, followed by a presentation of our framework. Next, we highlight issues related to interface design and introduce our method. We conclude by mentioning possible improvements.

2 Human-Computer Interaction and ITS

When studied in the context of intelligent tutoring systems (ITS), human-machine interaction has to deal with the important issue of semantics. A user interface designed for an ITS is the world in which the student's problem-solving will take place [1]. This means that there is a strong link between the user interface design and the semantics of the problem-solving environment around which an ITS is built. This issue is at the core of the ITS design process, because the interface through which the semantic knowledge is represented and the procedural knowledge is practiced determines the nature of the skills the student will acquire and their correlation with the

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 312–322, 2008.

domain that is being taught [1]. With an adequate representation of the domain and a well-designed interface, the tutor may then interpret the student's actions, evaluate them against a model of the solution and provide feedback in various forms when deemed necessary. The realization of these services, often called the inner loop [4], depends heavily on the information that is fed to it: correctly identified problem-solving steps.

At the threshold of the inner loop's implementation lies the significant problem of inferring meaningful steps from user actions on the interface. To accomplish that, the system must first detect that a step was completed, and then, gather information from the interface to associate parameters to that step. This process gives semantic value to user interactions, transforming them into steps that can be used by the tutor to assess the student's progress. Some existing ITS enforce a one-to-one relationship between user interactions and steps. One of them is the Cognitive Tutor Authoring Tools (CTAT), a generic framework that provides an interface builder enriched with « tutor-able » widgets [3]. Interacting with such a widget produces a *selection-action-input* triple, which is then tested against a set of constraints so it can be identified as the application of a rule prescribed by the model of the solution.[1] Other systems, not concerned with this constraint, have dedicated interfaces. An example is the Andes Physics Tutoring System, in which one specific user interaction is considered an indication that a step is completed (closing a dialog box, writing in a text field, etc) [2]. Information relevant to the step is then gathered from multiple widgets, and processed to produce the step's parameters. This process cannot be generalized easily, and it must be implemented for each distinct step.

CTAT provides reusable components and an easy coupling between user interactions and problem-solving steps. However, it does not allow a more coarse-grained step as is the case with Andes, in which many user interactions may be required to prepare the information relevant to the step. When building a framework for ITS design aiming to support any well-defined domain, this coupling needs to be standardized. Borrowing CTAT's solution to this problem would place a strong constraint on interface design: a one-to-one relationship between user interactions and steps. This constraint restricts the granularity of the user interface or the knowledge representation. A wrong grain size might yield a system with a user interface that is quite unlike the ones occurring naturally for the domain, or an imprecise knowledge representation that would be a gross approximation of what the student is inferring.[2] In order to correctly implement an ITS for a specific domain, it is crucial that the step size be determined by the knowledge that will be taught by the system, not by restrictions intrinsic to the framework. Lifting this constraint would also imply that some semantic knowledge must be associated with the interface. Hence, a standard method to specify how information can be extracted from it and transformed into parameters for a step may be developed.

Without a more flexible solution to step recognition than one user interaction for each step, or without any semantics explicitly attached to the interface, it would be difficult to implement an ITS with an interface as usable as Andes' without making some tradeoffs in the knowledge representation.

---

[1] http://ctat.pact.cs.cmu.edu/pubs/TDKTutorialSlides.ppt

[2] See VanLehn, K., & Niu, Z. (2001), p. 181 [5].

3 The ASTUS Framework and Its Interfacing Agent

Linking the user interface to the knowledge representation manipulated by the tutor is the main task of one of its modules, the interfacing agent (IA). Our assumption is that providing a semantically rich coupling with the interface will allow better assessment of what the student does and help the tutor implement various forms of feedback on the interface. This section explains the role of the IA within the ASTUS framework.

3.1 Knowledge Representation

The ASTUS framework makes a distinction between semantic, procedural and episodic knowledge and behaves like a model-tracing tutor. Semantic and procedural knowledge is used to define a problem-solving model with correct and erroneous paths. This model of the solution is the template used to instantiate solution graphs. Procedural knowledge is represented by goals and procedures. Each goal may be satisfied by many procedures, and each procedure can be either a script of subgoals (a complex procedure) or an atomic action that can be detected by the interface as an elementary step towards the solution (a primitive procedure). Goals and procedures are organized in a hierarchical graph, its root being the intention to solve the problem and its leaves being primitive procedures [6]. Each goal and procedure has a set of parameters, represented by semantic knowledge. Such knowledge is represented by concepts, which are constructs combining attributes containing either raw data or links to other concept instances.

3.2 The Interfacing Agent

The tutor in ASTUS is a set of four interlinked processes (see figure 1). These processes, or agents, represent the usual modules as seen in many ITS [7]. The tutor is

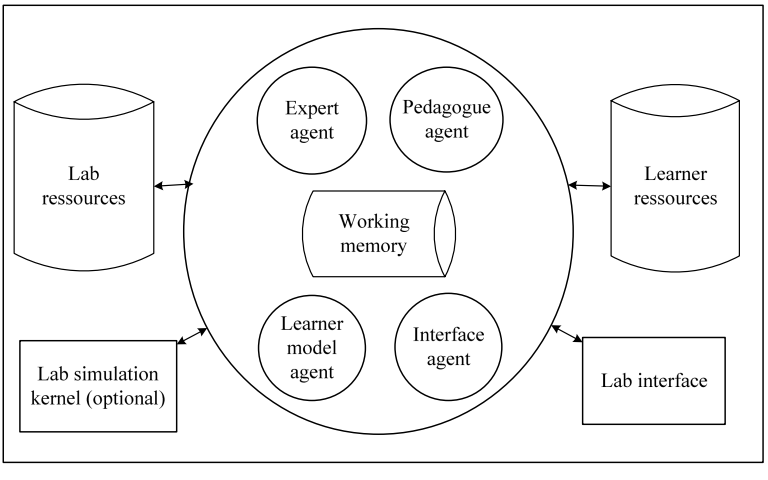


Fig. 1. The ASTUS framework

designed to work with different labs which are domain packages. These packages encompass information that is specific to a domain such as its knowledge representation and its interface.

The expert agent's (EA) services are all related to the interpretation of the procedural knowledge graph associated with the domain. The learner model agent's (LA) services are dedicated to the evaluation of the learner's experience. The pedagogical agent (PA) uses information from the learner and expert agents to decide what feedback to provide to the learner. The various forms of feedback are then carried out by the interfacing agent (IA), which also interprets the learner's actions on the interface

The interfacing agent handles the main aspects of communication with the interface as seen with many ITS: (1) translate actions on the interface into steps recognized by the model, (2) provide the tutor with means to produce interface actions, and (3) maintain a representation of the interface [1]. This last aspect is managed by associating semantics to the interface's parts in order to make the tutor aware of their state. In our framework, the interface provides problem-solving tools that have a semantic representation in the tutor's working memory. Furthermore, widgets that will change the state of the interface send signals that the tutor may interpret and combine to determine if a step has been achieved. Finally, having a conceptual representation of the interface along with detailed scripts for every problem-solving step allows the tutor to modify the interface in a meaningful way and reproduce the interface actions the learner has to perform to achieve a step.

4 Key Issues of Interface Design in ASTUS

There are three important issues that must be resolved for an interface to be properly designed using the ASTUS framework. This section explains the method that was developed to resolve these issues, and how it determines the design of an ITS' interface.

The first issue concerns the representation of semantic knowledge on the interface. Once properly handled, this aspect of the design will allow the tutor to have a representation of the interface in its working memory. That way, it can remain aware of the state of the system as a whole, including the interface. We resolve this issue by defining a *view* (see section 4.1) for each concept that will be reified on screen.

The second issue is the need to define how each primitive procedure is detected as completed on the interface. In order to resolve this in a generic ITS framework and lift the one action per step constraint, we associate a *script* (see section 4.2) of user interactions to each primitive procedure.

The last issue is the transformation of data scattered on the interface into a primitive procedure's parameters. Since we aim to let the designers of an ITS build a natural, usable interface, the many parts that constitute that interface will contain data in a form that is not always easily associated with a procedure's parameters. To solve this problem, we propose a standard method to collect information from the interface, through the use of *extractors* (see section 4.3).

4.1 Views: Linking Working Memory with the Interface

The tools that a student will use to solve a problem have semantic value within the ASTUS framework. These semantics only define a conceptual interface through the roles that the tools will play in the problem's resolution. The real interface, in terms of a layout of components or widgets, is implemented through a careful combination of normal GUI programming and definition of the views associated with some of the concepts from the domain's knowledge representation. When a concept is destined to be seen or manipulated by the student on the interface, its view must be designed. A view is a construct that organizes the coupling between a concept and its representation on screen. Having such an aspect to a concept allows the tutor to understand when the student is manipulating it, and facilitates the integration of feedback with the parts of the interface that are relevant to the situation.

The reification of a concept is typically built by using original components from the chosen library (our examples were made with Java's Swing) or customized components. Once a part of the interface that represents a concept is designed in this way, it is associated to a view, whose job is to manage its semantic aspects. The view contains the set of all interface parts that are used to reify a concept properly on screen, along with event handlers that are used to detect meaningful interactions. Some of these parts are only used to show a portion of the concept, while others are also used to collect information when the learner manipulates them. The composition of a view in terms of parts of the interface is often similar to the composition of the concept it reifies: its attributes are either reified by views when they contain links to other concept instances or by a set of widgets if they contain raw data.

A hexadecimal subtraction lab may have an interface that looks like figure 2. The learner may specify his answer in the bottom row and realize borrowing and conversion operations by clicking on the relevant digit. In that example, three important concepts are reified in the interface: columns, operands and digits. The first concept represents a subtraction problem for one column, and has a top *Operand* (its

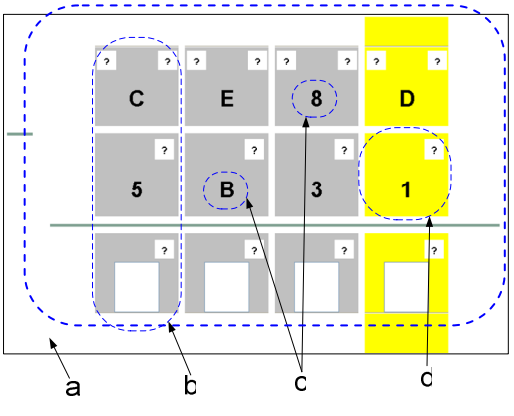


Fig. 2. The interface of an hexadecimal subtraction lab and some of its views. a) View of the *Subtraction*. b) View of a *Column*. c) Views of some *Digits*. d) View of an *Operand*.

minuend),a bottom *Operand* (its subtrahend) and an answer field (its difference). Each of these parts is a relation to an *Operand*. The view of an *Operand* contains tools used to apply borrowing and conversion operations to it, and the view of its current *Digit*. Those are defined as having a value attribute, a corresponding base-16 symbol. A *Digit*'s view is simply a panel containing a label with the value written on it. Another concept in this lab is the subtraction itself, defined as a set of columns. Its view is then defined as a set of column views.

The composition defined in the views does not always directly translate to a simple nesting of the components in the interface. In order to produce the layout shown on figure 2, the view of a column cannot be contained in one interface component (a panel with the top and bottom digits as well as the answer), because of restrictions stemming from the Swing toolkit. It is easier to regroup the numbers by row in order to produce an adequate layout. Here, the view of the subtraction concept should define how the parts of the column concept will be laid out in the interface. To achieve a flexible design, the parts that constitute the view of a concept do not need to be organized in a convenient, encapsulated space on screen. A view might be a collection of parts scattered across the interface. The semantic aspects of the interface implemented by views do not dictate how the interface will be laid out, only what the interface means.

A view also defines the aspect and the behavior of every instance of a concept in the interface. While most instances of concepts have one view object associated with them, it is possible for an instance to be viewed many times on the interface. In the example above, every hexadecimal digit is only instantiated once in the tutor's working memory; they are constants that are used throughout the problem. However, every digit may be viewed many times on screen. The definition of a concept and its view do not place any restrictions on the number places where an instance of a concept may be reified in the interface.

In order for this lab to work, the effect of picking a digit in an answer field must be defined. The only primitive procedure in the example involves specifying a digit for an operand. The interpretation of the signals from a widget like this example's buttons and text field to determine if the student has completed a primitive procedure is the subject of the next section. The last section explains how we may infer the parameters associated with the procedure, like the column and the digit from the example, from these signals.

4.2 Interactions, User Actions and Scripts: Determining How Steps Are Realized

The interfacing agent implements a generic method to identify completed primitive procedures without enforcing the constraint that one signal from the user interface has to be interpreted by the tutor as one completed procedure. This is achieved by identifying possible user actions on the interface and then associating a script that prescribes a sequence of those actions to each primitive procedure.

Before a click from the student on a widget can be interpreted by the tutor, it needs to be considered in the broader context of what the student might be trying to achieve. Every basic action on the interface is called an *interaction*. These are directly detectable with the toolkit's resources (property changes or mouse actions on widgets, etc) and usually don't carry any meaning on their own, except the fact that they have been

detected by a specific view's event handlers. An interaction object contains the information on such an event: what has been clicked, modified, entered, etc.

When designing the interface for an ITS, the role each interaction will play and how it is associated with what the student is doing must be defined. While implementing ASTUS' generic interfacing agent, we have determined that the user's actions belong to one of the following categories: selections, inputs, required interactions, and noise. Some of these actions are complex sequences of basic interactions, with their own associated script. Inputs, selections and required interactions are all meaningful actions that make up the parts of a script associated to a primitive procedure.

Selections are interactions or groups of interactions that the student does to indicate a choice among a number of existing instances. The subtraction lab from the previous section has an example of a selection: the only possible answers are between 0 and F, and the interface will not allow the user to specify any other answer. Thus, entering a symbol in the answer field's text box is the selection of one of the instances of the Digit concept that are already present in the tutor's working memory.

Inputs occur when the learner uses the interface to specify a concept instance that is not part of a predetermined set. In many cases, inputs are realized with only one interaction, but they may also be carried out by a set of interactions (using a set of widgets to specify the many parts of a complex concept, for instance). An input always results in the creation of an instance of a concept that is a representation of what the student has inferred.

Required interactions are simple manipulations that have no semantic value of their own but are required for the user to complete a set of interactions that has a higher purpose. For example, an interface might require the user to enter a value in a text field, and then press a button to confirm that the value is right. In order for this input event to be completed, the button must be pressed. The meaning of that interaction is only seen when it is done in conjunction with another interaction. No information other than the fact that the button was pressed may be extracted from that interaction. It is only required that this event be detected in order for the input to be completed.

Noise is the set of all interactions that cannot be interpreted as parts of complete and significant actions. For instance, if an input is realized by entering a value in a text field and then clicking a button to confirm the value, multiple clicks on the button without entering a value in the text field are considered noise. Such interactions may represent mishandling of the interface's controls, slips on the part of the student, or incoherent behavior.

Once the behavior of the user is categorized, it must be placed in the context of the realization of a primitive procedure. To specify how a primitive procedure is realized, we define a script. It prescribes the set of user actions that must be carried out in order for a procedure to be considered completed. When individual user actions are detected by the interface, the interfacing agent determines if the actions carried out so far may be generated by one of the scripts. The algorithm that is used to determine if there is a match and the language used to define scripts determine the level of freedom the learner may have while using the interface. Matching sets of actions that may contain noise to sets of actions prescribed by a script is a complex issue. It may be solved by a simple matching algorithm and rudimentary scripting language in some cases, but in interfaces with rich content and a high level of interactivity, these tools need to be quite powerful.

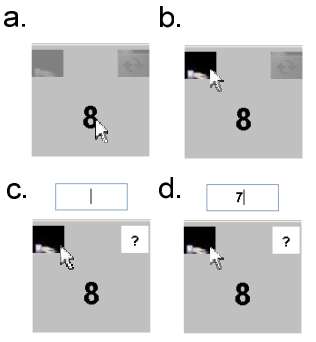


Fig. 3. User interactions involved in the borrowing operation. a) Press on a Digit. b) Drag over the borrowing symbol. c) Release on the borrowing symbol. d) Enter a number in the text field.

Figure 3 illustrates the sequence of interactions that is involved with the conversion of a number in the subtraction lab (presented in section 4.1). This is an example of a situation in which scripts are used and it demonstrates the role that some types of user actions play in the implementation of an interface.

The interactions the learner should do are the following: press on the *Operand* he wishes to borrow from (figure 3 a), drag the mouse to the hand symbol (figure 3 b), release over the hand symbol to activate borrowing (figure 3 c), enter the result of the borrowing operation in the text field (figure 3 d). The first three interactions are used to select which *Operand* we wish to borrow from. The last interaction is used to specify the result of the operation. However, the text field does not allow the learner to enter anything else than a hexadecimal digit, which are all known to the tutor. This means that the learner selects one of the existing digits rather than input a new value.

The primitive procedure that the student accomplishes through these actions is to indicate the result of a borrowing operation on a particular operand of the subtraction. Its script can then be defined like this:

1-Select an *Operand* to borrow from (a selection)
2-Select the *Digit* resulting from the borrowing operation (a selection)

If both these actions are accomplished, the interfacing agent can determine that the primitive procedure has been completed and then extract its parameters from the interface. The first selection is a complex action that also has its associated script. It involves three simple interactions:

1-Press on the desired *Operand* (a required interaction)
2-Drag the pointer to the borrowing button (a required interaction)
3-Release over the borrowing button (a required interaction)

This script is a sequence of three required interactions. Once they have been detected in the correct order, the interfacing agent can determine that the selection of an *Operand* has occurred. An interface with such detailed manipulations makes it harder for a tutor to demonstrate a part of the solution to the student. With our implementation, however, the interfacing agent is aware of enough signals to reproduce detailed interaction on the interface and thus better integrate its feedback to the learning

environment. Moreover, changing the way borrowing is done on the interface does not have any effect on the meaning of the procedure for the tutor and its place in the knowledge representation.

4.3 Extractors: Bridging the Gap between Semantics and Interface Data

Parts of the interface are associated with semantic knowledge through the implementation of views. Sequences of meaningful actions are associated with procedural knowledge with the definition of scripts. However, some semantic knowledge is not represented by those means: the parameters of a primitive procedure.

Each primitive procedure has a set of parameters, which will be instances of the concepts defined in the semantic knowledge. When a student interacts with the interface and produces a set of actions that matches a procedure's script, she specifies all of the procedure's parameters along the way. The actions the student has done must carry enough information for the tutor to infer which instances may become the values of the parameters. Such information may be gathered from any interface component and can bear any meaning. This association of a particular semantic meaning to data gathered from the interface is done through routines implemented by extractors.

There are two levels of extraction to consider in our framework. The first, lower level involves the transformation of data from the interface's widgets into concept instances or attribute values of such instances. This type of extraction is only used to define inputs and selections. The second, higher level involves associating instances already identified or built by inputs, selections or required interactions to the parameters of a procedure. This type of extraction is used to determine how the user's actions are semantically linked to the parameters of a procedure. High-level extractors are usually simple and integrated with the language used to specify scripts. Low-level extractors, however, may be specific to a given domain. If the designers of the interface are free to integrate semantic knowledge to their interface in any way they wish, then situations that are unique to one domain and its interface will arise. Fortunately, most cases can be resolved with the use of a standard library of low-level extractors.

In order to allow high-level extraction to take place, the actions that may appear in a primitive procedure's script must already identify one or many instances that they are related to. Thus, selections, inputs and required interactions all have instances associated with them. In all three cases, the actions have an *owner*, which is the concept instance whose view detected the action. Additionally, inputs specify what instance was *built*, while selections specify what instance was *selected*.

The primitive procedure studied in section 4.2 is named *Borrow*. The script associated to this procedure has two elements: the selection of an *Operand* in a *Column*, and the selection of a *Digit* in an *Operand*. A declarative representation of the script associated with the procedure would have this form:

```
Procedure Borrow (Column c, Operand o, Digit d)

Script:

    1-Selection where
          owner is a Column
          selected is an Operand
```

```
      2-Selection where
            owner is an Operand
            selected is a Digit


Extraction:

      1-Parameter c is the first element's owner.
      2-Parameter o is the first element's selected.
      3-Parameter d is the second element's selected.
```

This represents how high-level extraction takes place. Each parameter is associated with an instance manipulated by one of the script's elements. Complex scripts that are defined by multiple actions do not pose a serious extraction problem: parameters are simply gathered from any of the instances that were manipulated by the actions. Low-level extraction, on the other hand, is found with the definition of the second selection action:

```
Selection of a Digit in an Operand

Script:
    1- A text modification event from the text field at
       the top of an Operand, trapped by its view's
       event handler.

Extraction:
    1-owner is the Operand associated with the view.
    2-Selected is a Digit found by applying the method
      ExtractDigitFromTextField.
```

In this case, a specialized function must be used to determine how the symbol entered in the text field translates to an instance of the *Digit* concept. This low level extractor could be implemented in a number of ways, depending on how a character from the text field relates to its corresponding *Digit* instance.

Scripts and extractors establish a very explicit connection between the data that is kept in the low levels of the interface's model and the semantically rich data that is kept in the tutor's working memory. This connection enables the tutor to use a script and its extractors to generate a set of actions from a primitive procedure and its parameters and execute them on the interface. It also allows the detection of actions that undo a step, so the tutor knows when the student has backtracked along his solution path. The tutor may additionally carry out undo actions on the interface if it considers that the student has progressed too far on an erroneous solution path.

5 Conclusion

The presented examples showed that offering rich and usable interfaces to a generic ITS framework is possible, with the subsequent advantages. However, the design standards are more elaborate, forcing interface designers to take into consideration the semantics of the domain to be taught, from the early stages. Future improvements will

focus on a powerful scripting language to deal with noisy behavior and complex step recognition. Additionally, our method will facilitate the implementation of many forms of feedback involving precise interface manipulation by the tutor, such as redlining.

References

1. Anderson, J.R., Corbett, A.T., Koedinger, K.R., et Pelletier, R.: Cognitive tutors: Lessons learned. Journal of Learning Science 4(2), 167–207 (1995)
2. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The Andes physics tutoring system: Lessons Learned. International Journal of Artificial Intelligence and Education 15(3), 1–47 (2005)
3. Aleven, V., McLaren, B.M., Sewall, J., Koedinger, K.: The Cognitive Tutor Authoring Tools (CTAT): Preliminary evaluation of efficiency gains. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 61–70. Springer, Heidelberg (2006)
4. VanLehn, K.: The behavior of tutoring systems. International Journal of Artificial Intelligence in Education 16(3), 227–265 (2006)
5. VanLehn, K., Niu, Z.: Bayesian student modeling, user interfaces and feedback: A sensitivity analysis. International Journal of Artificial Intelligence in Education 12(2), 154–184 (2001)
6. Fournier-Viger, P., Najjar, M., Mayers, A., Nkambou, R.: A Cognitive and Logic based Model for Building Glass-box Learning Objects. The International Journal of Knowledge and Learning Objects (IJKLO) 2, 77–94 (2006)
7. Wenger, E.: Artificial intelligence and tutoring systems: computational and cognitive approaches to the communication of knowledge. Morgan Kaufmann Publishers, Los Altos (1987)

Helping Teachers Handle the Flood of Data in Online Student Discussions

Oliver Scheuer and Bruce M. McLaren

Deutsches Forschungszentrum für Künstliche Intelligenz, Saarbrücken, Germany
Oliver.Scheuer@dfki.de, bmclaren@dfki.de

Abstract. E-discussion tools provide students with the opportunity not only to learn about the topic under discussion but to acquire argumentation and collaboration skills and to engage in analytic thinking. However, too often, e-discussions are not fruitful and moderation is needed. We describe our approach, which employs intelligent data analysis techniques, to support teachers as they moderate multiple simultaneous discussions. We have generated six machine-learned classifiers for detecting potentially important discussion characteristics, such as a "reasoned claim" and an "argument-counterargument" sequence. All of our classifiers have achieved satisfactory Kappa values and are integrated in an online classification system. We hypothesize how a teacher might use this information by means of two authentic e-discussion examples. Finally, we discuss ways to bootstrap from these fine-grained classifications to the analysis of more complex patterns of interaction.

Keywords: Educational data mining, Natural Language and Discourse, Architectures, Machine Learning in ITS.

1 Introduction

E-discussion tools provide students with the opportunity not only to learn about the topic under discussion but to acquire argumentation and collaboration skills and to engage in analytic thinking. Tools such as Digalo[1] and Free Styler[2] [1] allow students to use a shared workspace to present ideas, debate and argue with one another, and ask questions. Visual languages consisting of typed text boxes and links provide additional scaffolds that help students structure the way they think about and discuss a topic. Nevertheless, too often discussions are unfruitful: students misuse the tools for private conversation instead of staying on topic, contributions lack critical reasoning, arguments and questions of other participants are ignored and some students don't participate at all, while others dominate discussions. Thus, there is a need for active help and guidance, be it from a machine tutor or a human teacher / moderator.

Our focus is on helping a teacher moderate a classroom of students using e-discussion tools in which the students comprise multiple discussion groups. The teacher can bring to bear his or her experience and moderation expertise to steer the

[1] http://dito.ais.fraunhofer.de/digalo/
[2] http://www.collide.info/software

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 323–332, 2008.

discussions when problems occur and provide encouragement when discussions are productive. However when multiple e-discussions occur simultaneously, a single teacher may struggle to follow all of the discussions. To direct the teacher's attention to the 'hot spots,' software tools that pre-process, aggregate, and summarize the incoming flood of data could be extremely valuable. In this respect, the task is reminiscent of that faced by systems that monitor power plants and medical patients where vast amounts of raw data are analyzed, filtered and/or condensed to support human decision making.

The ARGUNAUT project follows this approach with the ultimate aim of supporting teachers as they guide multiple, simultaneous e-discussions. Two analytical processes are particularly prominent in ARGUNAUT's analysis of e-discussions: (1) the "Shallow Loop" focuses on surface features that can be computed in a straightforward way (e.g. the total number of contributions per student) and (2) the "Deep Loop" evaluates situations requiring more complex analysis, combining textual, sequential and structural information to classify more abstract aspects of discussion (e.g. on-topicness, reasoned claim). The Deep Loop inference mechanism is based on machine-learned classifiers, developed from our corpus of annotated discussions, and is the focus of this paper. Our long-term goal is to use the classifiers also to automate support and feedback for collaborating students in typical intelligent tutoring fashion.

In this paper, we provide an overview of the ARGUNAUT project and an in-depth treatment of the Deep Loop Classification system. We describe the approach we have employed to develop the Deep Loop classifiers and present the quantitative results we've achieved, particularly emphasizing progress made since the work reported in [2]. Finally, we provide specific examples and discuss how a teacher might use the Deep Loop classifiers to identify good and bad discussion situations.

2 Related Work

The most relevant work to ours is by Rosé and colleagues, who have developed the text analysis tool TagHelper [3], also used in our work. Originally, they aimed at freeing corpus analysts from the tedious task of manually coding large amounts of data, rather than analyzing online discussions, which is our goal. In one application [4] they analyzed a corpus of 1,250 coded text segments along multiple dimensions of argumentation in order to derive machine-learned classifiers. Some of the phenomena of interest in their work, like argument-counterargument chains and grounded claims, are quite similar to the categories we are interested in. They achieved acceptable Kappa values of 0.7 or higher for six of seven dimensions. More recently, they developed an approach to providing dynamic support to dyads collaborating on a problem-solving task [5]. Similar to our approach, they perform online analysis of textual communication data, in their case, chat data. In contrast to our approach, their analysis results are not displayed to human teachers but are instead used to trigger automatic interventions in the students' activities. An empirical study showed significant learning benefits in terms of analytical knowledge and conceptual understanding. when dynamic support is provided

Goodman et al. [6] also have developed a machine-learning approach to support collaborative problem solving. Peer groups work together on a problem in the domain of object modelling techniques (OMT). Their collaboration takes place within a shared whiteboard (similar to the shared workspaces in ARGUNAUT) in which diagrams (e.g. class diagrams) have to be constructed. Peers communicate via a text chat with a sentence opener interface; task management is supported by an agenda tool. The system evaluates aspects concerning domain (e.g., domain knowledge of peers), task (e.g., progress in solving the task) and, similar to our objectives, possible problems in the collaboration process (e.g. unanswered questions). The sentence opener interface plays a critical role; it is used to automatically assign a dialogue act classification to each chat contribution. These dialogue acts are used as a meta-level description of the discourse and serve as features for machine-learning analyses, bypassing the complicated task of natural language processing. The provided support is twofold: Some of the results are displayed immediately to the peers via meters, while direct support is provided by means of an artificial peer agent that verbally interacts with the participants.

3 The ARGUNAUT Approach

In ARGUNAUT students discuss and debate questions within a shared workspace on different networked computers in synchronous fashion [7, 8]. A discussion starts with a shape containing the question to be discussed. Usually, controversial topics are chosen (like experiments on animals, abortion) to allow students to take different positions and to promote a lively exchange of arguments. Students contribute by adding shapes, entering text into the shapes and connecting the shapes by links. Shapes and links are not just simple text boxes and connectors; they have types and comprise a visual language: There are shape types to express claims, arguments, questions, etc, and link types to establish supporting and opposing relations.

A teacher can monitor multiple ongoing discussions in parallel using a tool called the "Moderator's Interface". Here, important aspects of the discussion are displayed in the form of "Awareness Indicators". As discussed above, "shallow indicators" can be computed in a straightforward fashion (e.g. the number of contributions per user); "deep indicators" result from a more sophisticated machine learning-based analysis. Currently, two types of deep indicators are computed: *Shape-level* indicators reflect characteristics of a single contribution (e.g. whether this contribution contains a reasoned claim); *paired-shape indicators* reflect characteristics of two linked contributions (e.g., whether two shapes constitute a contribution-counterargument pair). Six classifiers for deep indicators are currently available: "Reasoned Claim"[3] and "Topic Focus" at the shape level, "Question-Answer", "Contribution followed by Question", "Contribution followed by Counterargument" and "Contribution followed by Supporting Argument" at the paired-shape-level. (Note that this is four more classifiers than were available in an earlier reporting of our work [2].)

Although the Moderator's Interface has not yet been used in a real classroom, we anticipate that the combination of shallow and deep indicators will enable teachers to more effectively and efficiently moderate multiple, simultaneous discussions.

[3] Note that in previous work, this category was referred to as "Critical Reasoning".

4 Deep Loop Classifiers

The classifiers were developed using a three-stage process: First, a coding scheme was developed for categories of interest and the data was coded accordingly. Second, the data was translated into a format amenable to standard machine-learning algorithms. Third, experiments with a multitude of machine-learning techniques were carried out in order to derive the most effective classifiers. The resulting classifiers have been integrated into the Classification Web Service to enable a teacher to run classifications online.

Originally, we were interested in twelve discussion categories but after initial experiments we focused our efforts on the six categories described below. There are several reasons why we limited our scope: One category did not have sufficient inter-coder agreement. Other categories were shown to be less promising after some initial machine-learning experiments, partly because of an unbalanced class distribution and too few examples for one class, two problems that are well known for their detrimental effects on machine learning [9, 10].

4.1 Data Description

The first step in building an example corpus was to collect data and code this data with the categories of interest (Deep Loop indicators). The data was collected during real classroom sessions in Israel and the U.K. Because the discussion language in Israel is Hebrew, it was necessary to translate these discussions into English before coding. This was done for experimental purposes only; in the longer term our intention is to use customized versions of TagHelper applied to the language of interest.

After the pedagogical experts on our team agreed on a set of categories, coding instructions were developed, consisting of detailed explanations of when a code applies and additional illustrative examples for further clarification. The final corpus presented here is the product of several coding iterations carried out by our pedagogical experts in Israel and the U.K. To determine the reliability of the coding procedure, inter-rater reliability was computed by means of the Kappa statistic, yielding acceptable values (near or above .70) for all but one category. More details concerning the coding procedure can be found in [2].

The final corpus comprised data of 72 discussions covering ethical questions ('Should we clone humans?') as well as questions of opinion and fact ('How does the use of ICT affect learning experiences?'). In the end, we had 1,260 annotated shapes and approximately 1,000 annotated shape pairs. All but one category (Reasoned Claim) show a clear majority of one class, with proportions ranging between 75 % and 85 % of all instances.

4.2 Machine Learning Experimentation and Results

As a first step, the data was cast in a form suitable for machine learning. We used a data-centric approach by encoding as much information as possible in feature-value form, without considering the specific categories of interest, in hopes that the inference mechanism itself would choose the relevant pieces of information. Shapes and paired shapes were analyzed in terms of structural properties (shape and link types,

incoming and outgoing links), sequential properties (chronological sequence of shapes) and textual properties (textual content of shapes). The textual analysis was done using TagHelper, discussed earlier [3]. We pre-processed the data by reducing terms to their word stems and removing stop words. We extracted unigrams (single terms), bigrams (pairs of consecutive terms), part-of-speech bigrams (two consecutive part-of-speech classes), punctuation marks and text lengths.

The experiments were conducted with RapidMiner (formerly known as "Yale"), a machine learning toolkit offering a wide range of methods for data pre-processing, machine learning and validation [11]. We experimented with a variety of learning algorithms using different feature combinations, estimating the reliability of our classifiers by cross-validating data from one discussion (test set) against the data from the remaining discussions (training set). Because data from one discussion was never in the training and test set at the same time we avoided intra-discussion dependencies and bias. We measured the reliability using Cohen's Kappa [12] (a criterion more appropriate than the widely used error rate and accuracy measures which are both vulnerable to unbalanced class distributions). A Kappa value of 1.0 signifies a perfect classifier, a Kappa value of 0 means a classifier performing equally well as a trivial classifier that always chooses the majority class and Kappa below 0 means a classifier even worse than the trivial majority voter.

Support Vector Machines (SVM), Boosted Decision Trees and Decision Lists proved to be the most effective machine-learning algorithms. We achieved Kappa values ranging from .60 to .71, which can be interpreted, according to [13], as moderate (1 category) to substantial agreements (5 categories) between the human annotations and machine-learned model[4]. Because these six classifiers performed reasonably well, we integrated them into the classification web service.

5 How the Deep Loop Classifiers Could Help Teachers

In this section, we turn to the practical application of the Deep Loop. As discussed earlier, we have taken the approach of first having pedagogical experts on our project team identify discussion categories of interest, annotate instances of those categories, and then apply machine-learning techniques to create classifiers for those categories. However, at this stage of our project we have received only minimal feedback from teachers on the pedagogical value of the classifiers. The following question then arises: what could a teacher do with the results of the six classifiers described above? We address this question by showing and discussing the Deep Loop classifiers applied in two authentic discussions. We hypothesize how a teacher might interpret actual Deep Loop results to recognize one discussion situation as fruitful and another as requiring support.

In general, we expect a fruitful discussion to be lively, with (close to) equal contributions by all participants. Questions should be answered and claims should be

[4] There is no universal threshold for an acceptable Kappa value, the decision regarding what is acceptable and what is not depends on domain and application. A more rigorous threshold is given by Krippendorff [14] who recommends a value of .67 as the minimal acceptable inter-rater agreement for content analyses. Given that a teacher who is aware of uncertainties and possible misclassifications will ultimately interpret the provided Deep Loop results, we consider the slightly more generous interpretation as sufficient.

backed by supporting arguments. Contrary positions should be acknowledged and answered with counterarguments. Participants should always be open to persuasion when more compelling arguments than their own are raised. Such quality criteria are supported by the CSCL literature, see for instance [15] for a more detailed account of criteria for assessing the quality of collaboration.

Figure 1 shows a situation that matches much of this description (CASE 1). The question raised in this discussion was whether abortion is ethical when it is known that the child will be born with a serious handicap. The question is introduced through a fictional story that has been read by the students prior to the discussion. The original question posed by the teacher is marked by a double-framed box. The figure shows two students involved in a dialogue where student 1 (solid boxes) takes a position against and student 2 (dashed boxes) in favor of abortion. Student 1 starts by stating his opinion (contra abortion) and backs his position by pointing to the human right to life. Student 2 counters that both parents and child will suffer, that a lot of money will have to be spent, etc. Student 1 gives as a counterargument that, provided enough money for treatments is available, the family can nevertheless live a good life. Student 2 objects that possibly the family does not have enough money. Finally, student 1 closes the thread by integrating both views: An abortion might be acceptable if the family does not have enough money, otherwise not. Regardless of one's personal position on abortion, this thread exhibits positive discourse characteristics, including:

(1) both students react to the position of their counterpart resulting in an argument-counterargument chain (3 CCA paired-shape classifications),
(2) the chain contains a considerable amount of reasoned claims (3 RC shape classifications),
(3) all student contributions are on-topic (5 TF shape classifications),
(4) a posed question was answered (1 QA paired-shape classification), and
(5) the thread ends with an integration of both views.

Figure 2 shows a discussion in which a teacher's intervention might be helpful (CASE 2). This discussion addresses the same topic as example 1, namely abortion under special circumstances. To the left of the teachers' assignment (double-framed box) there are contributions in favor of abortion in this situation. In summary, the

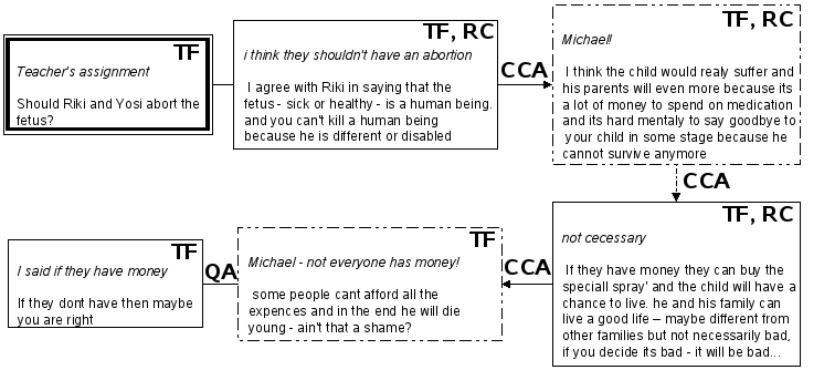


Fig. 1. Positive discussion situation (CASE 1)

main arguments here are that both child and parents will suffer, and that it will be easier for the parents to abort a still unborn fetus than seeing their child die later on. On the right-hand side we see contra-abortion contributions. The main arguments here are based on a human's right to life and religious convictions. Although here, too, almost all of the contributions are on-topic and the arguments are valid, the discussion suffers from a lack of interaction between participants. The arguments are made in isolation, almost exclusively linked to the original question. Only the three shapes at the lower right show some rudimentary interaction between the participants. Consequently, our classifiers detect only one contribution-counterargument pair in the entire discussion, in contrast to the three contribution-counterargument pairs in CASE 1 (and there we only show a fraction of the entire discussion graph). At this point, a teacher might find it valuable to intervene, encouraging the students to react to one another's positions and perhaps come to an integration/synthesis of multiple views.

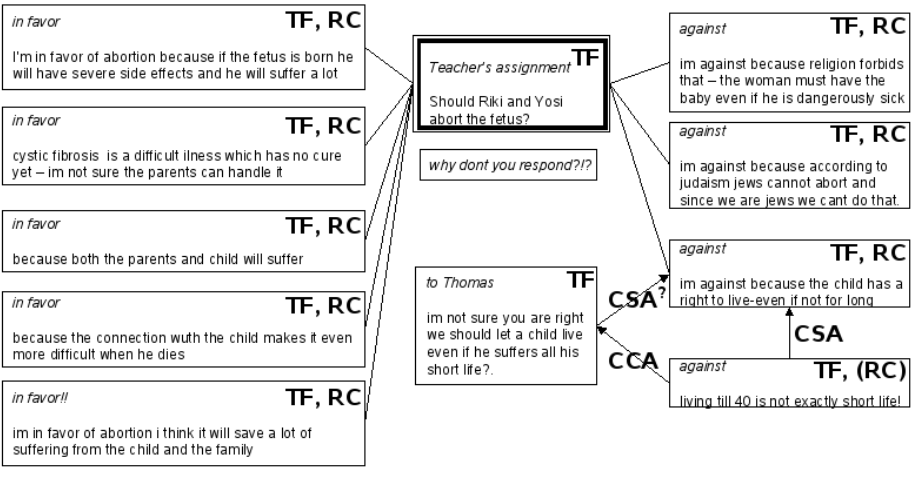


Fig. 2. Discussion situation in which a teacher might want to intervene (CASE 2)

With the current classifiers, a teacher might be able to use the information of Figures 1 and 2 in both a quantitative and a qualitative manner. Quantitative usage is supported in the Moderator's Interface through display within each ongoing discussion the number of occurrences of each classification type. In this way, a teacher can easily detect that CASE 1 contains a high proportion of counterarguments, whereas CASE 2 has a noticeable lack of this discussion element (there are 12 shapes and only 1 CCA relation), indicating that intervention may be helpful. Qualitative evaluation might also be helpful, as a purely quantitative analysis might point to critical situations but can also be deceiving because dynamic aspects of the discussion are no longer visible. For instance, discussions might be in need of intervention, even with the presence of many 'reasoned claims', as exemplified in CASE 2. Inspecting this situation on the level of *individual* classifications reveals that virtually all of the contributions refer to the original question shape and not to the shapes created by the participants. The students here are only enumerating supportive and opposing arguments

but do not *deepen* their understanding of the space of debate by arguing on arguments and negotiating the meaning of underlying concepts [16].

Furthermore, a qualitative examination – or visual impression – might provide an idea of what is going on without reading the contents. There might be controversial discussion with lots of reasoned claims and argument-counterargument pairs, situations in which students don't critically evaluate their peers' opinions, manifested through a lack of both supportive and opposing arguments, and places in which an accumulation of off-topic contributions suggests a drifting from the discussion. Once such regions are identified, a teacher can read the contributions in more detail and intervene as required.

Our approach enables a teacher to look at a discussion on different levels of detail: at the highest level information is summarized (and maybe also distorted), at the middle level the structure of interaction patterns is preserved but still abstracted from specific content, and at the lowest level the full information is offered without loss of veracity but at the possible cost of detail "overload."

6 Discussion, Conclusions, and Future Work

As we have seen, machine-learned classifiers can compute useful aspects of e-discussions even without an in-depth (i.e., semantic) analysis of natural language. We succeeded in deriving six classifiers that analyze discussion situations in terms of structural, chronological and shallow text characteristics and assign categories like "reasoned claim" and "contribution-counterargument" to the analysis units. The classifiers are integrated within the ARGUNAUT system and allow teachers to analyze simultaneous discussions online. We discussed how a teacher might use the system with two authentic examples. Our initial results are quite encouraging, but there are still open questions.

One crucial question is how far our classifiers generalize beyond the training corpus. Clearly, we cannot claim that our data sample has been drawn randomly from the population of *all* possible discussions. Although we have collected a considerable amount of discussion data, the number of covered topics is still somewhat restricted. Especially our 'topic focus' classifier might incorporate idiosyncrasies from the topics being covered in the training corpus and suffer when applied to different topics. Other categories, like "reasoned claim", "question-answer" and "contribution-counterargument", are largely topic-independent and may not be vulnerable to such dangers.

The real measure of success for any computer program is successful use by its intended audience working on tasks for which the program is intended for. Consequently, the logical next step is to test the Deep Loop classifiers 'in the wild' and to collect feedback from teachers charged with monitoring multiple discussions simultaneously. This will help us find answers to the question of which categories help (and which do not), which additional (missing) categories may be of interest, whether the visualization of the classifications is appropriate, and whether the reliability threshold we've adopted (Kappa > 0.6) is strict enough for this specific domain of application. Such feedback will help us to move towards a real usable system.

Currently, as demonstrated by the two examples above, the classifications might help a teacher find patterns of good and bad discussion situations (e.g. commission and omission of argument-counterargument chains). An alternative to such qualitative analyses is

for the classifications to be used in a quantitative way to *summarize* discussions along categories of interest. A first step in this direction has already been done: the total number of positive classifications is displayed to the teacher. We envision going much further by providing comparative statistics, in the form of appropriate visualizations (like bar charts or pie charts), showing, for instance, that 40 % of all links in discussion *X* represent an argument-counterargument relation whereas only 20 % a argument-supporting argument relation. A final step might be to infer automatically qualities of the whole discussion by analyzing its profile in terms of shape and paired-shape classifications. One could define a model using rules such as "If more than 30 % of all links define an argument-counterargument relation then the discussion qualifies as *controversial*". But of course, hand-crafting such a model requires superior human judgment and expertise. Alternatively, one could use inductive inference. Labels would be assigned to complete discussions and a machine-learned model computed that infers discussion-level classifications from shape and paired-shape classifications. One problem with such an approach is to obtain sufficient data: It is questionable whether the 72 discussions we currently have are sufficient. Another difficulty lies in the propagation of errors in a two-stage classification process: Erroneous classifications on the shape and paired-shape level might cause additional noise in the input for the discussion-level classifier and hence, might have a harmful effect on performance.

Given the uncertainties regarding whether it is possible to define or learn a reliable classification model for discussion characteristics, we see potential in another way of bootstrapping from our shape and paired-shape results: Mikšátko and McLaren [17] have developed a graph-matching algorithm, DOCE, that enables teachers to define and find pedagogically interesting *clusters*, i.e. arbitrary large patterns, in on-going discussions. Clusters are defined in terms of examples (e.g. an example for "Deepening discussion with multiple opinions") that can be used to find similar clusters in other discussions. We expect these example patterns will prove more useful to a teacher than shape and paired-shape indicators in isolation, which might be structurally too limited and fine grained to capture significant interactions between students. Initial evidence discussed in [17] shows that the use of shape- and paired-shape indicators as cluster features play an important role in the graph-matching process.

Acknowledgements

This work would not have been possible without Rakheli Hever, Reuma De Groot, Maarten De Laat, Matthias Krauß, and Adam Giemza, as well as other members of the ARGUNAUT project team. This research was sponsored by the 6th Framework Program of the European Community, Proposal/Contract No. 027728.

References

1. Zeini, S., Malzahn, N., Hoppe, U.: Kooperationswerkzeuge im Kontext virtualisierter Arbeit. Virtuelle Organisationen und neue Medien 2004. Gemeinschaften in neuen Medien GeNeMe (2004)
2. McLaren, B.M., Scheuer, O., De Laat, M., Hever, R., De Groot, R., Rosé, C.: Using Machine Learning Techniques to Analyze and Support Mediation of Student E-Discussions. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Proceedings of the 13th International Conference on Artificial Intelligence in Education (AIED 2007), pp. 331–340. IOS Press (2007)

3. Rosé, C., Wang, Y.C., Cui, Y., Arguello, J., Stegmann, K., Weinberger, A., Fischer, F.: Analyzing Collaborative Learning Processes Automatically: Exploiting the Advances of Computational Linguistics in Computer-Supported Collaborative Learning. International Journal of Computer-Supported Collaborative Learning (in press)
4. Dönmez, P., Rosé, C., Stegmann, K., Weinberger, A., Fischer, F.: Supporting CSCL with Automatic Corpus Analysis Technology. In: Koschmann, T., Suthers, D.D., Chan, T.-W. (eds.) Proceedings of the Conference on Computer Supported Collaborative Learning 2005 (CSCL 2005), pp. 125–134. Lawrence Erlbaum (2005)
5. Kumar, R., Rosé, C., Wang, Y.C., Joshi, M., Robinson, A.: Tutorial Dialogue as Adaptive Collaborative Learning Support. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Proceedings of the 13th International Conference on Artificial Intelligence in Education (AIED 2007), pp. 383–390. IOS Press (2007)
6. Goodman, B., Linton, F., Gaimari, R., Hitzeman, J., Ross, H., Zarrella, G.: Using Dialogue Features to Predict Trouble During Collaborative Learning. User Modeling and User-Adapted Interaction 15, 85–134 (2005)
7. De Groot, R., Drachman, R., Hever, R., Schwarz, B., Hoppe, U., Harrer, A., De Laat, M., Wegerif, R., McLaren, B.M., Baurens, B.: Computer Supported Moderation of E-discussions: the ARGUNAUT Approach. In: Chinn, C., Erkens, G., Puntambekar, S. (eds.) Proceedings of the Conference on Computer Supported Collaborative Learning 2007 (CSCL 2007), pp. 165–167 (2007)
8. Hever, R., De Groot, R., de Laat, M., Harrer, A., Hoppe, U., McLaren, B.M., Scheuer, O.: Combining Structural, Process-Oriented and Textual Elements to Generate Awareness Indicators for Graphical E-discussions. In: Chinn, C., Erkens, G., Puntambekar, S. (eds.) Proceedings of the Conference on Computer Supported Collaborative Learning 2007 (CSCL 2007), pp. 286–288 (2007)
9. Japkowicz, N., Stephen, S.: The Class Imbalance Problem - A Systematic Study. Intelligent Data Analysis 6, 429–450 (2002)
10. Weiss, G.M.: Mining with Rarity: A Unifying Framework. In: SIGKDD Explor. Newsletter, vol. 6, pp. 7–19. ACM Press (2004)
11. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: YALE: Rapid Prototyping for Complex Data Mining tasks. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006), pp. 935–940. ACM Press (2006)
12. Cohen, J.: A Coefficient of Agreement for Nominal Scales. Educational and Psychological Measurement 20(1), 37–46 (1960)
13. Landis, J.R., Koch, G.G.: The Measurement of Observer Agreement for Categorical Data. Biometrics 33, 159–174 (1977)
14. Krippendorff, K.: Content Analysis: An Introduction to its Methodology. Sage Publications (1980)
15. Meier, A., Spada, H., Rummel, N.: A Rating Scheme for Assessing the Quality of Computer-Supported Collaboration Processes. International Journal of Computer-Supported Collaborative Learning 2, 63–86 (2007)
16. Baker, M., Andriessen, J., Lund, K., van Amelsvoort, M., Quignard, M.: Rainbow: A Framework for Analyzing Computer-Mediated Pedagogical Debates. International Journal of Computer-Supported Collaborative Learning 2, 315–357 (2007)
17. Mikšátko, J., McLaren, B.M.: What's in a Cluster? Automatically Detecting Interesting Interactions in Student E-Discussions. In: Proceedings of the 9th International Conference on Intelligent Tutoring Systems (ITS 2008). Springer (2008)

What's in a Cluster? Automatically Detecting Interesting Interactions in Student E-Discussions

Jan Miksatko and Bruce M. McLaren

Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI)
Stuhlsatzenhausweg 3
D-66123 Saarbrücken Germany
{honza.miksatko,bmclaren}@dfki.de

Abstract. Students in classrooms are starting to use visual argumentation tools for e-discussions – a form of debate in which contributions are written into graphical shapes and linked to one another according to whether they, for instance, support or oppose one another. In order to moderate several simultaneous e-discussions effectively, teachers must be alerted regarding events of interest. We focused on the identification of *clusters* of contributions representing interaction patterns that are of pedagogical interest (e.g., a student clarifies his or her opinion and then gets feedback from other students). We designed an algorithm that takes an example cluster as input and uses inexact graph matching, text analysis, and machine learning classifiers to search for similar patterns in a given corpus. The method was evaluated on an annotated dataset of real e-discussions and was able to detect almost 80% of the annotated clusters while providing acceptable precision performance.

Keywords: Educational Data mining, Machine Learning in ITS, Collaborative Learning, Natural Language and Discourse.

1 Introduction

One of the important trends in Computer-Supported Collaborative Learning (CSCL) is the development and use of networked visual argumentation tools that allow students to work on separate computers and express their ideas, questions, and arguments in visual fashion. Students make contributions to the online discussion by dragging and dropping shapes with different meanings (e.g. "claim" or "question"), filling them with text containing their contributions to the discussion, and linking the shapes to other relevant shapes with labeled links, such as "opposes" or "supports." An example of such an e-discussion in the Digalo collaboration software is shown in Fig. 1 (text in the shapes shows only the title of the contribution).

Although computer-based tools for collaboration, argumentation, and discussion are becoming relatively commonplace in schools [1,2], there is a critical need for software that can help teachers observe, guide, and moderate such e-discussions. For instance, suppose a classroom of students, organized in small discussion groups of 4 to 6 students, is tasked with discussing and debating a social sciences topic such as "Is

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 333–342, 2008.

it ethical to perform experiments on animals?" using a visual collaboration tool. The teacher in such a classroom obviously cannot monitor and moderate all of these discussions simultaneously without some automated system support. Furthermore, past research suggests that discussion and collaboration tools used by students on their own with no support does not typically lead to fruitful collaboration [3].

In our work on the ARGUNAUT project [4], the Moderator's Interface (MI) – a software tool that displays multiple simultaneous e-discussions taking place in the classroom – provides the teacher with such support by pointing her to events requiring human intervention.

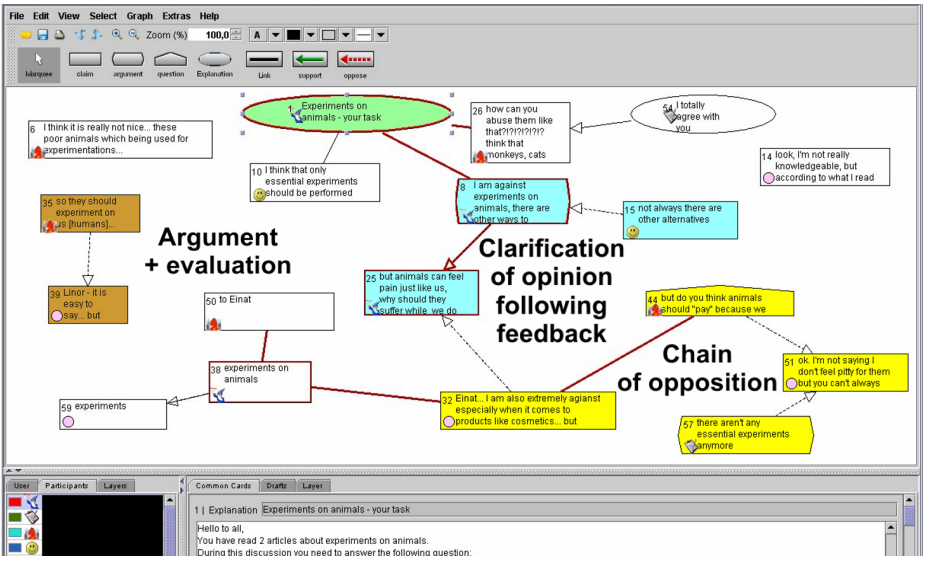


Fig. 1. A well-structured discussion in Digalo software with three simple clusters [1]

In the present work, we are trying to address the problem of identifying complex interaction patterns in the e-discussions. Such patterns, called *clusters* in the remainder of the paper, are multiple contributions, typically (but not exclusively) made by different students, that capture interesting interactions in the e-discussion. Fig. 1 shows an example of a few such clusters. For instance, "Clarification of opinion following feedback" involves a student clarifying his or her opinion and then getting feedback from other students. Types of clusters representing interesting interactions are specified and annotated by the pedagogical researchers on the ARGUNAUT project, with an eye toward moderating e-discussions. Our primary aim is to provide teachers, the users of the MI, with a tool that can point them to interesting conversational moves and clusters in the discussions [5]. A secondary goal is to support the pedagogical researchers in searching off-line for interesting patterns, as they evaluate and data mine past discussions.

[1] The names of students in this discussion have been anonymized to protect their identities.

Our task is a daunting one because

(1) we are dealing with highly complex data (i.e., a combination of graph structure and text),
(2) discussion "maps" (as they will be called henceforth) typically have quite a bit of noisy data,
(3) cluster types are difficult to precisely specify, and
(4) we have a limited source of annotated data, since annotating clusters in real discussions is extremely time-consuming and difficult.

We explored several approaches but ultimately designed and developed one approach that seemed to best fit the problem characteristics above: ***DOCE*** (***D***etection ***o***f ***C***lusters by ***E***xample). DOCE is based on the idea of using cluster examples to find similar clusters in other discussions and has demonstrated very promising preliminary results on an initial set of annotated maps. The main advantages of the DOCE algorithm can be summarized as follows:

- The algorithm does not require precisely defined clusters; instead, it employs an intuitive approach in which cluster examples are provided.
- Only a few annotations are required, as examples for queries, contrary to the large number of examples required by supervised methods. Furthermore, it provides a tool for collecting the annotations.
- It can detect clusters based on their structural and content features, important to the goals of the ARGUNAUT project.
- It is noise tolerant, as it looks for similar, not exactly the same, clusters.

In this paper we describe the DOCE algorithm and present our initial, encouraging results.

2 Related Work

Analyzing student contributions and assigning labels is common practice in designing and experimenting with intelligent educational technology. For instance, the researchers in [6] investigated machine-learning approaches by training classifiers on the *language* of a large corpus of labeled data and classifying single contributions into categories. These results led to the development of TagHelper – a tool for text classification that is also utilized in our work.

In addition to the text classification capability of TagHelper, our work with DOCE also incorporates the *structure* of the discussion by using machine-learned classifications of single contributions (e.g. Topic Focus, Reasoned Claim) and paired contributions (e.g. Contribution-Supporting Argument) [5,7]. Contributions are characterized by a combination of text features extracted by TagHelper and structural attributes relevant to the e-discussions, such as shape type and number of in- and out- links. Several highly reliable classifiers (with Kappa >0.6) have been trained and integrated in the Moderator's Interface as "Awareness Indicators", as discussed in [5,7].

However, such *supervised* learning approaches do not scale well to clusters of arbitrary size. Clusters not only need to be classified, as in standard machine learning approaches, but also *recognized* in the discussion. In addition, obtaining a sufficient

number of annotations for training at the cluster level is a very time consuming and difficult task – much more difficult than annotating single and paired contributions.

A related unsupervised method, detection of frequently reoccurring patterns, was applied in [8] for identifying common interaction patterns during student software development projects on data from source repository logs or Wiki pages. However, the clusters defined by our expert annotaters do not necessarily occur as frequently in our domain as in theirs. A similar approach was evaluated on the ARGUNAUT project in [9]. A tool was designed for mining sequences of actions in the discussions, such as "create shape"/"add link"/"modify text." The tool was able to detect some commonly occurring patterns. However, their exact-matching algorithm was unable to detect *all* of the patterns of given cluster types, especially when clusters differed in subtle and imprecise ways from one another. Our goal was to address these issues with DOCE.

3 Detection of Clusters by Example (DOCE)

The DOCE algorithm is based on the Query By Example (QBE) technique that has been applied to databases as a query method. The idea is to search for similar files or documents based on an input example: a text string, a document, or visual table example [10]. The AI subfield of case-based reasoning [11] is another research area in which examples (i.e., cases) are used to search for similar instances in a repository of data (i.e., a "case base"). A teacher or researcher selects a cluster (e.g. connected individual contributions) in an existing discussion that exemplifies an interesting pattern. The example cluster (also called a "model graph" in the following text) is then used as a search query for similar clusters across other discussion maps (called "input graphs"). The output of the algorithm is a list of matching clusters in the discussion map(s), sorted according to a similarity rating. DOCE can be used as a "live" classifier of clusters – characteristic example(s) representing a cluster of a particular type are stored in the database and used later as queries for automated cluster detection. Or, it can be used as a research tool for obtaining clusters and annotating them in the first place.

The DOCE algorithm is sketched in Fig. 2. First, the example cluster and the discussion map are parsed from an XML file format that is used by the Moderator's Interface for representing a snapshot of the discussion. Both graphs are preprocessed as follows: (1) an adjacency matrix representing the structure of the graph is constructed; (2) each contribution and link in the discussion graph is characterized by a feature vector that is extracted from the attributes associated with the discussion vertex and edge such as shape/link type, text length, link direction and whether the same user created two linked shapes. TagHelper [6] further enriches the feature vectors with additional information from the text analysis of contributions. It performs text processing (e.g. stemming) and extracts textual attributes such as *unigrams* and *bigrams* (single words and pairs of words occurring in the text), *punctuation* (indicator of question or mood of the author) and *contains non-stop words* (a value predicting if the text is meaningful or not). Additionally, we extend the feature vectors of shapes (links) with the high-accuracy output of shape (pair-shape) classifiers that assign contributions (pairs of contributions, respectively) into categories [5,7]. In the next step, DOCE compares the feature vectors of vertices/edges in the model and input graphs by calculating their distance in a manner similar to

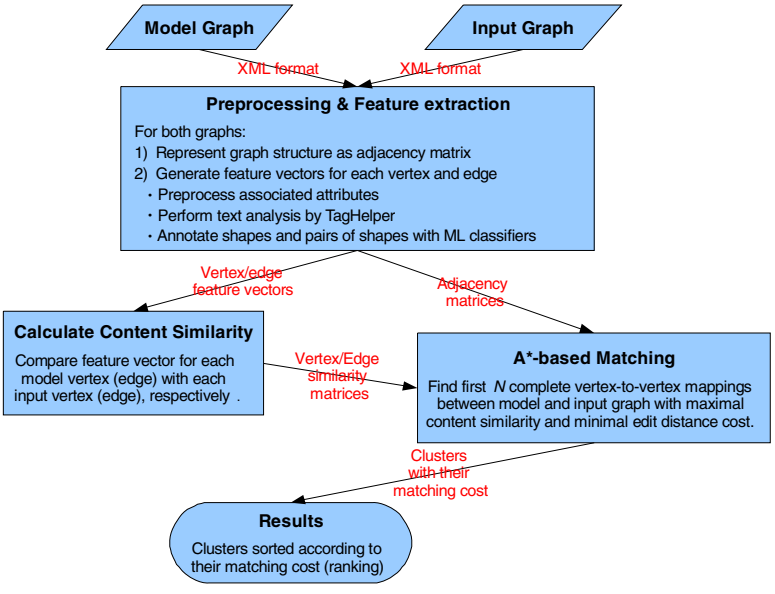


Fig. 2. The DOCE algorithm

unsupervised learning algorithms. The proximity is pre-computed for each pair of model/input objects and stored in the similarity matrices.

Finally, an inexact graph matching method based on a customized version of the edit distance algorithm [12,13] is employed to find clusters with the highest structural and content similarity to the model graph. Similar algorithms have been used for various purposes, such as computer vision [16], pattern recognition [14], and retrieving relevant principles from ethics cases [15]. For instance, in [15] engineering ethics cases and principles were represented in a stylized, graphical language. An undecided case was then matched against past cases and a human was provided with suggestions in deciding the current case.

The matching works as follows. An $A*$ search algorithm explores all possible vertex-to-vertex mappings between the model and input graph. In each step, a partial mapping of vertices is extended by adding a new vertex-to-vertex assignment that has the maximum content similarity (pre-computed in the similarity matrices) and the minimum structural difference, as measured by edit distance. The edit distance between partially matched graphs is calculated as a minimal sequence of primitive graph operations (such as "add an edge", "delete an edge", "delete a vertex") that are required in order to make the graphs isomorphic. The final matching cost is the sum of all vertex/edge similarities and penalties for the edit operations. The first n complete mappings (i.e. mappings that cover all model vertices) are returned as resulting clusters and sorted in ascending matching-cost order.

Thus, the algorithm matches *similar* clusters on *generic* graph structures in an inexact manner (e.g., some of our cluster examples are unconnected as well as shapes in the discussion may be unlinked). The matching is driven by both the graph structure

and *content* of contributions, for example, the text of the contribution, the users involved in the cluster, and shape type. Note that the detection of all subgraphs is an NP-Complete problem but only in theoretical, not practical, terms. The search space is significantly reduced by applying heuristics similar to [13,16] and the method performs well on graphs of moderate size (dozens of vertices). The graphs in our particular domain are certainly within this range.

The DOCE algorithm is described in further detail in [17].

4 Evaluation

We designed an evaluation methodology in which the pedagogical specialists analyzed 27 discussion maps and annotated cluster examples (referred to as "annotations" henceforth) for the three most important types of clusters (as suggested by pedagogical specialists): *Clarification of opinion following feedback, Chain of opposition, Argument + evaluation*. There were a total of 74 annotations. We used the annotations in each map as input to the DOCE algorithm to evaluate how well the algorithm could find the cluster examples in the *other* 26 annotated maps. The clusters detected by DOCE were then compared to the annotations in the maps. We compared the performance of DOCE using different feature sets of the algorithm and also compared it to a random algorithm, as there is no other comparable "gold standard" algorithm, at least not for the particular type of problem we are tackling in this work.

Our methodology is similar to Information Retrieval (IR) evaluations – the "Top10" results are considered in the evaluation and the relevancy of results is defined based on user feedback [18]. As already explained, DOCE does not always match clusters in an exact manner. Thus a matching cluster was considered "relevant" if the overlap of vertices between the matching cluster and an annotation is at least 70 % (rounded) of the annotation size (e.g. if an annotated cluster is {1,2,3,4}, then a "matched" cluster {2,3,4,5} is relevant). The pedagogical experts verified the acceptability of this definition of relevance, which is based on the idea that even a non-exact match can be valuable since the ultimate objective of DOCE is to draw a teacher's attention to interesting behavior in a discussion map, not perfectly match that behavior.

We used several metrics in our evaluation:

- *Recall* represents the number of relevant matches in the Top10 divided by the count of annotations in the searched map.
- *Precision* is the number of relevant matches in the Top10 divided by 10.
- *Ranking Quality*, known as *Average Precision* in IR, measures the quality of the *ordering* of the results. A higher value means better ordering of the matching clusters, with the best value being 1.0 (all matches are on at the top of the list).
- *Stability* is used to evaluate the consistency of the DOCE algorithm with different input models of the same cluster type against the same map. It is calculated as the average intersection size (ranging from 0 to 10) of all pair wise result sets.

We consider *Recall* to be the most important metric, as it is highly critical to find *all* of the interesting clusters in a given discussion. We believe the number of relevant matches (i.e. *Precision*) has somewhat lower importance since humans are typically clever enough to filter out irrelevant matches.

5 Results

Our overall results, averaged across all models and maps, are presented in Table 1 along with a comparison of different configurations of the algorithm and the Random Matcher. The DOCE *Baseline* feature set includes only attributes directly available from the structure of the discussion map (e.g. shape type, link direction, users involved in the cluster), with no text considered. The *Text* configuration includes annotations from the shape/pair-shape level analysis [5,7] and attributes obtained from the TagHelper linguistic analysis [6]. The weights are set to prefer the text attributes. The *Text* configuration was experimentally chosen as the best combination of features and attribute weights. The parameter π (ranging from 0 to 100) influences the ordering and balance of the content similarity and edit operations – high (low) values prefer matches with few (many) edit operations at the top positions in the result list, respectively. We present the results with a "neutral" π value (π=50) in order to avoid bias from parameter choice, and the manually tuned value (π=100).

Table 1. Overall results and comparison of DOCE algorithm to the Random Matcher

Configuration	Recall	Precision	Ranking Quality	Stability
Random Matcher	21,3%	6,6%	0,32	0,5
DOCE (Base, π=50)	62,7%	28,6%	0,51	4,3
DOCE (Text, π=50)	73,0%	35,8%	0,57	5,3
DOCE (Text, π=100)	79,0%	37,3%	0,57	6,2

The DOCE algorithm performs significantly better than the Random Matcher across all measures and configurations as confirmed by t-tests ($p<0.000001$ in all cases). DOCE can match more than ***60%*** more annotations than the random method. Furthermore, from the overall results the following conclusions can be drawn:

- The algorithm was able to detect almost 80% of cluster examples annotated by pedagogical experts (for the best configuration).
- The *Precision* result can be interpreted as meaning that only every third matching cluster is relevant. While this value is low, it is worth noting that the input maps contain 3.3 annotations on average; thus, fixing DOCE to always return the top 10, as we have done, will always produce relatively low Precision values.
- The *Stability* of the DOCE algorithm with respect to different models is relatively high. On average 6 clusters (for the best configuration) are in common when comparing two results sets produced by two different models against the same map, despite the fact that the models are often from discussions with different topics.
- A more fine-grained analysis of the results showed that, on average, more than 60% of relevant clusters are exact matches (in comparison to 11% for the random matcher).

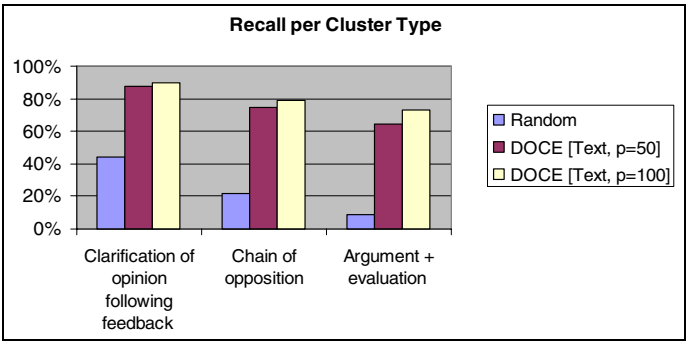


Fig. 3. Recall per cluster type

Finally, Fig. 3 shows the number of detected annotations in each map per cluster type. As can be seen, the algorithm delivered a relatively high *Recall* value for all three cluster types, significantly better than random, regardless of the π value.

6 Discussion and Further Work

Although the initial results are quite promising, the algorithm was evaluated on a relatively small dataset. We had hoped the pedagogical experts on our project would be able to provide a much larger set of annotated maps, say hundreds, that could also be evaluated for inter-rater reliability. In fact, a single annotator provided all of the 74 annotations used in our evaluation. Despite this shortcoming, we argue that the dataset and annotations are sound and the evaluation meaningful because DOCE detects clusters that are similar to the provided models – in other words, the algorithm adapts to the "style" of the annotator. In practical terms, it seems unlikely that we will obtain a high level of inter-rater reliability for such an arduous and inexact task as identifying "meaningful" clusters, at least not without detailed specifications and extensive training of coders. On the other hand, note that many annotations were marked as *borderline* examples and could have negatively influenced the results of our experiments, yet we kept and used *all* of the annotations.

Currently, the pedagogical specialists are annotating additional maps and cluster types, and we plan further evaluation of the algorithm on a larger dataset. Additionally, we are working on integrating the algorithm into the Moderator's Interface in order to provide researchers with a tool for searching for more annotations. Another planned step is to experiment with using a set of models against one discussion map and then merging the results. Such an approach might improve DOCE's search accuracy. We will also tap our pedagogical experts' knowledge and perform experiments to customize the parameters of the algorithm.

Our long-term goal is to obtain enough annotations to better understand the cluster types and develop an extended approach that can leverage domain knowledge. For example, an ML classifier may be used for filtering results produced by the DOCE algorithm.

7 Conclusion

Students in different classrooms around the world are using visual argumentation tools for e-discussions. In order to effectively moderate multiple, simultaneous discussions, a tool providing feedback to the teacher is required. The ARGUNAUT system is designed to help a teacher monitor the progress of multiple conversations through "Awareness Indicators" that display interesting events in the discussion.

In this work, we focused on analysis of segments of the discussion maps representing interaction patterns that are of pedagogical interest. Detection of such "clusters" of contributions is a complex task because the graph and text structure must be accounted for, the cluster types are imprecisely defined, and annotations are scarce.

We designed the DOCE algorithm to accept an example cluster and find similar clusters across different discussion maps. The method is an extension of the edit distance inexact graph matching algorithm and looks for subgraphs in the discussion maps that have the highest content similarity and lowest structural difference from an input model. The content similarity function accounts for discussion attributes, the text analysis performed by the TagHelper tool, and machine-learned classifications from the shape/pair-shape level.

We evaluated the algorithm on 27 actual discussion maps with 74 of the three most important clusters annotated by pedagogical experts. DOCE was able to detect almost 80% of the annotated clusters. We used all models in our evaluation, including ones from discussions with different topics and ones that were characterized as "borderline" examples. Furthermore, we compared the results with a random matcher, as there was no other "gold standard" algorithm available, and DOCE significantly outperformed this approach.

In sum, the experiments, although preliminary and on a limited dataset, have shown very promising results. However, deeper investigation and more extensive evaluation are planned. We intend to analyze the algorithm on a larger dataset and with more complicated clusters. We will also integrate the DOCE algorithm into the Moderator's Interface so it can help pedagogical experts define more annotations.

Acknowledgments. This work would not have been possible without project partners Rakheli Hever, Reuma De Groot, Maarten De Laat, Matthias Krauß, and Adam Giemza, as well as other members of the ARGUNAUT project team. The 6th Framework Program of the European Community, Proposal/Contract No. 027728, sponsored this research.

References

1. Lingnau, A., Harrer, A., Kuhn, M., Hoppe, H.U.: Empowering teachers to evolve media enriched classroom scenarios. Research and Practice in Technology Enhanced Learning 2(2), 105–129 (2007)
2. Schwarz, B., De Groot, R.: Argumentation in a changing world. International Journal of Computer-Supported Collaborative Learning 2, 297–313 (2007)
3. Dillenbourg, P., Baker, H.P.M., Blaye, A., O'Malley, C.: The evolution of research on collaborative learning. In: Learning in humans and machines: Towards an interdisciplinary learning science, pp. 189–211. Elsevier/Pergamon, Oxford (1995)

4. De Groot, R., Drachman, R., Hever, R., Schwarz, B.B., Hoppe, U., Harrer, A., De Laat, M., Wegerif, R., McLaren, B.M., Baurens, B.: Computer Supported Moderation of E-Discussions: the ARGUNAUT Approach. In: Proceedings of the Conference on Computer Supported Collaborative Learning (CSCL 2007), vol. 8, pp. 165–167 (2007)
5. McLaren, B.M., Scheuer, O., De Laat, M., Hever, R., De Groot, R., Rose, C.P.: Using Machine Learning Techniques to Analyze and Support Mediation of Student E-Discussions. In: Frontiers in Artificial Intelligence and Applications, pp. 331–338. IOS Press, Netherlands (2007)
6. Rosé, C., Wang, Y.C., Cui, Y., Arguello, J., Stegmann, K., Weinberger, A., Fischer, F.: Analyzing Collaborative Learning Processes Automatically: Exploiting the Advances of Computational Linguistics in Computer-Supported Collaborative Learning. International Journal of Computer-Supported Collaborative Learning (in press)
7. Scheuer, O., McLaren, B.M.: Helping Teachers Handle the Flood of Data in Online Student Discussions. In: The 9th International Conference on Intelligent Tutoring Systems to take place in Montreal, Canada, June 23-27, 2008 (submitted 2008)
8. Kay, J., Maisonneuve, N., Yacef, K., Zaïane, O.: Mining patterns of events in students' teamwork data. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 45–52. Springer, Heidelberg (2006)
9. Harrer, A., Hever, R., Ziebarth, S.: Empowering Researchers to Detect Interaction Patterns in e-Collaboration. Frontiers in Artificial Intelligence and Applications 158, 503–510 (2007)
10. Zloof, M.M.: Query-by-Example: A Data Base Language. IBM Systems Journal 16, 324–343 (1977)
11. Kolodner, J.: Case-based Reasoning. Morgan Kaufmann Publishers, San Francisco (1993)
12. Wong, A.K.C., You, M., Chan, S.C.: An algorithm for graph optimal monomorphism. IEEE Transactions on Systems, Man and Cybernetics 20, 628–638 (1990)
13. Tsai, W.H., Fu, K.S.: Error-correcting isomorphisms of attributed relational graphs for pattern recognition. IEEE Transactions on Systems, Man, and Cybernetics 9, 757–768 (1979)
14. Messmer, B.T., Bunke, H.: Automatic Learning and Recognition of Graphical Symbols in Engineering Drawings. In: Selected Papers from the First International Workshop on Graphics Recognition, Methods and Applications, vol. 1072, pp. 123–134 (1996)
15. McLaren, B.M.: Extensionally defining principles and cases in ethics: An AI model. Artificial Intelligence 150, 145–181 (2003)
16. Gregory, L., Kittler, J.: Using Graph Search Techniques for Contextual Colour Retrieval. In: Caelli, T.M., Amin, A., Duin, R.P.W., Kamel, M.S., de Ridder, D. (eds.) SPR 2002 and SSPR 2002. LNCS, vol. 2396. Springer, Heidelberg (2002)
17. Miksatko, J.: Using Machine Learning Techniques to Analyze and Recognize Complex Patterns of Student E-Discussions (M.Sc. Thesis). Charles University, Prague (2007)
18. Borodin, A., Roberts, G.O., Rosenthal, J.S., Tsaparas, P.: Link analysis ranking: algorithms, theory, and experiments. ACM Transactions on Internet Technology 5, 231–297 (2005)

Scaffolding On-Line Discussions with Past Discussions: An Analysis and Pilot Study of PedaBot

Jihie Kim, Erin Shaw, Sujith Ravi, Erin Tavano,
Aniwat Arromratana, and Pankaj Sarda

Information Sciences Institute/ University of Southern California,
4676 Admiralty Way, Marina del Rey, CA 90292, USA
{jihie,shaw,sravi,erinth,arromrat,psarda}@isi.edu

Abstract. PedaBot is a new discussion scaffolding application designed to aid student knowledge acquisition, promote reflection about course topics and encourage student participation in discussions. It dynamically processes student discussions and presents related discussions from a knowledge base of past discussions. This paper describes the system and presents a comparative analysis of the information retrieval techniques used to respond to free-form student discussions, a combination of topic profiling, term frequency-inverse document frequency, and latent semantic analysis. Responses are presented as annotated links that students can follow and rate. We report a pilot study of PedaBot based on student viewings, student ratings, and a small survey. Initial results indicate that there is a high level of student interest in the feature and that its responses are moderately relevant to student discussions.

Keywords: threaded discussion, discussion scaffolding, information retrieval, on-line learning environment.

1 Introduction

On-line discussion boards have been found to be an effective medium for collaborative problem solving and discovery-oriented learning [1;2]. However, some research indicates that existing systems for on-line discussion may not always be fully effective in promoting collaborative problem solving and learning as expected. Student participation may be low or weak, even when students are encouraged to participate [3;4]. And when instructors and teaching assistants are not available to fully guide interactions, or alternatively, when instructors provide too many answers, participation between students may decrease [4;5].

In this paper, we present PedaBot, a novel application for scaffolding student discussions with information from past student discussions, from the same or related courses. The system dynamically processes student messages or message threads, mines a corpus of relevant past discussions using information retrieval techniques, and displays the retrieved information. PedaBot was designed to aid student knowledge acquisition, promote reflection about course topics and encourage student participation in discussions. It *scaffolds* discussions in the sense that it provides different perspectives on the current discussion topic.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 343–352, 2008.

Prior analyses indicate that existing information retrieval techniques alone are not fully effective [6] for processing the noisy and incoherent text that is characteristic of student messages [7]. To assist in modeling these messages, we focus on domain terms that are semi-automatically extracted from textbooks. To organize and retrieve information according to the topics in the course syllabus and facilitate efficient retrieval, we classify messages with a topic profiler that is induced from the textbook. We then apply Latent Semantic Analysis (LSA) [8] and term frequency and inverse document frequency (TF*IDF) techniques [9] for retrieval.

PedaBot was piloted in the Fall of 2007 in a computer science course. It mines text from a corpus comprised of seven semester's of discussions from the same undergraduate course, two semester's of discussions from a related graduate course, and segments of text from related course documents. Preliminary results indicate that PedaBot has begun to meet our requirements for scaffolding on-line student discussions. Although many students were not fully aware of the new feature and its usage frequency was not so high, students rated the feature highly interesting and rated its responses as moderately relevant to their discussions.

2 Student Interaction with PedaBot

PedaBot was integrated into a modified version of phpBB (www.phpbb.com). Students in the undergraduate Operating Systems course that we are currently focusing

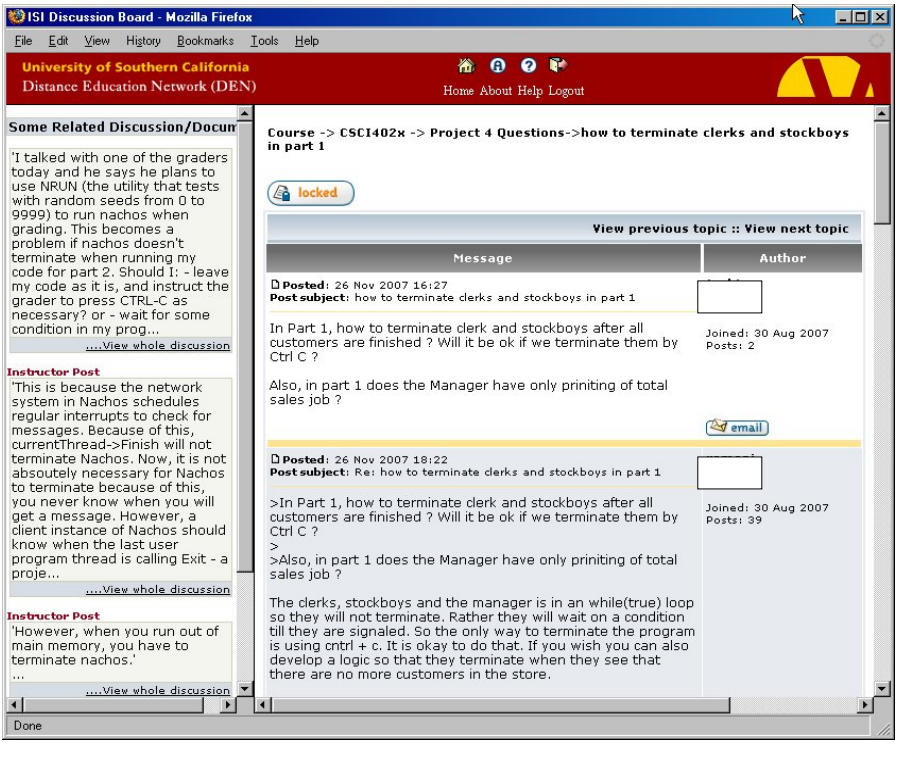


Fig. 1. Relevant messages from past discussions are displayed to left

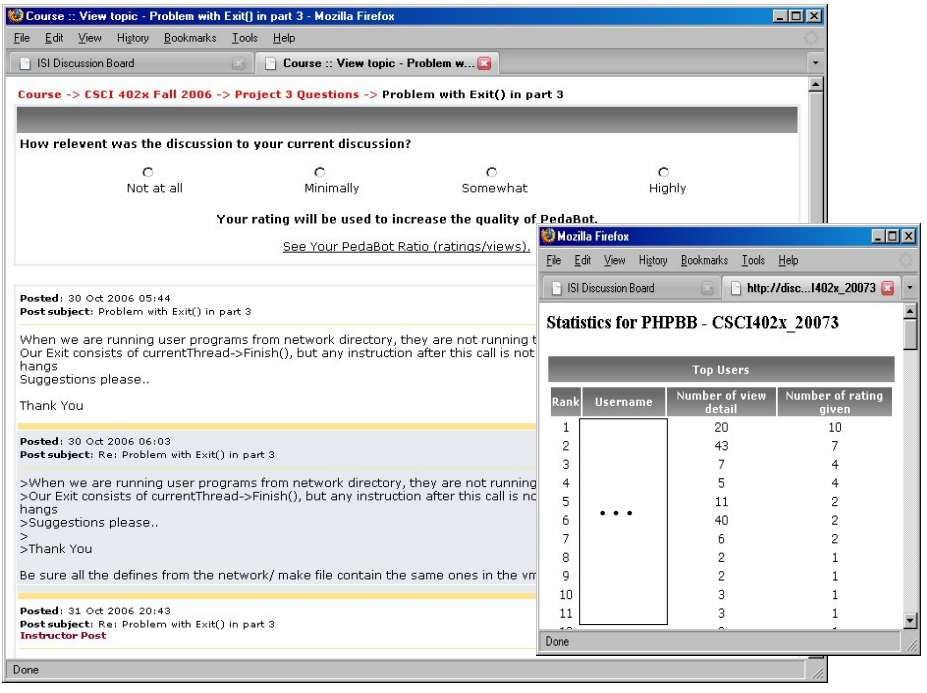


Fig. 2. The "View whole discussion" link displays the relevant message's thread

have used the board for the last eight semesters. It is typically used as a question and answer forum in which the instructor participates frequently.

In Figure 1, there is a screenshot of a discussion thread about terminating a function from the Fall 2007 study. Student names are blocked for privacy. PedaBot filters and submits the first message to the retrieval pipeline when it is posted and responds by displaying portions of the three messages that best match the student's question in the left frame. Messages from the instructor are highlighted as "Instructor Post". The number of responses was limited to avoid overwhelming the student. Although latency is not a critical issue, the results are displayed within 1 or 2 minutes.

The resulting messages are usually part of a longer discussion thread that can be viewed by following the 'View whole discussion' link. The results can also originate from a document; in this case, a portion, or *tile*, of the matching document is displayed and the link is labeled 'View document'. The discussion or document is displayed in a new, instrumented, window, shown in Figure 2, where matching domain terms are highlighted. Students can rate the relevancy of the resulting discussion to the current discussion. They can also view peer rating statistics. The "See your PedaBot Ratio" link displays a table of user ratings. We are adding more 'social' incentives, such as displaying the ratings as stars, to encourage students to browse and rate relevant discussions. The ratings will be used to guide the text retrieval in the future.

3 Mining Relevant Information from Past Student Discussions

This section describes the text retrieval pipeline shown in Figure 3. Section numbers correspond to the numbers in the figure.

Steps 1-3: Modeling Messages with Technical Terms
We use discussion corpora from two Operating Systems courses: an undergraduate level course and a graduate level course. Messages from administrative forums that contain non-technical discussions are excluded. The total number of messages in the current corpus is 6,622. TextTiling [10] is used to divide individual messages and documents into semantically related segments called *tiles*. The system also applies typical document processing steps including stemming and cleaning [7].

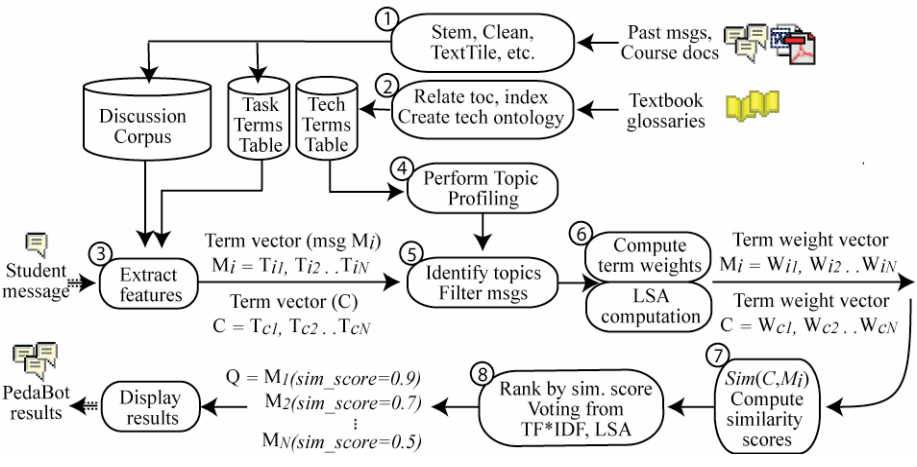


Fig. 3. Steps involved in retrieving relevant messages

Discussion data from undergraduate students is incoherent with respect to grammatical structure, and noisy with respect to individual words, phrases and punctuation. To help us model the messages, we use technical domain terms and terms related to student tasks within the course such as 'assignment' and 'project'.

The technical terms were extracted from the glossaries of the undergraduate and graduate text books, which were automatically scanned and processed, and stored in a 'TECH_TERMS' table in the database. Two text books produced 2,233 technical terms. In addition to the terms extracted from the textbook, we manually selected an additional 1,587 terms that relate to student tasks that were frequently used in discussions. These were stored in the 'TASK_TERMS' table. These correspond to step 1 and 2 in Figure 3. Individual messages are then modeled with a term vector of the following form (Step 3 in Figure 3):

$$M_i =< T_{i1}, T_{i2}, ..., T_{iN} >,$$

where N is the total number of TECH_TERMS and TASK_TERMS in the domain and $T_{ij} = 0$ if a term is missing in that message.

Steps 4-5: Filtering Messages with the Topic Profiler (TP)
The technical terms described above are also used to induce a topic profiler from the textbook. The system maps the main topic categories (chapters and sections) in the "table of contents" to the technical terms. That is, in creating the topic vector for individual sections, the system relates the pages covered by the section and the technical terms that appear in the pages. The system produces a term weight vector for individual topics categories based on TF*IDF (term frequency * inverse document frequency) transformations [9]. Each topic vector for a topic category c takes a form of a term-weight vector,

$$TPc \quad =< \; W_{c1}, W_{c2}, ..., \; W_{cN} \; > \; .$$

Each message is classified by calculating the similarity score between the topic vectors and the message vector described above. The details of the topic vector induction and classification are described in [6]. All of the messages in the corpus are pre-classified with the topic profiler, and tagged with the top three topics.

When a new message is submitted to the retrieval pipeline, the system classifies the message with the topic profiler. The messages that do not share the topics with the new message are removed from the candidate set.

Step 6: Computing Term Weights with TF*IDF and LSA
Term weights are used in calculating similarity scores between messages, such as similarity of a new message and a message in a past corpus. In computing term weights, we use TF*IDF, which is one of the most common ways to model term weights [9]. Messages with same technical terms are more likely to be semantically related. TF (term frequency) weights the commonly occurring terms more and give low weights to rare technical terms. IDF (inverse document frequency) introduces the general importance of the term to the weights. The term vector for each message is converted into a corresponding term-weight vector. Individual TF*IDF weight values for each term are computed as

$$W_{ik} = TF_{ik} * \log(N / n_k).$$

LSA (Latent Semantic Analysis) transforms the occurrence matrix into a relation between the terms and some *concepts*, and a relation between those concepts and the messages. Thus the terms and messages/documents are now indirectly related through the concepts [8]. LSA has been used in various educational applications including dialogue support for intelligent tutoring systems and essay grading.

The system first constructs a term frequency matrix from the term vector model constructed above (in the Modeling Messages section). A statistical technique called Singular Value Decomposition (SVD) is then applied to represent the messages and terms as vectors in a high dimensional abstract space [8]. We explored several different dimensions and used two separate dimension settings, (k=300 and k=75), which are commonly used in LSA applications.

Steps 7-8: Combining TP, TF*IDF, and LSA to Present Relevant Messages
We use cosine similarity to determine the relevance between the new message *C* and a message *Di* in the corpus using

$$sim(C, D_i) = \frac{\sum_{j=1}^{t} w_{q_j} w_{d_{ij}}}{\sqrt{\sum_{j=1}^{t} (w_{q_j})^2 \sum_{j=1}^{t} (w_{d_{ij}})^2}}$$

This measures the cosine of the angle between the two term-weight vectors representing the two messages. Messages are ranked by their scores in order to find the most relevant messages.

Similarly, for LSA, we calculate the cosine similarity of the vector with each document's "concept" vector that was generated by LSA (Step 7 in Figure 3). Once all the similarity scores are computed, the messages are ranked by the similarity scores.

We combine the above results from TF*IDF and LSA options (k=75 and k=300) based on average rankings of the retrieved messages. We take the top three results from each set and combine the rankings.

4 Preliminary Analysis of PedaBot Responses

Since a previous analysis with a smaller set of data showed that LSA alone did not produce better results, we explored three alternatives: TF*IDF only, TF*IDF with the topic profiler and a combination of TF*IDF and LSA with the topic profiler. Our analysis focused on the *degree of relevance* to a given student message. For the analysis, we created a new message corpus with the discussion data from Fall 2006. We extracted the first messages from all threads in the new corpus and then randomly selected 30 of the messages and processed them, using the steps described above, against a larger discussion corpus. Two human evaluators rated the retrieved messages using a Likert scale from 1 to 4 corresponding to 'not relevant', 'low relevance', 'medium relevance' and 'high relevance', respectively. The agreement ratio between the evaluators for the best messages was 76.7%.

Table 1. Average relevancy ratings and MRR for retrieved messages

	TF*IDF	TF*IDF + Topic Profiler	TF*IDF + LSA (k=75) + LSA (k=300) + Topic Profiler
With 6,622 messages from 9 courses			
Rank 1 msg.	0.59	0.60	0.37
Rank 2 msg.	0.51	0.51	0.49
Rank 3 msg.	0.54	0.55	0.43
Overall Avg. Rating	0.55	0.55	0.43
MRR for best message	0.65	0.64	0.57

Table 1 shows the average relevancy of top three responses retrieved by PedaBot. We assigned numbers between 0 and 1 (0, 1/3, 2/3, 1) for each of the four degrees of relevance. The preliminary results indicate that the tool can retrieve moderately relevant information (close to "medium relevance") from past discussions. Among the three combinations, TF*IDF with topic profiler provide slightly better results than TF*IDF for average relevance. Current LSA options did not effectively exploit concept relevancy,

especially among domain terms. The high degree of variance in student discussion data may have also affected the result. Table 1 also shows MRR (mean reciprocal rank) for the best message selected by the evaluators. The current results show that TF*IDF rates the best message slightly higher than other combinations. Based on these results, we are currently using TF*IDF in retrieving relevant messages.

$$MRR_{relevance} = \frac{1}{N\_of\_new\_msgs} * \sum \frac{1}{best\_msg\_position} -$$

5 Pilot User Study, Fall 2007

PedaBot was integrated into a live student discussion board during the Fall of 2007, for an upper-level undergraduate Operating Systems course offered by the Computer Science department at the University of Southern California As indicated in Table 2, student gender is predominantly male. The Pedabot feature was introduced during the fourth week into the 15-week semester. Initially, results were shown for only every other topic/thread, as part of a study was to determine their impact on discussion. But usage of the feature was low – students did not appear to notice the results – so the every-other-topic constraint was removed to familiarize more students with the new feature. PedaBot was active for 127 threads out of 301 total discussion threads. (Lecture and project threads only; administrative and humor forums were excluded.)

Usage Results

The usage results are shown in Table 2. The numbers are situated in the PedaBot study context, that is, the forums in which and the time periods during which PedaBot

Table 2. Discussion board and PedaBot usage

Number of students who…	Male	Female	Combined
Registered on discussion board	104	15	119
Participated in discussions (discussants)	82	9	91
Initiated discussion threads	70	8	78
Viewed entire discussion context of PedaBot retrieved messages	55 (55/82 = 67%)	7 (7/9 = 78%)	62
Rated PedaBot retrieved messages	15	0	15
Average number of …	Male	Female	Combined
Messages posted (#messages / #discussants)	582/82 = 7.09	18/9 = 2.0	600/91 = 6.59
Threads participated in (#threads / #discussants)	293/82 = 3.57	18/9 = 2.0	311/91 = 3.42
Average number of Pedabot ...	Male	Female	Combined
Discussion details viewed (#viewings / #viewers)	348/55 = 6.33	23/7 = 3.29	371/62 = 5.98
Results rated (#ratings / #raters)	39/15 = 2.6	0	39/15 = 2.6

was active. Female student discussion participation is lower than male student participation. However, among the discussion participants, more female students viewed the discussion thread details retrieved by PedaBot (78% vs. 67%).

Table 2 indicates that the frequency of female student usage is lower in terms of the average number of thread viewings and the average number of ratings provided. The seven female students who used PedaBot viewed the thread details for only about three times. We plan to investigate strategies for promoting more frequent use of PedaBot for the female students who access PedaBot. The average rating (2.46) on a scale of 4-1 (highly, somewhat, minimally, or not at all) is consistent with our prior analysis that PedaBot retrieves moderately relevant messages.

We hoped that having previous results appear would encourage discussion and increase the length of the discussion. In Table 3, we analyze PedaBot's effect on the average number of messages in a thread. The number of messages per thread seems a little higher with PedaBot, especially for female students.

Survey Results

A plain-text questionnaire was emailed to students at the end of the semester. Only nine students responded, one of whom did not use the feature at all. Students who used the feature were asked to rate the results on a scale of 4-1 (highly, somewhat,

Table 3. Difference in thread length with and without PedaBot

Fall 2007		**with** PedaBot	**without** PedaBot
Average number of messages per thread	Male	426/124= 3.43	533/169 = 3.15
	Female	65/12 = 5.41	12/6 = 2.0
	Combined	431/127 = 3.39	543/174 = 3.12

Table 4. Survey answers

Survey question (n=7)	Avg. rating
How interesting the feature was for you?	3.42/4
How useful the feature was for you?	2.83/4
How relevant the resulting discussion or document was to your discussion?	3.14/4
Student Comments: Answers to 'What did you like and/or dislike about the new feature?"	
I didn't always use the feature. But I found it to be useful when I did use it.	
Good way to find all the relevant questions at one place. Results of past discussions were useful for my project work.	
All-around, it was a great feature. Would like to see more relevance and the bot actually checking for phrases instead of just specific terms.	
It is very hard to follow most of the 'Whole Discussions' because none of the names are attached to the messages. At first you couldn't even tell when it was the instructor posting (I know this was fixed during the semester) but I think it is still important.	
I liked that the system tried to provide me with relevant posts (even though most of the results were not relevant). This new feature saved me the trouble of typing in a search keyword to explicitly fetch related topics or posts.	

minimally, or not at all) with respect to their interest, usefulness and relevancy to the student. All students rated their interest in the feature positively and most also found the feature relevant. Students found the feature moderately useful. Student comments in Table 4 support the high interest ratings.

6 Related Work

In the area of on-line learning, much attention has been paid to the analysis of student learning behaviors in on-line communications. Various frameworks have been proposed for characterizing and analyzing computer mediated communication in the context of collaborative discussions [11], knowledge sharing [12] and general argumentation or dialogue [13;5]. In particular, the work in [14] classifies student on-line discussions for collaborative learning support. We provide a complementary approach for facilitating scaffolding in threaded discussions by extracting useful information from past discussions.

There are many approaches to assessing and extracting knowledge from collaborative activities generally and from computer supported collaborative argumentation specifically [15]. Machine learning techniques have been applied to train software to recognize when participants have trouble sharing knowledge in collaborative interactions [12]. Analyzing and utilizing discourse information at the paragraph level still remains a challenge to the natural language community.

Developing automatic natural language-based question answering systems has been a field of research for many years [16], but question-answering in an open domain is still an unsolved problem. Most existing question-answering systems assume that queries are concise, coherent and of a particular form (e.g., a factoid question), and can be answered with short phrases. Queries posted to a discussion board often span multiple sentences, are incoherent and include extra (informal) content and lengthy descriptions, especially in technical discussions. Our current work focuses on finding 'relevant or interesting' messages for student discussions rather than retrieving answers.

7 Summary and Discussion

We have presented a new type of "tutor" that helps the participants of on-line course discussions acquire knowledge in a compelling new way: by sending related discussions. The results of a Fall 2007 pilot study indicate that PedaBot were encouraging. Although students showed interest in PedaBot, some students were unaware of the feature and the usage frequency was not very high. We plan to ensure that students are informed of the feature.

In the immediate future, we will focus on improving the relevancy of the results, encouraging students to view the results and promoting further discussion based on the results. For example, we may allow students to share their ratings for retrieved message or to collaboratively annotate retrieved course documents.

Further out, we wish to improve the usefulness of PedaBot by incorporating different tutorial scaffolding strategies such as generating questions and comments or inviting participation and clarification. To this end are investigating ways to characterize student messages and discussion threads. For example, speech act classifiers can

identify whether a message contains questions or answers [7]. This type of information can be useful for selecting the information to send the students (e.g. answers instead of additional questions). An analysis of student participation over different course forums and domain topics could be used to pro-actively invite students with a known knowledge or interest to participate in a particular discussion.

Acknowledgements

This work is supported by the NSF, CCLI-Phase 2 grant (Award No. 0618859).

References

1. Scardamalia, M., Bereiter, C.: Computer support for knowledge building communities. In: Koschmann, T. (ed.) CSCL: Theory and practice of an emerging paradigm. Erlbaum, Mahwah (1996)
2. Koschmann, T. (ed.): CSCL: Theory and practice of an emerging paradigm. Lawrence Erlbaum, Hillsdale (1996)
3. Pallof, R.M., Pratt, K.: Building Learning Communities in Cyberspace: Effective Strategies for the Online Classroom (1999)
4. Kim, J., Beal, C.: Turning quantity into quality: Supporting automatic assessment of online discussion contributions. In: AERA 2006 (2006)
5. Painter, C., Coffin, C., Hewings, A.: Impacts of Directed Tutorial Activities in Computer Conferencing: A Case Study. Distance Education 24(2) (2003)
6. Feng, D., Kim, J., Shaw, E., Hovy, E.: Towards Modeling Threaded Discussions through Ontology-based Analysis. In: Proceedings of National Conference on Artificial Intelligence (AAAI 2006) (2006)
7. Ravi, S., Kim, J.: Profiling Student Interactions in Threaded Discussions with Speech Act Classifiers. In: Proceedings of AI in Education (2007)
8. Landauer, T., Dumais, S.: A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. Psychological Review 104(2) (1997)
9. Salton, G.: Automatic Text Processing, The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley, Reading (1989)
10. Hearst, M.A.: Multi-paragraph segmentation of expository text. In: Proc. of the 32th Annual Meeting of the Association for Computational Linguistics, Las Cruces, pp. 9–16 (1994)
11. Shaw, E.: Assessing and Scaffolding Collaborative Learning in Online Discussions. In: Proceedings of AI in Education (2005)
12. Soller, A., Lesgold, A.: Computational Approach to Analyzing Online Knowledge Sharing Interaction. In: Proceedings of AI in Education (2003)
13. Graesser, A.C., Person, N., Harter, D.: TRG: Teaching tactics and dialog in AutoTutor. International Journal of AI in Education 12 (2001)
14. Kumar, R., Rosé, C.P., Wang, Y.C., Joshi, M., Robinson, A.: Tutorial Dialogue as Adaptive Collaborative Learning Support. In: Proceedings of AI in Education (2007)
15. Shum, B.S.: Workshop report: Computer Supported Collaborative Argumentation for Learning Communities. In: SIGWEB Newsletter, pp. 27–30 (2000)
16. Pasca, M., Harabagiu, S.: High Performance Question/Answering. In: Proceedings of SIGIR 2001, pp. 366–374 (2001)

How Who Should Practice: Using Learning Decomposition to Evaluate the Efficacy of Different Types of Practice for Different Types of Students

Joseph E. Beck[1] and Jack Mostow[2]

[1]Computer Science Department, Worcester Polytechnic Institute
[2]Robotics Institute, Carnegie Mellon University
joseph.beck@EducationalDataMining.org, mostow@cs.cmu.edu

Abstract. A basic question of instruction is how much students will actually learn from it. This paper presents an approach called *learning decomposition,* which determines the relative efficacy of different types of learning opportunities. This approach is a generalization of learning curve analysis, and uses non-linear regression to determine how to weight different types of practice opportunities relative to each other. We analyze 346 students reading 6.9 million words and show that different types of practice differ reliably in how efficiently students acquire the skill of reading words quickly and accurately. Specifically, massed practice is generally not effective for helping students learn words, and rereading the same stories is not as effective as reading a variety of stories. However, we were able to analyze data for individual student's learning and use bottom-up processing to detect small subgroups of students who did benefit from rereading (11 students) and from massed practice (5 students). The existence of these has two implications: 1) one size fits all instruction is adequate for perhaps 95% of the student population using computer tutors, but as a community we can do better and 2) the ITS community is well poised to study what type of instruction is optimal for the individual.

Keywords: learning decomposition, educational data mining, learning curves, bottom-up processing.

1 Introduction

The goal of this paper is to investigate how different types of practice affect a student's progress in learning a skill. Specifically, we utilize an approach, learning decomposition [1], as a means of leveraging fine-grained interaction data collected by computer tutors and present a case study of applying the technique to the domain of reading. The goal is twofold: 1) Be able to make claims that are interesting to domain researchers, and 2) Develop a technique for analyzing tutor log data that applies to other domains and tutors. The first goal should not be underestimated; if we make discoveries about how students learn a domain that remain limited to those students using computer tutors that would be an unfortunate result. Only a small minority of students use computer tutors, so if we wish our research to have broad impact then

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 353–362, 2008.

finding a means of explaining our results to those outside of the ITS community is essential. To address these issues we present an approach that uses learning curves to measure the relative impact of various types of learning events.

The two most common types of learning curves are exponential and power curves. In this paper we discuss exponential curves as they have been shown to more accurately model individual observations [2] and are simpler analytically. However, the approach we present can be trivially adapted to work with power curves. The standard form of the exponential learning curve can be seen in Equation 1a. The free parameter *A* represents how well students perform on their first trial performing the skill; *e* is the numerical constant (2.718), the free parameter *b* represents how quickly students learn the skill, and *t* is the number of practice trials the learner has had at this skill. This model can be solved with any non-linear regression package (we use SPSS 11.0).

$$performance = A * e^{-b*t} \qquad performance = A * e^{-b*(B*t_1 + t_2)}$$

Equation 1. (a) Exponential model of practice, (b) Learning decomposition model of practice

By simply using one parameter, *t*, to represent the number of prior trials, learning curves assume that all types of practice are equally valuable. But what if all types of practice are not equally valuable? For example, we could believe that the subject will learn better the first time he practices the skill that day, and rather than simply lumping all of the learning opportunities together as *t*, we can create two new variables t_1 and t_2. The variable t_1 represents the number of learning opportunities where it was the first time the learner practiced the skill that day; t_2 represents the number of practice opportunities where the learner has already practiced the skill that day. This method of factoring learning opportunities into various types of practice does not change the amount of prior practice to student has had; $t = t_1 + t_2$ since learning opportunities are either the first one of the day or are not.

The basic idea of *learning decomposition* is to find how to weight two types of learning opportunities to construct a best fitting learning curve. Equation 1b shows a learning curve model designed to find how to weight the two types of practice. Similar to standard learning curves, we estimate the *A* and *b* parameters. However, we also estimate a new parameter, *B,* that represents the relative impact of the first learning opportunity of a day relative to learning opportunities occurring later in the same day. Note that t_2 does not receive a weight of its own, as it is assumed to be worth 1.0 learning opportunities. That t_2 has this implicit weight does not affect the conclusions we draw from the model as our goal is only to estimate relative efficacy the two types of practice.

The parameter *B* is very interpretable: it is how many trials that learning opportunities of type t_1 are worth relative to those of type t_2. If $B>1$ then learning opportunities of type t_1 are better for learning than those of type t_2. If $B<1$ then the opposite is true, and if $B=1$ then neither type of learning opportunity is preferable. Although the example presented is about first practice opportunity of the day vs. later ones, it is possible to split the data in any way that may be interesting. We could split learning opportunities by those that occur on Monday, Wednesday, or Friday vs. those that occur on Tuesday and Thursday. For this decomposition we would hopefully get $B \approx 1$,

as we have no reason to believe the day of the week matters for learning. Thus, the technique of learning decomposition is broadly applicable.

The remainder of this paper explores applying learning decomposition to answer some questions about how children acquire reading skills. However, the approach itself is applicable to a variety of learning tasks and possible ways to decompose learning.

2 A Case Study: Applying Learning Decomposition to the Domain of Reading

The goal of this case study is to show how to apply learning decomposition to an actual data set and draw scientifically useful conclusions. We are trying to better understand how students learn to read by analyzing performance data about individual words recorded by the Reading Tutor [3] during the 2003-2004 school year. Rather than have explicit experimental and control groups, our approach is to examine how student progress in reading words quickly and accurately varies based on which type of practice he has had at the word. These data include 346 students from the Pittsburgh area attempting to read 6.9 million words. The student readings were scored by an automated speech recognizer (ASR). The ASR is far from perfect, and for that year detected approximately 25% of student misreadings and scored 4% of student correct readings as incorrect [4]. The ASR also records how long students took to read a word. Our general logging mechanism also records when students request help. Furthermore, all entries are time stamped so we know the relative temporal relations between events. Students used the tutor from September 2003 through May 2004 with a median usage of 5.9 hours.

We now show how we integrate student help, speed, and correctness into a single outcome measure of learning; explain what we believe constitutes a learning opportunity for a word; and finally show how we decomposed the learning opportunities into their component parts.

2.1 Creating an Outcome to Measure Learning

There are a variety of approaches for representing student performance at reading fluency. We choose to model the student's reading time since it is a continuous variable and best able to track student progress; help requests and accuracy are binary and so cannot improve smoothly. Although it is possible to aggregate help requests and accuracy to create a continuous learning curve, we did not perform such aggregations as one goal of the research is to use individual observations (rather than aggregate descriptions) to construct our learning curves. It is a known potential pitfall that aggregate learning curves may not describe the learning trajectory of actual individual learners [5]. Therefore, fitting individual data points can produce a more authentic model of student learning

Although reading time is continuous, it is misleading to use it as an outcome and ignore accuracy and help requests. Our approach was to use the student's reading time as an outcome measure. However, when the student either asked for help or skipped the word, or the word was scored as incorrect by the ASR, then that word was

assigned a reading time of 3.0 seconds. Also, words whose reading time was greater than 3.0 seconds were capped at 3.0 seconds. The penalty of 3.0 seconds is on the high end of reading times as only 0.1% of time exceeded this threshold, but not overly so as to be an unfair penalty.

2.2 What Constitutes a Learning Opportunity?

Given that help can cause a short-term boost in student performance, a natural question is what other types of events can cause a similar effect? If our goal is to measure student *learning*, we should try to exclude such data from our learning curve construction. One example of such short-term scaffolding is that if a student reads a word and then shortly thereafter reads that same word again, we should be skeptical that the second reading really demonstrates the student's knowledge of the word (as opposed to just retrieving it from short term memory). Therefore, to model student reading development we only consider as an outcome variable his first encounter with a word on a particular day.

However, we do count subsequent encounters later in the day as opportunities to *learn* the word. Table 1 illustrates our approach. For the first encounter, the student requests help and then reads the word quickly. Since the student requested help, the outcome is set to 3.0 seconds. For the next learning opportunity, since it is the same day, that reading does not count as an outcome. Similarly, the next learning opportunity's performance is also ignored. However, note that column labeled "Overall" in the prior encounters field, which tracks the student's experience with this word, has been incremented to account for these two exposures.

2.3 Learning Components of Fluency Development

For reading, what types of practice are likely to be more (or less) effective for students' fluency development? There are many possible ways to think about what are ways of factoring apart learning opportunities at learning to read a word. We start with a known psychology principle: distributed practice is generally superior to massed practice for long term retention [6]. This general rule suggests a decomposition: we consider a learning opportunity as *distributed* practice if the student has not encountered the word in the preceding 16 hours. *Massed* practice would be times when the student encountered the word in the prior 16 hours (effectively during the same day). Table 1 shows how we decompose the prior encounters based on massed vs. distributed practice.

The other type of learning decomposition we performed was to examine whether reading the same story multiple times provides the same benefits as students reading different stories. This debate of wide- vs. re-reading has been ongoing in the reading community. We therefore decompose prior practice into learning opportunities where this student encounters this word while reading *new* material vs. *rereading* old stories. Since students can memorize a particular story, we only permit as an outcome variable the first time a student reads a particular story. However, analogous to how we handled massed practice learning opportunities, repeated readings of the same story count as learning opportunities for learning (in particular, the variable for *rereading* would be increased in each case).

To summarize, we only count the first opportunity each day as an outcome variable, and only if the student has not read this story in the past. However, we count all exposures to words as possible learning opportunities. To estimate the learning caused by different types of learning opportunities, we created four types:

1. RM represents **r**ereading-**m**assed learning opportunities. I.e. cases where the student has already read the story in the past and is seeing the word a second (or greater) time today.
2. RD represents **r**ereading-**d**istributed learning opportunities. I.e. cases where the student is rereading the story but has not seen the word earlier today.
3. NM represents **n**ew-**m**assed learning opportunities; cases where students are reading a story for the first time and have read the word previously today.
4. ND represents **n**ew-**d**istributed learning opportunities; students have not seen this story before and have not read the word previously today.

Table 1. Decomposing prior learning opportunities as massed and distributed practice

Day	Helped?	Reading time (seconds)	Prior encounters			Outcome (seconds)
			Overall	Distributed	Massed	
1	Yes	0.5	0	0	0	3.0
1	Yes	1.5	1	1	0	-
1	No	1.3	2	1	1	-
2	No	3.8	3	1	2	3.0
3	No	1.7	4	2	2	1.7
3	No	1.2	5	2	3	-

Our model of reading development is shown in Equation 2. The term *A*, represents first trial performance, and *b* is the rate of learning. For brevity, the model presented omits some terms such one to control for word length (since reading time is correlated with word length) and another to control for amount of prior assistance. The remainder of the model is a learning decomposition model to simultaneously estimate the impact of massed- vs. distributed-practice and wide- vs. re-reading. Note that RM, RD, NM, and ND account for all possible trials, and are thus equal to *t*. The goal is to find best-fitting values of the *r* and *m* parameters to find the relative impact of rereading and massed practice, respectively, on student reading development.

$$readingTime = A * e^{-b*(r*m*RM + r*RD + m*NM + ND)}$$

Equation 2. Simplified model for examining effect of practice schedule and type of reading

Again, there are many possible ways to decompose learning opportunities. We chose two that were motivated by existing theories of learning and a current debate in the reading literature.

3 Results

To train the model, we had 959,455 learning opportunities (i.e. a student's first attempt at reading a word on a particular day) and a total of 6.9 million words read. For each of the 346 students in our data set who read at least 20 words in the Reading Tutor during the 2003-2004 school year, we fit the model shown in Equation 2 to each student's data (i.e. we had 346 estimates of each parameter—one for each student). Table 2 shows the median parameter estimates for the effects of rereading and massed practice. The column labeled "overall" contains the median for the entire population. The next three columns are estimates by the bottom third (reading pretest score below a beginning first grader), middle third, and upper third (reading pretest above that of a second grader at mid year) of the student population.

Main Effects. We found that rereading had a coefficient of 0.49 for the entire student population. In other words, rereading a story only results in 49% as much learning as reading a story for the first time. So if a student reread a story twice that would result in as much learning as reading a new story for the first time ($2 * 0.49 = 0.98 \approx 1.0$). Therefore, our results suggest that students learn to read words better when they read a wide selection of stories rather than read the same story multiple times, and this trend holds for all of the levels of student proficiency that we examined. For massed practice, the picture was even more bleak. Overall, students learned very little from multiple opportunities to practice a word on the same day, with high proficiency students deriving almost no benefit at all from the exposure. Seeing the word again is almost a complete waste of time for these students.

Table 2. Median parameter estimates for learning decomposition model

	Overall (N=346)	Low proficiency (N=118)	Medium proficiency (N=106)	High proficiency (N=122)
Reread (r)	0.49	0.71	0.42	0.33
Massed (m)	0.19	0.36	0.28	0.02

We report median rather than mean scores due to difficulties with accounting for outliers. For example, student rereading parameters range from -1754 to 14211. Clearly those extreme value are outliers and would bias the mean. However, it is difficult to determine exactly what constitutes an outlier. For example, 3.0 is an unlikely level of benefit from rereading, should we disallow that? How about 2.0? Rather than inventing an arbitrary cutoff, we instead use the median and treat improbably high values as a vote that the true value of the parameter is higher than 1.0.

For the rereading parameter, 95 students had an *r* parameter that was reliably less than 1.0, while only 7 had a parameter estimate that was reliably greater than 1.0. Using a sign test gives $p \approx 10^{-17}$, thus the majority of students have less effective learning as a result of rereading. For massed practice, 177 students had an *m* parameter that was reliably less than 1.0, with only 6 students having an *m* parameter reliably greater than 1.0 ($p \approx 10^{-35}$). Thus, the majority of students benefit less from massed practice. This result for massed practice is not novel, as there has been ample research in

psychology investigating spaced practice effects. However, it serves as a sanity check on our results: if this aspect our investigation disconfirmed 120 years of psychology research we should be hesitant about accepting our other results.

Which Students Benefit from Rereading and Massed Practice?. One benefit of estimating a per-student parameter for the effect of rereading and massed practice is that it enables fine-grained detection of student subgroups who benefit. Although Table 2 shows that low proficiency students do not benefit from rereading or from massed practice, there is a definite trend with weaker readers receiving relatively more benefit than more proficient readers. Perhaps there are subgroups of low proficiency students who are benefiting, but we cannot detect them since they are averaged in with a larger group?

Our approach is to treat the problem as one of classification via logistic regression. For our dependent variable, if the student's *r* parameter exceeded 1.1, we treated that student as benefiting from rereading; if it was below 0.9 the student is considered to benefit from wide reading. Values between 0.9 and 1.1 were not considered conclusive evidence either way, and those students were not used for the classification process. By compressing the student's *r* parameter into a single bit of information, we greatly reduce the inaccuracy of poorly estimated per-student parameters, and instead focus on the simpler task of finding commonalities between students whose *r* parameter indicates they would benefit from rereading. The act of creating a classifier results in a smoothing of the noisy data, and any reliable predictors are indicative of a subgroup that benefits. (We performed the identical procedure for the *m* parameter to determine if students would benefit from massed practice.) For the independent variables, we used the student's gender, grade, learning support status (yes, no, or not known), and words read correctly per minute in a paper-based fluency pretest. We chose these variables as they are easily available to classroom teachers or other classroom policy makers.

Of the 235 students for whom we had complete demographic and testing data, the rereading classifier found a subgroup of 11 students for whom it thought rereading would benefit. The student's learning support status ($p=0.00003$) and fluency ($p=0.04$) were both reliable predictors in the model. Only 24 students were noted as having learning support, of those the model felt 4 would benefit from rereading (as opposed to predicting benefit for 7 of 122 for students who were known to not be receiving learning support). The students who would benefit from rereading also had a sharply lower fluency: 44 words per minute as opposed to 56 words per minute for those who would not benefit. There were similar trends for the students who would benefit from massed practice. The classifier found a subgroup of just 5 students who would benefit from massed practice, with the only reliable differences being the student's grade, with a mean of 2.15 from those who do not benefit vs. 4.6 for those who do ($p=0.013$) and fluency, with a mean reading rate of 55 wpm for those who do not benefit vs. 47 for those who do ($p=0.04$). Although not statistically reliable, it is interesting to note that all 5 of these students were categorized as receiving learning support. As a general trend, those who benefited from massed practice and from repeated reading were older, less proficient readers who were tagged as requiring learning support (but who were still able to operate the Reading Tutor software effectively). Gender was not a reliable predictor in either model.

4 Limitations and Future Work

This paper explored two learning decompositions, but there is a large space of possible ways of splitting the data. Automating the construction and evaluation of possible decompositions is a fruitful avenue of research. One crucial problem that must be overcome is finding some method for seeding this search space with expert knowledge. Expert knowledge both reduces the size of the space and biases the results so that it better fits with what is known. The output of educational data mining can certainly improve computer tutors, but if that is all it does that would be unfortunate. As a field, we have several novel methodological hammers that are unavailable to domain researchers who aren't using these approaches. We must find ways to transfer what we learn to the broader research community. By hand selecting two major hypotheses of learning and reading, we manually biased the search to have output that will (hopefully) have high impact. Can we have automated search that produces results that are equally shareable?

Given that we have a set of high-level decompositions to perform, we still need to operationalize them. For our analysis, massed practice was equated with seeing a word a 2nd time on the same day. However, there are many ways to view "massed practice." Perhaps it means more than 5 practice attempts within 3 minutes? Maybe the first 2 attempts on the same day aren't massed but subsequent ones are? There is a wide space of possible ways to instantiate the theory. How do we know which is best? Searching across instantiations of distributed practice is itself a large search space. Can we afford to search both the space of good decompositions and the ways to instantiate the decompositions?

One hybrid approach is to accept that the high-level decompositions should come from humans who will (hopefully) use existing theory (e.g. mass vs. distributed practice) to generate decompositions, and spend computational resources on exploring that search space to find a good way to operationalize the decompositions. Such an approach would seem to draw on the strengths of humans and computers. Science is a social process and we need results that fit with existing theory (perhaps to disconfirm it) for domain researchers to take it seriously [7]. Spending time searching for new theory may not be productive if no one understands the results. However, a specific instantiation of an expert generated decomposition should be understandable. For example, if instead our model of reading used the "5 practice attempts within 3 minutes" definition of massed practice, the results wouldn't be any harder or easier for others to understand. Such a hybrid approach seems a promising route forward.

We would like to make statements of the form "Rereading is less helpful for developing reading proficiency than wide reading." Unfortunately, our data were not gathered from randomized trials, but rather are observational in nature. However, by estimating practice effects per-student, we are able to make stronger claims than might be expected. For example if Chris has a rereading parameter of 0.8, that means when rereading he learns 80% as much *as Chris would normally learn*. By having the student act as his own control, we remove other constant factors that could act as confounds for our result. For example, if less proficient students are the ones doing more rereading, our result would not be biased by the fact. In general, by building per-student models we control for all *trait* information about the learner such as language aptitude, memory capacity, interest in learning, etc. However, we do not

control for transitory *state* information. For example, if students only reread when they are tired after a poor night's sleep (and are presumably less able to learn), that is a possible confound for our results.

Finally, our results were based on a set of assumptions about which words (first attempt at reading a word for the day) and which stories (first time the student read the story) were most indicative of learning. These assumptions represent a best-effort on the part of the authors. However, a sensitivity analysis of specific the results are to our assumptions would be helpful. We performed one such analysis and found that, qualitatively, the results were similar whether we looked at the first time a word was read in a day or the first time in a week.

5 Contributions and Conclusions

This paper makes several contributions both methodologically and scientifically. From a methodology standpoint, learning decomposition extends learning curve analysis to enable estimation of the impact of various types of learning opportunities. The learning decomposition approach is broadly applicable to a wide variety of learning phenomena and is not specific to reading. Furthermore, it is fast computationally and can be applied via a variety of off the shelf software packages. Finally, the output is easy to interpret and share with other researchers.

From a scientific standpoint, this work may resolve a debate in the reading community (is wide- or re-reading better and for whom). If the goal of our work is to have impact beyond our own tutors, finding modeling approaches that are easily understandable by other communities must be a priority. Our results on what type of reading practice helps the most have not yet been fully disseminated to the reading community so it is premature to comment on whether this approach will result in conclusions understandable to domain researchers. However, an earlier version of this work was presented at the 2005 and 2006 Scientific Studies of Reading Conferences and was well received.

The closest related research is learning factors analysis (LFA) (e.g. [8]) Both LFA and learning decomposition are concerned with better understanding student learning. LFA focuses on modifying the domain representation by adding, removing, or combining skills to create better fitting learning curves where the impact of various types of practice is assumed to be constant. Learning decomposition focuses on determining the impact of various types of practice, and assumes the domain representation is constant. A unified framework that simultaneously allows both the skills and impact of practice to vary would be desirable.

In conclusion, we have shown how learning decomposition can be applied to use observational data to estimate the effectiveness of different types of learning opportunities. Our analyses show that in the domain of reading, different types of practice are more effective than others. Specifically, reading new stories and spacing exposure to words is good for long-term learning. Although our case-study was in the domain of reading, there is nothing domain specific about the learning decomposition approach, and it is broadly applicable to a variety of ITS. Furthermore, the massed practice result has implications both for sequencing instruction and for student modeling in an ITS. If a student model is not discounting learning opportunities that are temporally

near each other, it is probably overestimating student knowledge. Finally, our bottom-up approach of using classification to detect important student subgroups, rather than relying on *a priori* beliefs about what disaggregation are important, was able to detect a subpopulation of students who benefits from an otherwise less effective treatment. If fully realized, this capability to truly adapt an ITS's instruction to meet the needs of learners would be a large step forward an ITS.

Acknowledgements

This work was supported by the National Science Foundation, ITR/IERI Grant No. REC-0326153. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation

References

1. Beck, J.E.: Does learner control affect learning? In: Proceedings of the 13th International Conference on Artificial Intelligence in Education, Los Angeles, pp. 135–142 (2007)
2. Heathcote, A., Brown, S., Mewhort, D.J.K.: The Power Law Repealed:The Case for an Exponential Law of Practice. Psychonomics Bulletin Review, 185–207 (2000)
3. Mostow, J., Aist, G.: Evaluating tutors that listen: An overview of Project LISTEN. In: Feltovich, P. (ed.) Smart Machines in Education, pp. 169–234. MIT/AAAI Press, Menlo Park (2001)
4. Banerjee, S., Beck, J., Mostow, J.: Evaluating the Effect of Predicting Oral Reading Miscues. In: Proc. 8th European Conference on Speech Communication and Technology (Eurospeech 2003), Geneva, Switzerland, pp. 3165–3168 (2003)
5. Brown, S., Heathcote, A.: Averaging learning curves across and within participants. Behaviour Research Methods, Instruments & Computers 31, 11–21 (2003)
6. Ebbinghaus, H.: Memory: A Contribution to Experimental Psychology. Teachers College, Columbia University, New York (1885)
7. Pazzani, M.J., Mani, J.S., Shankle, W.R.: Acceptance by medical experts of rules generated by machine learning. Methods of Information in Medicine 40(5), 380–385 (2001)
8. Cen, H., Koedinger, K., Junker, B.: Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement. In: Intelligent Tutoring Systems, pp. 164–175. Springer, Jhongli (2006)

How Does Students' Help-Seeking Behaviour Affect Learning?

Moffat Mathews and Tanja Mitrović

Computer Science & Software Engineering,
University of Canterbury, New Zealand
{moffat,tanja}@cosc.canterbury.ac.nz

Abstract. We examined high-level help (HLH) seeking behaviour of students by data mining in SQL-Tutor. Students who used HLH very frequently had the lowest learning rate; their learning was also shallow. They attempted very difficult problems compared to other groups but only solved very easy problems, suggesting that they were usually situated well beyond their Zone of Proximal Development. They also abandoned a large number of problems without solving them. Manual inspection of the logs showed erratic problem solving behaviour, suggesting a "guess and copy" strategy.

The group of students who used HLH very infrequently seemed to contain two distinct sub-groups: students with high expertise, and students with very low expertise who still did not use HLH. Learning rates were highest for students who used moderate HLH. Students with lower usage of HLH solved the most difficult problems comparatively, without the use of HLH, and had high learning rates, suggesting the ITS is most beneficial for this group of students.

Keywords: Help-seeking behaviour, data mining.

1 Introduction

This paper presents the initial results of data mining student models and log files in SQL-Tutor [1]. The research conducted in this paper is the first in a series of steps towards a project designed to develop a general framework for adapting pedagogical strategies in Intelligent Tutoring Systems (ITSs). Part of the challenge of designing such a framework is to identify the various contexts in which students find themselves when working in an ITS, and understand their behaviour when confronted with such a context. As researchers and developers, we are also interested in the effects of this behaviour on learning. Research is continuing into various aspects of help seeking behaviour [2], including investigating the misuse of help and feedback in ITSs [3,4]. In this step of the research, we explore one aspect of help-seeking behaviour in students: the frequency of high-level help (HLH) sought and its effect on learning.

Help in an ITS usually consists of tutor-specific help and domain-specific help. Tutor-specific help is help regarding the ITS itself. For example, this type

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 363–372, 2008.

of help might show the student steps on how to submit a particular problem, or how to request a new problem. In contrast, domain-specific help concentrates on concepts within the domain irrespective of any tutoring system used.

Domain help can be categorised in many ways. One way is to class it into either problem-dependent or concept-dependent help. Examples of problem-dependent help are hints or feedback messages provided for a particular problem or step within that problem. These can either be given before the problem is attempted (forward hints) or after a solution has been submitted (feedback). Concept dependent help is help on domain concepts and is usually given in the form of tutorials or feedback during a meta-cognitive dialogue (e.g. during reflection or self explanation).

Domain help can also be categorised into adaptive or non-adaptive help. Adaptive help is customised for the particular student, and is based on the student's solution or their student model. An example of non-adaptive help is the full solution.

For this research, we have also categorised domain help into low-level help (LLH) and high-level help (HLH). Low and high level refer to the degree of help given. For example, showing the student a partial or full solution is classed as high-level help, whereas telling the student that they have made some errors is classed as low-level help. Sometimes, the process of giving higher degrees of help till the highest degree is reached is referred to as "bottoming out". The highest degree of problem-dependent help is usually the full solution, either for the step or the entire problem. In this paper, we focus on the use of LLH and HLH to explore its effect on learning.

2 Problem-Dependent Help in SQL-Tutor

SQL-Tutor is an ITS designed to guide and support students in their deliberate practice in the domain of querying databases in Structured Query Language (SQL). SQL-Tutor has a long history of high learning rates and evaluation studies, and is used in tertiary database courses as part of the curricula. Being a constraint-based modelling tutor, it stores domain knowledge as constraints (domain principles) and provides the student with multiple levels of domain-help depending on various factors, such as their current solution, their student model, and the constraints violated on this submission.

In SQL-Tutor, the problem-dependent, domain help is divided into six numbered categories. These categories in order of degree of help are: 1. Simple feedback, 2. Error flag, 3. Hint, 4. Partial solution, 5. List all errors, and 6. Full solution. The help level (also known as feedback level) is controlled on the interface by means of a simple combo box (see Fig. 1). At the start of a new problem, the tutor defaults to help level 1. On subsequent incorrect submissions, the tutor automatically increments the help level by one, to a maximum of 3 (i.e. hint). The student can override the help selection at any time by changing the value of the combo box. Help levels 4, 5, and 6 have to be specifically requested by the student. For this research, we have classed help levels 1-3 (inclusive) as

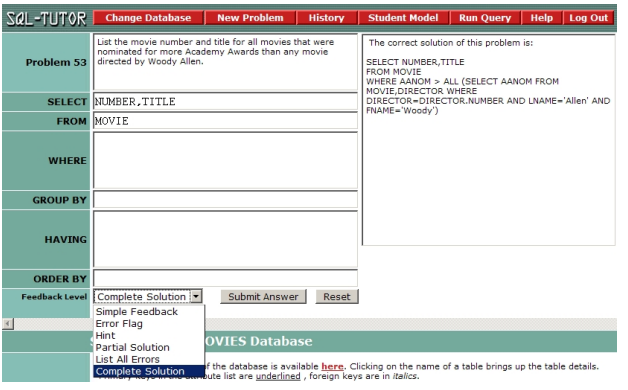


Fig. 1. The SQL-Tutor task environment showing the combo box with the various levels of problem-dependent help. Here, the student has requested HLH (in this case, the full solution) for the problem, which is displayed in the feedback pane.

LLH, and help levels 4-6 (inclusive) as HLH. Furthermore, LLH is automatically incremented, but HLH is requested by the student.

When a student reaches the task workspace (Fig. 1) in SQL-Tutor, they are presented with the problem in text format. More information regarding the context of the problem, such as information about the schema, is also presented with each problem. The solution workspace contains text areas in which students can work on their query. Once the student is content with their solution, they can submit it and receive feedback. The degree to which this feedback is given is dependent on the help level selected, as discussed above. It is this help - and more specifically, the difference between LLH and HLH used by students, that interests us for this paper.

3 Learning Curves

Learning curves are used to plot students' learning over time. Learning a skill generally follows a power law, where the greatest improvement occurs early in the learning process. The formula for the power law is given in equation 1. More details about learning curves can be found in [5,6].

$$E = \chi n^{\alpha}. \tag{1}$$

Where: E is the error rate, χ is the performance on the first trial, n is the opportunity to practice the skill, and α is the learning rate.

Four main points are worth noting. First, the learning rate (α) shows the speed at which errors are reduced over the number of occurrences of a particular skill or concept. The slope of the graph at various points depicts the learning rate at that point. Second, χ describes the students' performance on the first trial, and therefore shows how difficult the students found the particular domain or

problem set; the higher the number, the more difficult. Therefore, if the domain or problem set was kept similar and yet the χ values differed between groups of students, it could be argued that the group with the higher initial error rate (i.e. the group that performed better on the first trial and therefore found the domain easier than the other groups) must have had prior domain knowledge or higher expertise than the other groups. Third, the fit of the graph (R^2) illustrates the variability of the data within that group. The fit shows how well students used the skill they learned previously i.e. transferability of the skill learned. The better the fit, the higher the transferability. And finally, learning curves gradually decrease in slope over time i.e. the learning rate reduces. The value of the bottom of the curve shows the probability of the student making an error in subsequent occurrences of the same concept i.e. how well the student has learned the concept. If the learning rate has decreased to near-zero, and the probability of making errors is still quite high (the bottom of the curve is high), then the student's learning can be considered shallow. The lower the bottom of the curve, the deeper the learning.

4 The Data and the Methods Used

The main dataset (named dataset A) for this research was taken from an online version of SQL-Tutor that is available to students from around the world who were given free access when they bought certain SQL text books. Data for any student that made less than five attempts was excluded. The remaining data consisted of 1803 users who made a total of 100,781 attempts and spent just over 1,959 active hours on the system. Active hours is time spent actively solving the problem; not just session times. Students in this dataset on average solved 70% of the problems attempted; giving a grand combined total of 19,604 solved problems.

We extracted the number of submissions for each student from the individual student logs. Each submission was then categorised as either a valid attempt or a request for help (RFH).A valid attempt occurs whenever a student submits a solution that is different to their previous solution. An attempt need not be correct to be valid. When a student submits either an empty solution or the same solution twice in a row, this is interpreted as a RFH. Here, the student is hoping that more hints will be given on each submission. SQL-Tutor automatically increments the help level to a maximum of 3. Equation 2 shows the relationship between submissions, attempts, and RFH.

$$\text{Submissions} = \text{Attempts} + \text{Requests for help} \tag{2}$$

We further categorised each valid attempt into two categories: high-level help (HLH) attempts or low-level help (LLH) attempts, and deduced the total number of HLH attempts for each student. To normalise the HLH attempts value over all the students, we calculated an HLH ratio; $0 \leq HLH \leq 1$. (See equation 3). This ratio categorises students' HLH seeking behaviour by showing us how frequently a student uses high-level help; low frequency HLH students are those that used

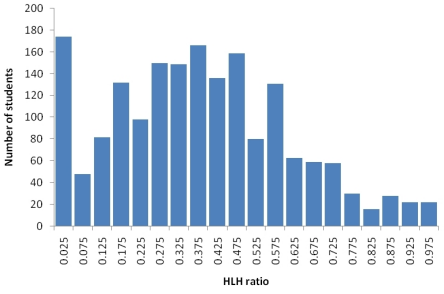


Fig. 2. Frequency distribution of users in the various HLH user groups in dataset A

high-level help very infrequently and vice versa. For example, a student with an HLH ratio of 1 uses HLH on every attempt, whereas a student with HLH ratio of 0.1 uses HLH once every ten attempts. Students were then ordered according to their HLH ratio; from low frequency HLH users to high frequency HLH users.

$$\text{HLH ratio} = \frac{\text{Number of high-level help attempts}}{\text{Total number of attempts}} \tag{3}$$

A frequency graph of students and their HLH ratio is shown in Fig. 2. We then divided the population into ten groups (A1-A10) depending on their HLH ratio; the groups were 0.1 HLH apart. The logs and student models were analysed for each group, and learning curves were plotted.

Each problem in SQL-Tutor is assigned a difficulty level by the teacher. Difficulty levels range from 1 (easiest) to 9 (most difficult). For each student, we mined the difficulty levels of problems attempted and recorded the *maximum difficulty level of problem solved* (MDLS). In our initial analysis, we used the MDLS to see how students from each group were able to learn skills and progress through to more difficult problems. Manual analysis of the logs showed that for students in the higher HLH groups, there was a large difference between the difficulty level of problems attempted and those solved; they attempted much more difficult problems than they solved. There also seemed to be a high number of abandoned problems after valid attempts were made. Furthermore, in many cases the initial (incorrect) solutions were very different to the ideal solution. It was as though the high HLH students were employing a "guess then copy" method to solving problems. In contrast, the lower HLH students created answers that were usually more similar to a correct solution, then with each attempt submitted closer approximations to a correct solution until the problem was solved. Due to this, another metric, termed the *maximum difficulty level of problem solved without using HLH* (MDLS-WH) was also calculated and used.

The results are summarised in the next section.

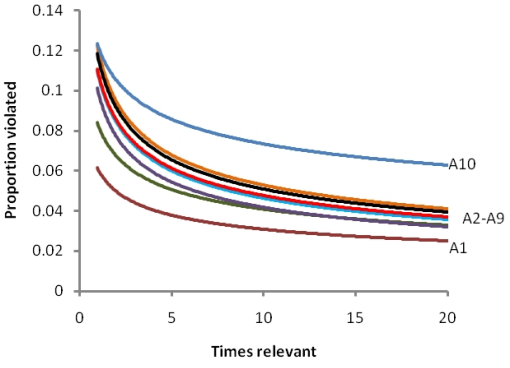


Fig. 3. The learning curves for the ten HLH user groups (A1-A10)

5 Results and Discussion

5.1 Frequency Distribution of Users According to Their HLH

The frequency distribution graph in Fig. 2 shows the number of users grouped according to their HLH. This graph has an interesting shape. From $HLH > 0.05$, a positive skewed normal curve exists. There is also a peak external to the normal distribution in the first section of the graph: $HLH \leq 0.05$; these are students that are very low users of HLH. Manual inspections of the logs seem to indicate the presence of at least two distinct types of students within this first section ($HLH \leq 0.05$): one with higher domain expertise and the other with low domain expertise. This section of students needs to be analysed further. It is understandable that the students with high expertise are low users of HLH. However, the students with low expertise who do not use HLH could be those who either have low meta-cognitive (help-seeking) skills or, due to social norms, feel that looking at HLH constitutes a form of "cheating". For comparison, frequency graphs were also plotted for two other smaller sets of data, and we found that the results were very similar. This shows that this trend is persistent across populations, and requires deeper analysis.

5.2 Learning Curves for Each HLH User Group

Learning curves were calculated and drawn for each of the ten groups (A1-A10). Fig. 3 shows the learning curves for all the groups plotted on the same graph for ease of comparison. To avoid clutter, the equations have been excluded from the graph but are listed in table 1.

Several points can be noted from the learning curves:

1. The curves follow the same approximate order of the HLH use. The higher the HLH use, the shallower the learning. For example, students in the group A10 (those with the highest HLH) portray shallow learning; even when their

Table 1. Power curve equations and fits (R^2) for the ten HLH groups (A1-A10)

Group	HLH ratio	Power curve equation	R^2 (Fit)
A1	0.0 - 0.1	$y = 0.061x^{-0.30}$	0.844
A2	0.1 - 0.2	$y = 0.084x^{-0.31}$	0.956
A3	0.2 - 0.3	$y = 0.101x^{-0.38}$	0.955
A4	0.3 - 0.4	$y = 0.109x^{-0.37}$	0.956
A5	0.4 - 0.5	$y = 0.110x^{-0.36}$	0.965
A6	0.5 - 0.6	$y = 0.123x^{-0.39}$	0.961
A7	0.6 - 0.7	$y = 0.122x^{-0.36}$	0.953
A8	0.7 - 0.8	$y = 0.115x^{-0.35}$	0.953
A9	0.8 - 0.8	$y = 0.118x^{-0.36}$	0.912
A10	0.9 - 1.0	$y = 0.123x^{-0.22}$	0.956

learning rate approaches zero, their probability of making errors is still comparatively high.

2. The fit for power curve is very high across all groups. The lowest fit for curve is in A1 ($R^2 = 0.844$). This could be because, as proposed earlier, A1 contains at least two distinct sub-groups of students: the students with high expertise, and the students with low expertise who persevere with problem solving without utilising HLH. The transferability of skills could vary within these sub-groups, reducing the overall fit.
3. Since learning curves A2-A9 are very similar, this leaves us with three distinct categories of learning curves: A1, A2-A9, and A10. Fig. 4 shows the three learning curves plotted on a single graph. The middle range of HLH users displays similar learning (i.e. depth, learning rate, etc.), while the extreme high and low HLH users display markedly different learning.
4. The χ value increases as HLH use increases. This means that high HLH students find the domain more difficult than low HLH students. There is a marked difference between the χ value of A1 and the other groups. This could also show that A1 contains experts (or at least students with prior domain knowledge).
5. Curves A2-A9 report similarly high learning rates. A10, the highest HLH has the lowest learning rate (0.22). This could be because students that use HLH extremely frequently do not actively engage in the material, think for themselves, utilise their meta-cognitive skills (such as reflecting or explaining their actions), or participate in the benefits of deliberate practice (e.g. learning from errors). This causes much lower learning rates. The group of users that use HLH the least (i.e. group A1) has the second lowest learning rate. This could be because both the sub-groups (experts and low expertise students) in A1 both have low learning rates. The experts in A1 already find the domain easy, thus starting with low error rates (i.e. χ is low), and therefore reach their asymptote of learning very quickly. The novices in A1 who persevere with problem solving without utilising HLH also have low learning rates. In this case, the ITS is therefore most beneficial for the majority of students who are in the middle HLH range, producing relatively similar, high learning rates.

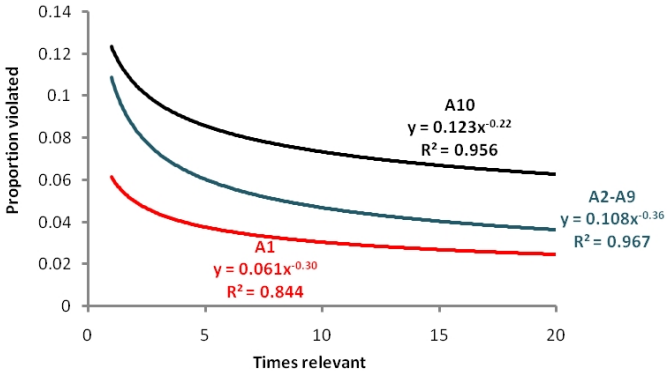


Fig. 4. The learning curves for the HLH user groups. In this graph (unlike Fig. 3), groups A2-A9 are combined. This was done as the curves for A2-A9 were very similar in nature.

5.3 Maximum Difficulty Level of Problems Solved by Students

The Zone of Proximal Development (ZPD) [7] is the area just beyond the student's ability or more specifically, it is the difference between what a learner can do without help and what they cannot do without help. In ITS terms, the problems found within the ZPD are just challenging enough for the student to solve with some help (i.e. LLH and moderate amounts of HLH) from the tutor. It is also said to be the area in which the highest rate of learning occurs. Like most ITSs, SQL-Tutor attempts to keep the student within their ZPD by guiding them through increasing levels of difficulty, while providing LLH, as their expertise in the domain increases; the student is also free to use HLH as necessary.

As mentioned earlier, problems in SQL-Tutor range in difficulty level from 1 to 9. Although the range (1-9) seems small, there is quite a difference in difficulty between levels, such that the difficulty (and type of concepts covered) between a problem of level 1 and another of level 5, for example, is considerable.

As discussed in the section above, we initially collected the *maximum difficulty level of problems solved* (MDLS); this included problems solved irrespective of the level of help used. Another metric, *the maximum difficulty level of problems solved without HLH* (MDLS-WH) was also ascertained for each user from their logs. We determined that the MDLS-WH would give an indication of how far the user progressed through the domain on their own, and thus give us a clue, albeit a vague one, of the student's expertise in the domain. We also collected the maximum difficulty level of problems attempted (MDLA) for each student. The average MDLA, MDLS, and MDLS-WH for each HLH user group was calculated and plotted, and is shown in Fig. 5.

As shown by the graph, the MDLA is always higher than the maximum difficulty of solved problems irrespective of any help levels. This is expected. The MDLS-WH increases steadily as HLH use increases, peaking at the A4 group (MDLS-WH = 4.32). After this, as HLH use increases, MDLS-WH decreases.

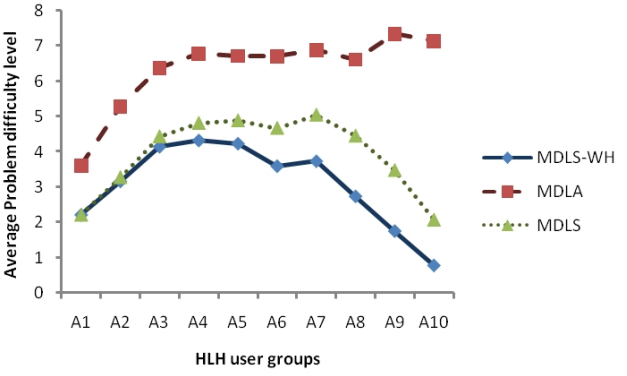


Fig. 5. The average maximum difficulty of problems attempted (MDLA), solved (MDLS), and solved without using HLH (MDLS-WH) for students in the HLH groups. Note the difference between the MDLA and MDLS-WH as HLH use increases.

The high HLH groups had very low MDLS-WH values (e.g. average MDL for A9 = 1.75) with the lowest MDLS-WH average occurring in the highest HLH user group (A10). Although the minimum problem difficulty level in SQL-Tutor is 1, the average MDLS-WH in A10 was 0.77. This was due to a number of students not solving any problems on their own, without HLH. The lowest HLH user group (A1) also reported a low MDLS-WH average (2.21). This is similar to the trend found in the learning curves above.

What is most interesting about this graph is the difference between the MDLA and MDLS-WH. The higher the HLH use, the greater the difference between the two graphs.

This shows that students who use low to moderate amounts of HLH, progress through the problem set, solving more difficult problems on their own compared to students who are either high or low frequency users of HLH. Students who are high HLH users attempt increasingly difficult problems, much harder problems on average than any other group of students. However, these students only solve very easy problems (either on their own, or using HLH), choosing to abandon problems after viewing the HLH. It is as if they have a very low expertise level, but choose to attempt problems far beyond their ZPD.

6 Future Work

Immediate future work for the authors include deeper analysis of certain groups more thoroughly (for example the composite group A1). In this paper, we divided the entire sample (evenly by HLH ratio) into ten groups. Another method would be to divide the sample into two separate groups first before beginning analysis. This first split would be dependent on the shape of the frequency distribution graph (Fig. 2), i.e. students who fall in the normal distribution of the graph ($HLH > 0.05$), and students outside the normal distribution (the group with

$HLH < 0.05$). This could be interesting as this frequency trend is common across the three separate samples that we analysed. We also would like to attempt to ascertain if students remain in one group or migrate between groups during the duration of their learning.

This research forms a part of the larger observational analysis work done on learning behaviour by various researchers. These types of analysis can then be used to form a basis to categorise students, as they work on an ITS, into various groups depending on their help-seeking behaviour. Students in each particular group can then receive customised pedagogical and intervention strategies, appropriate to the group to which they belong.

One of the challenges in creating systems that attempt to increase the effectiveness of learning is observing and comprehending the behaviour displayed by various groups of students in particular contexts. This research examines students' behaviour with regards to seeking high-level help. It forms a piece in the larger set of observations gathered by researchers, which then gives us some basis for creating customised pedagogical strategies.

References

1. Mitrović, T.: An Intelligent SQL Tutor on the Web. IJAIED 13, 173–197 (2003)
2. Aleven, V., Koedinger, K.: Limitations of Student Control: Do Students Know when They Need Help? In: Gauthier, G., VanLehn, K., Frasson, C. (eds.) ITS 2000. LNCS, vol. 1839, pp. 292–303. Springer, Heidelberg (2000)
3. Baker, R., Corbett, A., Koedinger, K., Evenson, S., Roll, I., Wagner, A., Naim, M., Raspat, J., Baker, D., Beck, J.: Adapting to When Students Game an Intelligent Tutoring System. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 392–401. Springer, Heidelberg (2006)
4. Baker, R., Corbett, A., Koedinger, K., Roll, I.: Detecting When Students Game the System, Across Tutor Subjects and Classroom Cohorts. In: Ardissono, L., Brna, P., Mitrović, A. (eds.) UM 2005. LNCS (LNAI), vol. 3538, pp. 220–224. Springer, Heidelberg (2005)
5. Koedinger, K., Mathan, S.: Distinguishing Qualitatively Different Kinds of Learning Using Log Files and Learning Curves. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 39–46. Springer, Heidelberg (2004)
6. Martin, B., Koedinger, K., Mitrovic, T., Mathan, S.: On Using Learning Curves to Evaluate ITS. In: AIED 2005, pp. 419–426. IOS Press, Amsterdam (2005)
7. Vygotsky, L.: Mind in Society: The Development of Higher Psychological Processes. Harvard University Press, Cambridge (1978)

Toward Automatic Hint Generation for Logic Proof Tutoring Using Historical Student Data

Tiffany Barnes and John Stamper

University of North Carolina at Charlotte, Computer Science Department,
9201 University City Blvd., Charlotte, NC 28223, USA
tbarnes2@uncc.edu, john@stamper.org

Abstract. We have proposed a novel application of Markov decision processes (MDPs), a reinforcement learning technique, to automatically generate hints for an intelligent tutor that learns. We demonstrate the feasibility of this approach by extracting MDPs from four semesters of student solutions in a logic proof tutor, and calculating the probability that we will be able to generate hints at any point in a given problem. Our results indicate that extracted MDPs and our proposed hint-generating functions will be able to provide hints over 80% of the time. Our results also indicate that we can provide valuable tradeoffs between hint specificity and the amount of data used to create an MDP.

1 Introduction

Logic proof construction is an important skill for both computer science and philosophy. However, in our experience, this topic is of particular difficulty for students, especially in determining a strategy to derive a conclusion from given premises. Once students apply rules that are easy for them, they often become stuck, performing unnecessary steps or giving up when the next step is unclear. In one-on-one tutoring sessions, however, suggesting one or more intermediate goal states helps many students achieve complete proofs.

Our long-term goal is to provide real-time, individualized hints to support on-going student proof construction efforts. In this paper, we present our system for automatically generating strategic hints using historical data and Markov Decision Processes, and the results of two experiments demonstrating the feasibility of automated hint generation. The first is something like a cross-validation study, comparing the hints we can generate using various semesters of data for MDP creation. The second is a simulation of creating MDPs incrementally as students work proofs, and calculating the probability of being able to generate hints as new attempts are added to the MDP.

The Proofs Tutorial is a computer-aided instructional (CAI) tool implemented on NovaNET (http://www.pearsondigital.com/novanet/). This program has been used for practice and feedback in writing proofs in our university-level discrete mathematics courses since 2002. In the tutorial, students type in consecutive lines of a proof, which consist of 4 parts: the premise, reference lines, the axiom used, and the substitutions which allow the axiom to be applied. After the student enters these 4 parts to a line, the premise, reference lines, axiom, and substitutions are verified. If a mistake is

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 373–382, 2008.

Table 1. Sample Proof 1 Solution

	Premise	Line	Reason
1.	a → b		Given
2.	c → d		Given
3.	¬ (a → d)		Given
	¬ a v d	**3**	**rule IM (error)**
4.	a ^ ¬ d	3	rule IM implication
5.	a	4	rule S simplification
	b	**4**	**rule MP (error)**
6.	b	1,5	rule MP modus ponens
7.	¬ d	4	rule S simplification
8.	¬c	2,7	rule MT modus tollens
9.	b ^ ¬c	6,8	rule CJ conjunction

made, a warning message is shown, and the line is deleted (but saved for later analysis). In this work, we examine student solutions to Proof 1, as in Table 1.

Our goal in this work is to augment the Proofs Tutorial with goal feedback that helps focus student attention on an appropriate next sub-goal. This type of feedback has been shown to improve learning and skill transfer over minimal and condition violation feedback [1]. Since the Proofs Tutorial has been used for several years, we have a large corpus of data to use in building student models from historical data. We create a student model for each problem, and use it to generate intelligent hints. As a new student works a problem, we record his or her sequence of actions as a state. If the current state is matched in the model, and the matched state has a successor closer to the goal, we enable a Hint Button to give contextual help. From the successor state with the highest reward value, we derive a hint sequence: 1) indicate a goal expression to derive, 2) indicate the rule to apply next, 3) indicate the premises (lines) where the rule can be used, and 4) a bottom-out hint combining 1-3.

Table 2 shows an example hint sequence, generated using the solution in Table 1 for a student solving proof 1 and requesting a hint after line 3. If a student's state is not found in the model, the Hint Button will be disabled. Such a student will not get goal feedback. However, we can add the student's action and its correctness to our database, and periodically run reinforcement learning to update the reward function values. Before an update is applied, we could test the update by examining new MDP states to ensure that unusual solutions have not superseded existing good solutions.

Table 2. Example hint sequence derived from example student solution

Hint #	Hint Text
1	Try to derive: a ^ ¬ d
2	Use line 3, ¬ (a → d) to derive it
3	Use the rule: IM, implication
4	Enter a ^ ¬ d with ref. line 3 and rule IM implication

2 Related Work

The problem of offering individualized help and feedback is not unique to logic proofs. Through individual adaptation, intelligent tutoring systems (ITS) can have significant effects on learning, but take considerable time to construct [2]. Constraint-based tutors, which look for violations of problem constraints, require less time to construct and can be used for problems that may not be heavily procedural [3]. However, constraint-based tutors can only provide condition violation feedback, not goal-oriented feedback, that has been shown to be more effective [1].

Example-based authoring tools such as CTAT use demonstrated examples to learn ITS production rules [4]. In these tools, teachers work problems in what they predict to be frequent correct and incorrect approaches, and then annotate the learned rules with appropriate hints and feedback. This system has also been used with data to build initial models for an ITS, in an approach called Bootstrapping Novice Data (BND) [5]. However, in both of these approaches, considerable time must still be spent in identifying student approaches and creating appropriate hints.

Machine learning has also been used to improve tutoring systems. In the ADVISOR tutor, machine learning was used to build student models that could predict the amount of time students took to solve arithmetic problems, and to adapt instruction to minimize this time while meeting teacher-set instructional goals [6]. In the Logic-ITA tutor, student data was mined to create hints that warned students when they were likely to make mistakes using their current approach [7].

Similar to the goal of BND, we seek to use student data to directly create student models for an ITS. However, instead of feeding student behavior data into CTAT to build a production rule system, our method generates Markov Decision Processes that represent all student approaches to a particular problem, and use these MDPs directly to generate hints. In [8], we used visualization tools to explore how to generate hints based on MDPs extracted from student data and verified that the rules extracted by the MDP conformed with expert-derived rules and generated buggy rules that surprised experts. In [9], we applied the technique to visualize student proof approaches to allow teachers to identify problem areas for students.

Our method of automatic hint generation using previous student data reduces the expert knowledge needed to generate intelligent, context-dependent hints and feedback. The system is capable of continued refinement as new data is provided. In this work, we demonstrate the feasibility of our hint generation approach through simulation experiments on existing student data. Although our approach is currently only appropriate for generating hints for specific problems with existing prior data, we believe that machine learning applied to MDPs may be used to create automated rules and hints for new problems in the same domain.

3 Markov Decision Processes to Create Student Models

A Markov decision process (MDP) is defined by its state set S, action set A, transition probabilities P, and a reward function R [10]. On executing action a in state s the probability of transitioning to state s' is denoted P(s' | s, a) and the expected reward associated with that transition is denoted R(s'| s, a). For a particular point in a student's proof, our method takes the current premises and the conclusion as the state,

and the student's input as the action. Therefore, each proof attempt can be seen as a graph with a sequence of states (each describing the solution up to the current point), connected by actions. Specifically, a state is represented by the list of premises generated in the student attempt, and actions are the axioms (rules) used at each step.

We combine all student solution graphs into a single graph, by taking the union of all states and actions, and mapping identical states to one another. Once this graph is constructed, it represents all of the paths students have taken in working a proof. Typically, at this step reinforcement learning is used to find an optimal solution to the MDP. For the experiments in this work, we set a large reward for the goal state (100) and penalties for incorrect states (10) and a cost for taking each action (1). Setting a non-zero cost on actions causes the MDP to penalize longer solutions (but we set this at 1/10 the cost of taking an incorrect step). We apply the value iteration reinforcement learning technique using a Bellman backup to assign reward values to all states in the MDP [10]. The equation for calculating values V(s) for each state s, where R(s) is the reward for the state, γ is the discount factor (set to 1), and $P_a(s,s')$ is the probability that action a will take state s to state s':

$$V(s) := R(s) + \gamma \max_a \sum_{s'} P_a(s,s')\, V(s')$$

For value iteration, V is calculated for each state until there is little change in the function over the entire state space. Once this is complete, the optimal solution in the MDP corresponds to taking a greedy traversal approach in the MDP [8]. The reward values for each state then indicate how close to the goal a state is, while probabilities of each transition reveal the frequency of taking a certain action in a certain state.

4 Method

We use historical data to estimate the availability of hints using different types of state-matching functions and differing datasets for training. We use data from the four fall semesters of 2003-2006 (denoted f3-f6), where an average of 220 students take the discrete math course each fall. Students in this course are typically engineering students in their 2nd or 3rd years, but most have not been exposed to a course in logic. Students attend several lectures on logic and then use the Proofs Tutorial to solve 10 proofs. Sixty percent of students used direct proof when solving proof 1. We extracted 537 of students' first attempts at direct solutions to proof 1.

The data were validated by hand, by extracting all premises generated by students, and removing those that 1) were false or unjustifiable, or 2) were of improper format. We also remove all student steps using axioms Conjunction, Double Negation, and Commutative, since students are allowed to skip these steps in the tutorial. After cleaning the data, there were 523 attempts at proof 1. Of these, 381 (73%) were complete and 142 (27%) were partial proofs, indicating that most students completed the proof. The average lengths, including errors, were 13 and 10 steps, respectively, for completed and partial proofs. When excluding errors and removed steps, the average number of lines in each student proof is 6.3 steps.

The validation process took about 2 hours for an experienced instructor, and could be automated using the existing truth and syntax-checking program in our tutorial. We

realized that on rare occasions, errors are not properly detected in the tutorial (less than 10 premises were removed). We plan to correct this in future work.

We performed two experiments to explore the capability of our method to generate automated hints. In each experiment, we isolated the data into training and test sets, where the training set was used to generate the Markov Decision Process (MDP) as described above, and the test set was used to explore hint availability. The process for comparing the test set to the MDP consists of several steps. Because of the structure of the tutorial, we first removed all error states from the MDP and from student attempts before comparison, since the tutorial provides error messages and deletes the corresponding error from the student proof. Then, each attempt in the test set is mapped onto a sequence of states. For each test state, there are two requirements for a hint to be available: 1) there must be a "matching" state in the MDP, and 2) the "matching" state must have a successor state in the MDP (i.e. it cannot be a dead end). The closer the match between a test state and the corresponding MDP state, the more context-specific the hint based on that match will be.

4.1 Matching Functions

To maximize the probability that our generated hints are in line with a student's current strategy, we seek to give hints based on states very similar to the current state. We considered four matching functions: 1) ordered (exact), 2) unordered, 3) ordered minus the latest premise, and 4) unordered minus the latest premise. An ordered, or exact, state match means that another student has taken the same sequence of steps in solving the proof. An unordered state match means that there is a state with exactly the same premises, but they were not necessarily reached in the same order. An "ordered-1" match looks for an exact match between the student's previous state and an MDP state. An "unordered-1" match looks for an unordered match between the student's previous state and an MDP state. Once a match is made, we generate a hint using the optimal successor state from the matching state. The more specific the match, the more contextualized the hint. Hints generated using unordered matches will reveal steps taken by other students in the same problem state, but who might be using a different approach to problem solving, so these hints may differ from hints based on ordered matches.

To determine hint availability, we calculated two numbers for each match type. The first is "***move matches***": the percentage of test states, or "moves", including duplicates, with matches in the MDP. The second is the "***unique matches***": the percentage of unique test states that have matches in the MDP. *Move matches* gives us a measure of the probability that a hint is available for each move. *Unique matches* reflects the percent overlap in test and training sets, and could indicate if one class is particularly different from the training set.

5 Experiment 1: Comparing Classes

In this experiment, we explored the ability of our system to provide hints using one, two, three, or four semesters of data to build MDPs. Similar to a cross-validation study, each semester is used as a test set while all the remaining semesters are used as

training sets for MDPs. This experiment provides us insight into the number of semesters of data we might need to provide hints a reasonable percentage of the time while students are solving proofs. Table 3 presents the data for each semester. Semester f5 was unusual: there were a small number of states, but a large number of moves, suggesting that students solved this proof in very similar ways.

Table 3. Semester data, including attempts, moves, and states in the MDP for each semester

Semester	# Attempts	MDP states	# Moves
f3	172	206	711
f4	154	210	622
f5	123	94	500
f6	74	133	304

We hypothesized that we could provide hints a majority of the time using just one semester as our MDP training data. Table 4 shows the percent ordered matches between each semester and the remaining combinations of training sets. We were very encouraged by these data, suggesting that our system would provide highly contextualized hints over sixty-six percent of the time, in the worst case, after just one semester of training. In all cases, adding more data increased the probability of providing hints, though we do see diminishing returns when comparing the marginal increase between 1-2 (6.8%) and 2-3 (2.8%) semesters of data.

Table 4. Average % move matches across semesters using the ordered test sets and MDPs

Test set	1-sem. MDPs	2-sem. MDPs	3-sem. MDPs
f3	68.73%	75.67%	78.62%
f4	69.77%	77.71%	81.03%
f5	86.33%	90.80%	92.00%
f6	66.34%	74.12%	77.63%
Average	**72.79%**	**79.57%**	**82.32%**

Our experiments using the remaining matching techniques showed consistent increases going from 1-semester MDPs up to 2-semester MDPs, as expected. However, the increases between 2- and 3-semester MDPs are decreasing, suggesting consistent diminishing returns for adding more data to the MDPs.

Table 5 lists the average percent matches for each of our experiments using the four match functions. This table gives an indication of the tradeoffs between using multiple semesters of data versus multiple techniques for matching. Here, we see that, on average, for 72% of moves, we can provide highly contextualized (ordered) hints using just one semester of data. With two semesters of data, we can provide these hints almost 80% of the time, but this only increases to 82% for three semesters of data. If we wished to provide hints after collecting just one semester of data, we could provide less contextualized hints for those who don't have ordered matches in

the MDP. There is a nearly identical increase in the match rate, to almost 80%, by providing hints using either unordered or ordered-1 searches. We can provide hints an additional five percent of the time if we add the unordered-1 match function.

When analyzing these data, we observed a skew in all statistics because of the unusual distribution of states and moves in f5. We therefore repeated all experiments excluding f5, and the results are given in Table 6. The differences caused by skew in f5 had a smaller effect moving from top left to bottom right, suggesting that more data or less sensitive matching can mitigate the effect of unusual training data.

Table 5. Comparison of % move matches across multiple semesters and matching techniques

Matching	1-sem. MDPs	2-sem. MDPs	3-sem. MDPs
Ordered	72.79%	79.57%	82.32%
Unordered	79.62%	85.22%	87.26%
Ordered-1	79.88%	87.84%	91.57%
Unordered-1	85.00%	91.50%	93.96%

Table 6. Comparison of % move matches, excluding f5 from all sets

Test set	1-sem. MDPs	2-sem. MDPs
Ordered	70.97%	78.05%
Unordered	78.69%	83.59%
Ord-1	79.02%	87.99%
Unord-1	85.77%	91.86%

Table 7 shows the marginal increase, with ordered as a baseline, of each matching technique for each MDP size, to illustrate the tradeoffs between additional data and matching technique. When considering matching functions, the easiest technical change is from ordered to ordered-1, where one premise is removed from the test state before comparison with the MDP states. In all cases, the probability of providing these hints is higher than that of providing hints based on unordered matches. This is probably because there is some inherent partial ordering in proofs, so only limited benefit is seen from reordering premises. When an ordered hint cannot be matched, it is perhaps more likely that the student has just performed a step that no one else has done before, rather than generating a new ordering of steps, so the benefit of ordered-1 can exceed that of unordered. Providing the unordered search requires us to maintain 2 separate MDPs to make the search more efficient, so there are both time and space tradeoffs to using unordered matching. However, adding unordered-1 after adding unordered provides a very large difference in our capability to provide hints, with little investment in time.

As part of this study we also compared the unique states across semesters, as shown in Table 8. This gives us a measure of the percent overlap between MDPs. Using 3 semesters of data with ordered matching, or using 1 semester of data with unordered-1 matching, both give us over 50% matching of states across MDPs. When compared with the much higher move matches, this suggests that although a new

Table 7. Marginal increases when comparing matching techniques to ordered

Technique	1-sem. ordered	2-sem. ordered	3-sem. ordered
Unordered	6.83%	5.65%	4.94%
Ordered-1	7.09%	8.27%	9.25%
Unordered-1	12.21%	11.93%	11.64%

Table 8. Unique state % matches across semesters and techniques

Test set	1-sem. MDPs	2-sem. MDPs	3-sem. MDPs
Ordered	34.55%	45.84%	51.93%
Unordered	43.62%	55.23%	59.90%
Ordered-1	48.25%	63.07%	71.39%
Unordered-1	57.28%	71.98%	77.87%

semester may bring many more different solution steps, the ones actually used for complete solutions already exist and are those most often used by students.

6 Experiment 2: Exploring the "Cold Start" Problem

One critique of using data to generate hints has been the expected time needed for the method to be applied to a new problem, or in other words, the "cold start" issue. Our hypothesis was that a relatively low number of attempts would be needed to build an MDP that could provide hints to a majority of students. One method for building our hint MDP would be to incrementally add MDP states as students solve proofs. This experiment explores how quickly such an MDP is able to provide hints to new students, or in other words, how long it takes to solve the cold start problem. For one trial, the method is given in Table 9.

Table 9. Method for one trial of the cold-start simulation

1. Let Test = {all 523 student attempts}
2. Randomly choose and remove the next attempt a from the Test set.
3. Add a's states and recalculate the MDP.
4. Randomly choose and remove the next attempt b from the Test set.
5. Compute the number of matches between b and MDP.
6. If Test is non-empty, then let a:=b and go to step 3. Otherwise, stop.

For this experiment, we used the ordered and unordered matching functions, and plotted the resulting average matches over 100,000 trials, as plotted in Figure 1. These graphs show a very quick rise in ability to provide hints to students, that can be fit using power functions, whether using ordered or unordered MDP states and matching. Clearly, the availability to give hints ramps up very quickly. Table 10 lists the number of attempts needed in the MDP versus target hint percentages. For the unordered

matching function, the 50% threshold is reached at just 8 student attempts and the 75% threshold at 49 attempts. For ordered matching, 50% occurs on attempt 11 and 75% on attempt 88. These data are encouraging, suggesting that instructors using our MDP hint generator could seed the data to jump-start new problems. By allowing the instructor to enter as few as 8 to 11 example solutions to a problem, the method might already be capable of automatically generating hints for 50% of student moves.

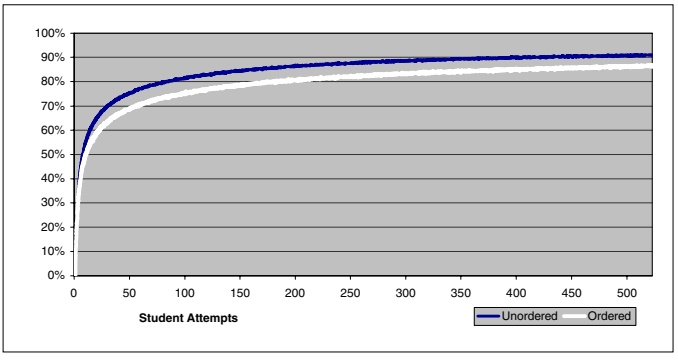


Fig. 1. Percent hints available as attempts are added to the MDP, over 100,000 trials

Table 10. Number of attempts needed to achieve threshold % hints levels

	50%	55%	60%	65%	70%	75%	80%	85%	90%
Un-Ordered	8	11	14	20	30	46	80	154	360
Ordered	11	15	22	33	55	85	162	362	?

7 Conclusions and Future Work

We have proposed and explored the feasibility of an approach to mining Markov decision processes from student work to automatically generate hints. This approach differs from prior work in authoring tutoring systems by mining actual student data, rather than relying on teachers to add examples the system can learn from. In addition, the generated hints are not created by hand as in example-based tutors, but created based on past student work. Our novel MDP-based approach enables us to automatically provide highly contextual hints, and also allow our knowledge model to learn from new student data. We note that on cold start for a new problem that has no student data, the system will still act as a problem-solving environment, but after even one semester of data is collected, a significant amount of hints can be generated.

In our future work, we plan to empirically validate our findings with actual students, and to measure the impact of our generated hints on learning. We will continue to explore ways to learn general rules to build intelligent hints, feedback and help with greater coverage and robustness. For instance, we plan to group students according to their class performance and behavior in solving proofs, and create tailored MDPs for each group of students. In [8], we proposed three modified reward

functions that may benefit different types of students. The first, used in this experiment, is the "expert" function that rewards solutions with fewer steps and errors. This is most related to existing knowledge-based help systems that emphasize efficiency, and the steps in expert solutions are probably most understandable to high-performing students. Other students may better understand less efficient approaches that have been taken by many students, inspiring our second reward function that suggests steps that many students have taken in successful past solutions. On the other hand, at-risk or highly frustrated students may benefit from hints that help them avoid complex or error-prone solutions as in our least-error prone reward function.

We plan to explore machine learning techniques to generalize our models to other problems in the same domain. We also plan to apply our MDP approach to less procedural domains, where creating goal feedback may be difficult, but providing insight into prior student solutions may help current students.

References

1. McKendree, J.: Effective Feedback Content for Tutoring Complex Skills. Human-Computer Interaction 5(4), 381–413 (1990)
2. Murray, T.: Authoring intelligent tutoring systems: An analysis of the state of the art. Intl. J. Artificial Intelligence in Education 10, 98–129 (1999)
3. Mitrovic, A., Koedinger, K., Martin, B.: A comparative analysis of cognitive tutoring and constraint-based modeling. User Modeling, 313–322 (2003)
4. Koedinger, K.R., Aleven, V., Heffernan., T., McLaren, B., Hockenberry, M.: Opening the door to non-programmers: Authoring intelligent tutor behavior by demonstration. In: 7th Intelligent Tutoring Systems Conference, Maceio, Brazil, pp. 162–173 (2004)
5. McLaren, B., Koedinger, K., Schneider, M., Harrer, A., Bollen, L.: Bootstrapping Novice Data: Semi-automated tutor authoring using student log files. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220. Springer, Heidelberg (2004)
6. Beck, J., Woolf, B.P., Beal, C.R.: ADVISOR: A Machine Learning Architecture for Intelligent Tutor Construction. In: 7th National Conference on Artificial intelligence, pp. 552–557. AAAI Press / The MIT Press (2000)
7. Merceron, A., Yacef, K.: Educational Data Mining: a Case Study. In: 12th Intl. Conf. on Artificial Intelligence in Education. IOS Press, Amsterdam (2005)
8. Barnes, T., Stamper, J.: Toward the extraction of production rules for solving logic proofs. In: Proc. 13th Intl. Conf. on Artificial Intelligence in Education, Educational Data Mining Workshop, Marina del Rey (2007)
9. Croy, M., Barnes, T., Stamper, J.: Towards an Intelligent Tutoring System for propositional proof construction. In: Brey, P., Briggle, A., Waelbers, K. (eds.) European Computing and Philosophy Conference. IOS Publishers, Amsterdam (2007)
10. Sutton, S., Barto, A.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)

Does Help Help? Introducing the Bayesian Evaluation and Assessment Methodology

Joseph E. Beck[1], Kai-min Chang[2], Jack Mostow[2], and Albert Corbett[2]

[1] Computer Science Department, Worcester Polytechnic Institute
[2] School of Computer Science, Carnegie Mellon University
joseph.beck@EducationalDataMining.org,
{mostow,kkchang,corbett}@cs.cmu.edu

Abstract. Most ITS have a means of providing assistance to the student, either on student request or when the tutor determines it would be effective. Presumably, such assistance is included by the ITS designers since they feel it benefits the students. However, whether—and how—help helps students has not been a well studied problem in the ITS community. In this paper we present three approaches for evaluating the efficacy of the Reading Tutor's help: creating experimental trials from data, learning decomposition, and Bayesian Evaluation and Assessment, an approach that uses dynamic Bayesian networks. We have found that experimental trials and learning decomposition both find a negative benefit for help--that is, help hurts! However, the Bayesian Evaluation and Assessment framework finds that help both promotes student long-term learning and provides additional scaffolding on the current problem. We discuss why these approaches give divergent results, and suggest that the Bayesian Evaluation and Assessment framework is the strongest of the three. In addition to introducing Bayesian Evaluation and Assessment, a method for simultaneously assessing students and evaluating tutorial interventions, this paper describes how help can both scaffold the current problem attempt as well as teach the student knowledge that will transfer to later problems.

Keywords: educational data mining, dynamic Bayesian networks, assessment, evaluation, Bayesian Evaluation and Assessment.

1 Introduction

An important property of an Intelligent Tutoring System (ITS) is its ability to help students. Thus, measuring the effectiveness of tutor help is an important evaluation of an ITS. Does help help? Does one type of help work better than the others [1]? Even though the tentative answer is "yes" by most ITS researchers (otherwise, why include help at all in the tutor?), answering such questions is surprisingly difficult. One of the difficulties is that the question "does help help?" is ill-defined; what does it mean to help students? Does it mean to assist students in performing correctly on the current attempt or does it mean to assist in learning of persistent knowledge that will help on future attempts?

To measure the effectiveness of tutor help, we would ideally set up a controlled pre- and post- test experiment. A typical experimental setup works as follows: in the

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 383–394, 2008.

pre-test, we assess student performance before using the ITS. Then, we randomly assign students into two groups. The experimental group uses one version of ITS *with* the tutor help that we're evaluating, whereas the control group uses another version of ITS *without* the particular tutor help. After students use the ITS for some time, we assess student performance of the two groups again in the post-test. We test the hypothesis that the performance improvement in the experimental group is significantly different than in the control group. This experimental design is sound and has been extensively practiced in the field of psychology. Nonetheless, the experimental design is often impractical for evaluating an ITS because a controlled experiment takes a long time to conduct and is often too expensive to conduct, although exceptions exist [2].

Given that the ideal pre- and post-test experimental studies are often impractical, there are several other approaches to measure the effectiveness of tutor help. For example, we may conduct user case studies and directly ask the students whether they find the tutor help effective. Unfortunately, while user case studies provide valuable qualitative feedback, they lack the ability to draw conclusive relationships. Alternatively, we can try to infer tutor help efficacy *from data*. For instance, one might claim that tutor help is effective if student performance improves when they receive help, compared to when they do not receive help. However, this approach raises the question of *when* to assess student performance. Immediate performance is prone to scaffolding effects where tutor help merely provides a short-term performance boost. For example, some help types provide students the answer; if students simply imitate the answer we should not count that as learning.

In this paper, we describe a methodology to model both tutor help and student knowledge in one coherent framework. This configuration allows us to tease apart the effect of help into 1) scaffolding immediate performance and 2) teaching persistent knowledge that improves long term performance. We evaluate the proposed framework on student performance data from the Reading Tutor, an Intelligent Tutoring System that listens to children read aloud [3]. The Reading Tutor uses automated speech recognition to listen to children read aloud and tries to score their reading as correct or incorrect. Students can ask for assistance on a challenging word, and the Reading Tutor chooses randomly which type of help to give. For example, if the student clicks on the word "cat" the tutor could say "Rhymes with…bat"; it could sound out the word, break longer words into syllables; or simply speak the word for the student. If the student does not like the help provided, he can click again and receive another random selection.

2 Naïve Approaches to Modeling Help

There are several approaches one could apply to observational data to estimate the efficacy of the tutor's help. We first discuss experimental trials and learning decomposition.

2.1 Experimental Trials

For experimental trials, two items are needed: what is being compared, and the outcome measure with which to perform the comparison. Note that all of the scoring in this paper is performed by automated speech recognition (ASR), which is not nearly as accurate as typed input. Therefore, interpreting a number in isolation or numbers

derived from a small number of observations is suspect. In terms of what is being compared, the issue is somewhat problematic. One natural item of interest is student performance on words on which he clicked for help. One possible comparison is words on which he did not ask for help.

In terms of an outcome measure, student performance on the word is a natural measure to use. Since students are reading stories aloud to the Reading Tutor, we expect students to periodically encounter words simply in the course of reading, and we can use those as our outcome. If we use student performance at reading the word immediately after receiving help, we find that words on which the student receives help are read more accurately than words on which he does not. However, this outcome is contaminated by recency effects. For example, if tutor read *antidisestablishmentarianism* aloud, and the student immediately mimicked the tutor, we should not necessarily be confident that he actually knows the word. Perhaps the pronunciation was simply in his working memory buffer [4].

A stronger outcome is one that avoids memory effects by waiting for a later day to test the student's performance. If we change the outcome to only consider cases where the student encounters the word on a later day, then we find that words that did not receive help were read with 83% accuracy, while words that received help were read with 73% accuracy. In other words, help is providing a "benefit" to students of 10% worse performance. Although the Reading Tutor's help could certainly be improved, we are skeptical that its assistance is *that* bad. A more likely explanation is that students click for help on words they do not know. If a student doesn't know a word, the help might help him to learn it, but even after receiving the help he probably will not understand the word as well as one that he already knew. Therefore, the difference in performance on later days is more a function of the student's starting knowledge than a function of receiving the help.

2.2 Learning Decomposition

A slightly more sophisticated technique is learning decomposition [5-7]. Learning decomposition is a variant of learning curves. Typically learning curves estimate how much students learn as a result of a practice opportunity. Learning decomposition instead estimates the relative worth of different types of learning opportunities. For example, prior work with learning decomposition has shown that students learn approximately 25% more in stories they choose to read vs. those selected by the tutor [8]. It is possible to apply learning decomposition to this analysis by considering two types of learning encounters: reading words and receiving help. In this way, we can see how valuable help is compared to simply reading the word.

Unlike the experimental trials approach, it is not necessary to construct a comparison set of words. Learning decomposition simply computes the relative impact of help compared to reading the word. Similar to the experimental trials approach, it is necessary to decide what the set of allowable outcomes will be. Again, to avoid recency effects, we only consider words that students encounter on later days. We fit the model shown in Equation 1 to each student's data. By doing so, we get an estimate of the impact of help for each student, controlling for the fixed traits of the student (this control is analogous to that from having the student be a factor in logistic regression).

$$readingtime = A * e^{-b*(r*m*RM + r*RD + m*NM + ND + h*\#helps)}$$

Equation 1. Learning decomposition formula for evaluating the impact of help

The learning decomposition model finds that reading a word is, by definition, worth 1.0 practice opportunities. Relative to this baseline, depending on the exact model used, help is worth roughly -1.5 to -4 trials of learning (e.g. the model reported in [5] produces a result of -1.91, while the model shown in Equation 1 gives a result of -3.3 exposures). That is, receiving help caused students to perform worse on later trials compared to words on which they did not receive help. Even after controlling for student properties by constructing a per-student model, and comparing the effect of help relative to a baseline of simply reading a word, help is still found to be unhelpful. We suspect a similar effect is occurring is with the experimental trials approach: students request help on words on which they have low knowledge. The help thus acts as *evidence* of a lack of knowledge, rather than a direct *cause* of that lack of knowledge.

3 New Approach: Bayesian Evaluation and Assessment

There are two primary failures with the above-mentioned naïve approaches to modeling the impact of help. First, controlling for students is not sufficient. Rather, it is necessary to control for the student's knowledge of the skill (in our case, word) being helped. Second, it is necessary to refine exactly what is meant by help helping the student. Although both of the prior analyses ignored the short-term impact of help on performance, that may not be the best approach. Students typically request help when they are stuck; if help can get them unstuck then it can be said to be at least partially effective. Such a temporary boost is of course no substitute for truly effective help, otherwise ITS designers would have help systems that simply told students the answer. However, there should at least be some acknowledgement of short term benefits.

To achieve these goals, we unify two common analysis goals in ITS. The first of these is *assessing* student knowledge. Most ITS have some form of assessment or student modeling. Fig. 1 shows a graphical representation of knowledge tracing [8], a fairly common student modeling approach. This relatively simple dynamic Bayesian network suffices to completely describe knowledge tracing [9]. The shaded nodes represent things the model can directly observe, in this case student performance. The unshaded nodes represent unobservable latent parameters, in this case the student's knowledge. Each pair of knowledge and performance represents one practice opportunity for a particular skill. So in this example there are two practice opportunities represented.

The arrow from student knowledge to student performance indicates that knowledge influences performance. Performance is assumed to be a noisy reflection of knowledge, and is mediated by two parameters. The *guess* parameter represents that a student may sometimes generate a correct response in spite of not knowing the correct skill. The *slip* parameter acknowledges that even students who understand the skill can make an occasional careless mistake. The definition of each of the parameters in Fig. 1 is shown in Equation 2. The link between student knowledge across time slices indicates that students maintain and hopefully increase their knowledge across time slices. Although both learning and forgetting can occur in the real world, we follow standard practice for knowledge tracing and set the forgetting parameter to be 0.

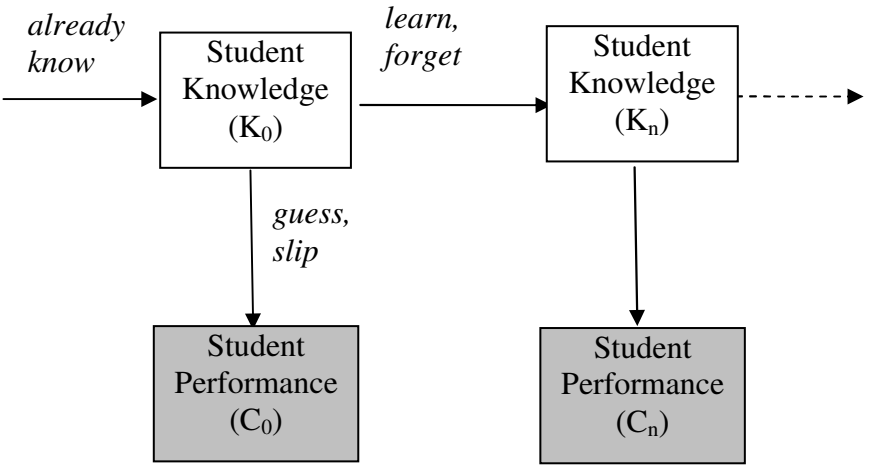


Fig. 1. Diagram of knowledge tracing

$$
\begin{aligned}
\textit{already know} &\equiv \Pr(K_0 = \textit{true}) \\
\textit{learn} &\equiv \Pr(K_n = \textit{true} \mid K_{n-1} = \textit{false}) \\
\textit{forget} &\equiv \Pr(K_n = \textit{false} \mid K_{n-1} = \textit{true}) = 0 \\
\textit{guess} &\equiv \Pr(C_n = \textit{true} \mid K_n = \textit{false}) \\
\textit{slip} &\equiv \Pr(C_n = \textit{false} \mid K_n = \textit{true})
\end{aligned}
$$

Equation 2. Equations representing knowledge tracing parameters

The second common analysis goal in ITS is to evaluate tutorial interventions. Fig. 2 shows graphically how such evaluations are frequently performed [e.g. 1, 10]. In this case, both the student performance and the intervention are observable. The approach is to determine how the intervention influences student performance. Since both nodes are observable, this task is typically easier than student modeling. Our approach is to combine these two methodologies, assessment and evaluation, into a single modeling framework, shown in Fig. 3. The Student Knowledge and Student Performance nodes are similar to the ones in knowledge tracing. The new node represents a binary variable: did the tutorial intervention we are evaluating occur during this practice opportunity? This node creates two new arcs in the network. The first one, *teach*, connects the tutorial intervention with student knowledge. It models the impact the tutorial intervention could have on the student's knowledge. Note that this arc carries forward to future time slices, so the impact on the student will persist. The second new arc, *scaffold [11]*, represents the impact the intervention has on the current practice opportunity. This temporary support does not persist across problems, and only serves to aid the student on the current problem.

This approach simultaneously assesses the student, since performance and knowledge are linked, and evaluates the tutorial intervention both in the context of its

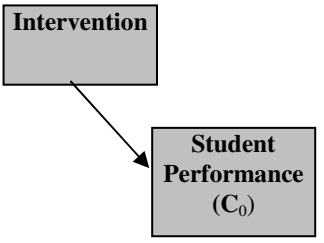


Fig. 2. Common approach for evaluating tutorial interventions

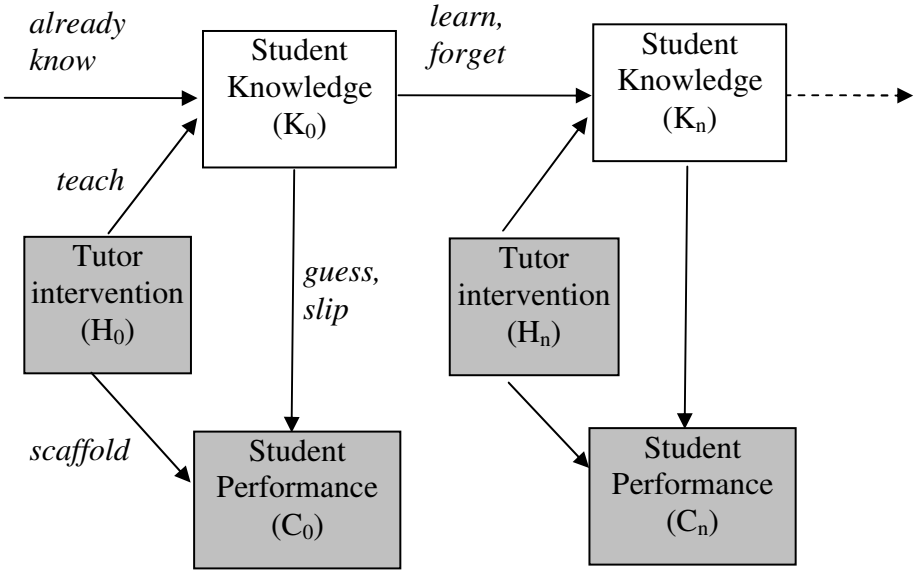


Fig. 3. Bayesian Evaluation and Assessment architecture

temporary benefit to student performance and its lasting impact on student knowledge. The impact of the tutorial intervention can be determined by examining the parameters learned by the model. For example P(learn | tutor intervention = false) is the baseline probability that a student will acquire a skill simply by practicing it. P(learn | tutor intervention = true) is the probability the student will acquire the skill as a result of receiving both the intervention and a chance to practice the skill. Comparing these two parameters permits us to estimate how much learning the intervention causes. Similarly, the scaffolding effect on student performance can be estimated by comparing P(correct response | student didn't know the skill, intervention=false) vs. P(correct response | didn't know the skill, intervention=true). Any difference in performance is the scaffolding effect of the intervention. Equation 3 shows all of the equations for the Bayesian Evaluation and Assessment model.

$$already\ know \mid help \equiv \Pr(K_0 = true, \mid H_0 = true)$$
$$already\ know \mid no\ help \equiv \Pr(K_0 = true, \mid H_0 = false)$$

$$learn \mid help \equiv \Pr(K_n = true \mid K_{n-1} = false, H_n = true)$$
$$learn \mid no\ help \equiv \Pr(K_n = true \mid K_{n-1} = false, H_n = false)$$

$$forget \mid help \equiv \Pr(K_n = false \mid K_{n-1} = true, H_n = true)$$
$$forget \mid no\ help \equiv \Pr(K_n = false \mid K_{n-1} = true, H_n = false)$$

$$guess \mid help \equiv \Pr(C_n = true \mid K_n = false, H_n = true)$$
$$guess \mid no\ help \equiv \Pr(C_n = true \mid K_n = false, H_n = false)$$

$$slip \mid help \equiv \Pr(C_n = false \mid K_n = true, H_n = true)$$
$$slip \mid no\ help \equiv \Pr(C_n = false \mid K_n = true, H_n = false)$$

Equation 3. Equations for parameters in Bayesian Evaluation and Assessment model

4 Results

Our data came from 360 children between six and eight years old who used Project LISTEN's Reading Tutor [3] in the 2002-2003 school year. On average, students used the tutor for 8.5 hours. Over the course of the school year, these students read approximately 1.95 million words (as heard by the automatic speech recognizer). We separated the data into training and testing sets by splitting the students into two groups. We sorted the students according to their amount of Reading Tutor usage and then alternately assigned students to the two sets.

During a session with the Reading Tutor, the tutor presented one sentence (or fragment) at a time for the student to read aloud. The student's speech was segmented into utterances delimited by silences. Each utterance was processed by the Automatic Speech Recognizer (ASR) and aligned against the sentence. This alignment scored each word of the sentence as either accepted (classified by the ASR as read correctly) or rejected (thought to be misread or omitted). ASR acceptance is modeled as the observed performance (C_n). The tutorial intervention node is instantiated by whether the student received help on a word. For modeling purposes, this paper treats each English word as a separate skill.

We make use of a generic Bayes net toolkit for student modeling, BNT-SM [12], for our experiments. BNT-SM inputs a data set and a compact XML specification of a DBN model hypothesized by a researcher to describe causal relationships among student knowledge and observed behavior. It generates and executes the code to train and test the model using the Bayes Net Toolbox [13]. BNT-SM allows researchers to easily explore different hypotheses on how is knowledge represented in a student

model. We now show how to use BNT-SM to construct a DBN that models the effectiveness of tutor help on student knowledge.

We used the BNT-SM and Expectation Maximization (EM) algorithm to optimize data likelihood (i.e. the probability of observing our student performance data) in order to estimate our model's parameters. EM is the standard algorithm used in the machine learning community to estimate DBN parameters when the structure is known and there exist latent variables (e.g., student knowledge, K_n). EM is guaranteed to converge to a local maximum on the likelihood surface. We used the junction tree procedure for exact inference (estimating the value of the hidden variables). See Jensen [14] for a thorough introduction to Bayes nets and the standard training and inference algorithms.

4.1 Results for Evaluating Help

To evaluate the effectiveness of tutor help, we first compare the model parameters estimated by our Bayesian Evaluation and Assessment model with those obtained by estimating a simpler knowledge tracing model.

Table 1 shows the parameters estimated for the KT model and the Bayesian Evaluation and Assessment (Help, for short) model, respectively. Notice that the KT model does not consider the help information, whereas the Help model has the parameters conditioned on whether or not tutor help is given or not. As seen in the Help model of Table 1, the probability of *already know* (i.e. does the student know the word when first starting to use the tutor) is much higher when there is no help than when there is help. This suggests that tutor help is more likely to be provided for words the student is less likely to know– a positive finding. Also, the probability of *learning* is higher when there is help than when there is no help. Even though the effect of help is only an 8% relative improvement, it is at least in the right direction (unlike the two baseline approaches), suggesting that tutor help does have a positive effect on long term knowledge acquisition.

Table 1. Comparing the parameters estimated by the KT model and the Help model

	KT model	Help model	
		No Help Given	Help Given
Already know	0.618	0.660	0.278
Learn	0.077	0.083	0.088
Guess	0.689	0.655	0.944
Slip	0.056	0.058	0.009

Also as seen in Table 1, the probability of guess is higher when there is help than when there is no help and the probability of slip is higher when there is no help than when there is help. In other words, even if the student does not know the skill he is much more likely to generate a correct response when he receives than when he does not: 94% vs. 66% (the guess rate is inflated when applying knowledge tracing to student models that use speech recognition for scoring [15]). This finding suggests that tutor help does have a scaffolding effect on assisting immediate performance. Notice

that, although we have argued that teaching effect is more beneficial in the long run than the scaffolding effect, we cannot ignore the latter. For instance, if a student is stuck when using the tutor, the tutor should still help the student to become unstuck.

Finally, both the teaching and scaffolding effects are statistically reliable at $p<0.05$ (paired samples t-test, done per-skill to avoid intraskill correlation), suggesting that tutor help does have an effect on both student knowledge and student performance.

4.2 Results for Modeling Students

Although the goal of this paper is to determine whether and how help helps students, our model also estimates student knowledge as a side effect. Therefore, we evaluate its performance at doing so. Since student knowledge is a latent variable that cannot be directly observed, we have no gold standard against which to compare. Instead, we used the trained student model to predict whether the ASR would accept or reject a student's next attempt to read the word. That is, we observe reading item by item and predict whether the next word will be read correctly (in not yet seen test data). An ROC (Receiver Operating Characteristic) curve measures the performance of a binary classifier by plotting the true positive rate against the false positive rate for varying decision thresholds. The area under the ROC curve (AUC) is a reasonable performance metric for classifier systems, assuming no knowledge of the true ratio of misclassification costs [16].

On our training data, the new model had near identical performance to classic knowledge tracing: AUC of 0.654 vs. 0.652. On the held out test data, performance was again a tie: AUC of 0.612 vs 0.615. It is disappointing our approach of simultaneously assessing students and evaluating the tutor did not yield more accurate assessment.

5 Contributions

This paper makes three main contributions to the ITS literature. First, the Bayesian Evaluation and Assessment framework unifies several strands of research. It is based on knowledge tracing [8] for assessing students. There has also been work on creating a node to measure the impact of help and connecting it to student knowledge [17]. However, this work used a simplified version of knowledge tracing, and was never evaluated with actual student data. The third strand is the ANDES system [18], which has a link between student knowledge and performance—that is, it assumes that help provides a scaffolding effect—but not between help and knowledge. Furthermore, the parameters in the ANDES system were not estimated from data.

The second contribution this paper makes is the conceptual one of simultaneously representing tutor interventions and the student's knowledge. Previous approaches addressed these problems separately by ignoring one to solve the other [1,3]. Specifically, KT ignored help, and some other experiments [3] ignored student knowledge, or how it changed over time.

The third contribution this paper makes is on distinguishing between two effects of help: scaffolding immediate performance vs. boosting actual learning. Prior work either assumed help has no direct impact on student learning [18] or that help has no direct impact on student performance [17]. Moreover, because we model tutor help

and student knowledge in one coherent framework, we can estimate the scaffolding and teaching effects. This separation of immediate vs. persistent effect of help allows researchers to understand what the tutor intervention is really doing. For instance, it is possible to investigate whether some tutor interventions help persistent learning while others mainly help immediate performance.

6 Future Work

Currently, due to limitations in BNT-SM, we could only test models with discrete, binary variables. For example, in the Help model, we only answer the question "*does help help at all*?" A more interesting question to ask is "*which type of help helps more, and when is it effective?*" Thus, a future study is to extend BNT-SM to handle multinomial variables, which allows modeling of different help types.

One question that we are interested in exploring is how does our dynamic Bayes net framework compare to the pre- and post- test experimental design [2]. Do they draw similar conclusions, despite the fact that an experimental design is usually more expensive to conduct than data fitting with DBNs? Moreover, what kinds of causal relationship can we infer with our Bayesian framework? That we were able to get a positive result with a non-randomized intervention, whether a student receives help, suggests the framework should perform well at analyzing actual randomized controlled trials, and may even be a more sensitive measure due to its accounting for student knowledge.

Another issue that we are interested in addressing is to better understand why we cannot better model students in our new framework. One possible explanation is there are too many parameters. The impact of help is modeled independently for all 3000 skills (words) in the domain. Some way of simplifying the parameter search by using hierarachical models or Dirichlet priors [19] may be a solution.

Finally, we would like to conclude that our Bayesian Evaluation and Assessment architecture is the most accurate of the three approaches proposed in this paper. On balance, given that the other two approaches found that help is harmful, we can tentatively conclude that the new approach is better. However, better clarifying which approach is most accurate and when would be helpful. This question cannot be answered for interventions whose true effectiveness is unknown (i.e. all ITS interventions that exist in the real world). Therefore, evaluations with synthetic data [e.g. 17] are a promising route forward.

7 Conclusions

This paper presents a new approach, Bayesian Evaluation and Assessment, that we used to measure the impact of help. Of the three approaches we considered, our new framework gave the only plausible answer to the question "does help help?" Although the result is equivocal, as we do not know the "real" answer, it is important to note that this drawback is fundamental to **every** method of measuring an intervention's effectiveness. Typically when a number is presented purporting to represent how well an intervention worked, there is no discussion of alternate methods of doing the measuring. By putting the three numbers forward we acknowledge the problem,

and argue that only one of the numbers is plausibly correct. The Reading Tutor's on-demand help is potentially useless, and we would not disregard the possibility, but the negative impacts claimed by the other two approaches are simply implausible.

The reason our new framework is superior is that it controls for student knowledge while estimating the intervention's effectiveness. Conceptually, this simultaneous modeling is similar to item response theory [20], which enables better comparisons of students across groups by simultaneously estimating student proficiency and question difficulty.

Finally, we feel it is important to enumerate both impacts of assistance: short term performance boosts (scaffolding) as well as longer term learning gains (teaching). By simultaneously addressing all of these aspects of assessment and evaluation, this framework represents a step forward in ITS evaluation methodology.

Acknowledgements

This work was supported by the National Science Foundation, ITR/IERI Grant No. REC-0326153. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation

References

1. Heiner, C., Beck, J.E., Mostow, J.: Improving the help selection policy in a Reading Tutor that listens. In: Proceedings of the InSTIL/ICALL Symposium on NLP and Speech Technologies in Advanced Language Learning Systems, Venice, Italy, pp. 195–198 (2004)
2. Arroyo, I., Beck, J.E., Beal, C.R., Wing, R.E., Woolf, B.P.: Analyzing students' response to help provision in an elementary mathematics Intelligent Tutoring System in Help Provision and Help Seeking in Interactive Learning Environments. In: Workshop at the Tenth International Conference on Artificial Intelligence in Education, San Antonio (2001)
3. Mostow, J., Aist, G.: Evaluating tutors that listen: An overview of Project LISTEN. In: Feltovich, P. (ed.) Smart Machines in Education, pp. 169–234. MIT/AAAI Press, Menlo Park (2001)
4. Anderson, J.R.: Rules of the mind. Lawrence Erlbaum Associates, Hillsdale (1993)
5. Beck, J.: Using learning decomposition to analyze student fluency development. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053. Springer, Heidelberg (2006)
6. Zhang, X., Mostow, J., Beck, J.E.: All in the (word) family: Using learning decomposition to estimate transfer between skills in a Reading Tutor that listens. In: AIED 2007 Educational Data Mining Workshop, pp. 80–87 (2007)
7. Beck, J.E.: Does learner control affect learning? In: Proceedings of the 13th International Conference on Artificial Intelligence in Education, Los Angeles, pp. 135–142 (2007)
8. Corbett, A.T., Anderson, J.R.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User Modeling and User-Adapted Interaction 4, 253–278 (1995)
9. Reye, J.: Student Modelling based on Belief Networks. International Journal of Artificial Intelligence in Education 14, 1–33 (2004)

10. Mostow, J., Beck, J., Bey, J., Cuneo, A., Sison, J., Tobin, B., Valeri, J.: Using automated questions to assess reading comprehension, vocabulary, and effects of tutorial interventions. Technology, Instruction, Cognition and Learning 2, 97–134 (2004)
11. Vygotsky, L.: Play and its role in the mental development of the child. In: Bruner, J., Jolly, A., Sylva, K. (eds.) Play: Its role in development and evolution (1976), pp. 461–463. Penguin Books, New York (1933)
12. Chang, K.-m.K., Beck, J.E., Mostow, J., Corbett, A.: A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems. In: 8th International Conference on Intelligent Tutoring Systems, Jhongli, Taiwan (2006)
13. Murphy, K.: Bayes Net Toolbox for Matlab (1998)
14. Jensen, F.V., Jordan, M., Lauritzen, S.L., Lawless, J.F., Nair, V. (eds.): Bayesian Networks and Decision Graphs. Statistics for Engineering and Information Science. Springer (2001)
15. Beck, J.E., Sison, J.: Using knowledge tracing in a noisy environment to measure student reading proficiencies. International Journal of Artificial Intelligence in Education 16, 129–143 (2006)
16. Hand, D., Mannila, H., Smyth, P.: Principles of Data Mining. MIT Press, Cambridge (2001)
17. Jonsson, A., Johns, J., Mehranian, H., et al.: Evaluating the Feasibility of Learning Student Models from Data. In: Educational Data Mining: Papers from the AAAI Workshop, pp. 1–6. AAAI Press, Pittsburgh (2005)
18. Conati, C., Gertner, A., VanLehn, K.: Using Bayesian Networks to Manage Uncertainty in Student Modeling. User Modeling and User-Adapted Interaction 12(4), 371–417 (2002)
19. Beck, J.E., Chang, K.-m.: Identifiability: A Fundamental Problem of Student Modeling. In: International Conference on User Modeling, Corfu, Greece, pp. 137–146 (2007)
20. Embretson, S.E., Reise, S.P.: Item Response Theory for Psychologists. In: Harlow, L.L. (ed.) Multivariate Applications, p. 371. Lawrence Erlbaum Associates, Mahwah (2000)

Using Knowledge Discovery Techniques to Support Tutoring in an Ill-Defined Domain

Roger Nkambou[1], Engelbert Mephu Nguifo[2], and Philippe Fournier-Viger[1]

[1] Université du Québec à Montréal, Laboratoire GDAC, Montréal (QC), Canada
[2] Université Lille-Nord de France, Artois, F-62307 Lens, CRIL, F-62307 Lens
CNRS UMR 8188, F-62307 Lens , France
nkambou.roger@uqam.ca, mephu@cril.univ-artois.fr,
philippe.fv@gmail.com

Abstract. Domain experts should provide relevant knowledge to a tutoring system so that it can guide a learner during problem-solving learning activities. However, for ill-defined domains this knowledge is hard to define explicitly. As an alternative, this paper presents a framework to learn relevant knowledge related to procedural tasks from users' solutions in an ill-defined procedural domain. The proposed framework is based on a combination of sequential pattern mining and association rules discovery. The resulting knowledge base allows the tutoring system to guide learners in problem-solving situations. Preliminary experiments have been conducted in CanadarmTutor.

1 Introduction

An ill-structured problem is defined by Simon [15] as one that is complex, with indefinite starting points, multiple and arguable solutions, or unclear strategies for finding solutions [17]. Domains that include such problems and in which, tutoring targets the development of problem-solving skills are said to be ill-defined (within the meaning of Ashley et al. [16]). According to Aleven, Ashley, Lynch and Pinkwart [1], ill-defined domains present a number of unique challenges for researchers in Intelligent Tutoring Systems and Computer Modeling, such as "(1) defining a viable computational model for aspects of underspecified or open-ended domains; (2) developing feasible strategies for search and inference in such domains; (3) providing feedback when the problem-solving model is not definitive; (4) structuring of learning experiences in the absence of a clear problem, strategy, and answer; (5) user models that accommodate the uncertainty of ill-defined domains; and 6) user interface design for ITSs in ill-defined domains where usually the learner needs to be creative in his actions, but the system still has to be able to analyze them".

The method of cognitive task analysis that aims at producing effective problem spaces or task models by observing expert and novice users is a good solution for capturing different ways of solving problems. This supports model and knowledge tracing, coaching, errors detection, and plan recognition. However, this process is very time-consuming [2] and cannot be applied easily for ill-defined domains. Constraint based modeling (CBM) was proposed as an alternative [3]. It consists of specifying sets of constraints on what is a correct behavior, instead of providing a

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 395–405, 2008.

complete task description. Though this approach was shown to be effective for some ill-defined domain, a domain expert has to design and select the constraints carefully.

Alternatively, our proposal aims at learning a knowledge base that can replace (or represent) the problem space, from users' interactions. It combines two knowledge discovery techniques. Sequential patterns mining is used to discover frequent actions sequences among the recorded usage of expert, intermediate and novice users. Association rules discovery finds associations between these significant actions sequences, relating them together. The goal is to use solutions from users as a source of knowledge about the ill-defined problem. In this paper, we show how the proposed framework is used to discover new domain knowledge that the tutor uses to track learners' actions and provides them with relevant hints. The framework is applied in the CanadarmTutor [4], a simulation-based tutoring system that we have developed to teach astronauts how to operate a robot manipulator deployed on the International Space Station (ISS). During the robot manipulation, operators do not have a direct view of the scene of operation on the ISS and must rely on cameras mounted on the manipulator and at strategic places in the environment where it operates. Furthermore, for a given robot manipulation problem, there are many possibilities for moving the robot to a goal position and thus, it is not possible to define a complete and explicit task model. In fact there is no simple 'legal move generator' for finding all the possibilities at each step. Hence, CanadarmTutor operates in an ill-defined-domain [15].

The paper is organized as follows. First, we present some related works and their limitations. Second, we describe the framework and some techniques that are used. We then present a few possible tutoring services based on the framework. Finally, we describe an experiment of using the framework in CanadarmTutor and illustrate how the results enable CanadarmTutor to provide more realistic tutoring services.

2 Related Works

Creating cognitive tutors usually rests on the implicit assumption that one should predefine a task model describing correct and incorrect solution paths. CTAT [2] offers a set of tools that allows ITS designers to specify the behavioral graph (BG) of a task, presenting correct and buggy paths. BGs (sometimes transformed into production rules) are used to track student actions. The behavior recorder can automate the translation of user actions into a BG. This concept was improved by the BND (Bootstrapping Novice Data) approach [5]. BND records the actions of many students in a log file which is then used to create a common BG that can be improved by designers. However, the BND approach is devoid of learning, reducing the approach to a simple way of storing or integrating raw user solutions into a structure, as in [6] and [7]. This is very limiting because the system does not try to extract useful knowledge from those solutions, which could enrich the problem space.

In CanadarmTutor, to automatically detect errors of a student learning to operate the manipulator and to produce illustrations of correct and incorrect motions in training and give feedback to the learner, our first solution was to integrate a special path-planner based on probabilistic roadmap approach into the system. The path-planner we developed ([4]) acts as a domain expert and can calculate the arm's moves avoiding obstacles and consistent with the best available cameras views to achieve a

given goal. The path-planner enables CanadarmTutor to answer several learners' questions such as: *How to...*, *What if…*, *What next…, Why…* and *Why not*. However, solution paths provided by the path-planner are sometimes too complex and difficult to follow by the users. To sustain more effective learning, an effective problem space that captures real users' knowledge is needed. We decided to create a partial effective solution using the cognitive tutor approach [8]. We modeled the spatial knowledge for the Canadarm manipulation task as semantic knowledge. To achieve this, we discretized the 3D space into 3D sub spaces named elementary spaces (ES). Spatial knowledge is encoded as relationships such as (1) a camera can see an ES or an ISS module, (2) an ES comprise an ISS module, (3) an ES is next to another ES or (4) a camera is attached to an ISS module. The procedural knowledge of how to move the arm to a goal position is modeled as a loop where the learner must recall a set of cameras for viewing the ESs containing the arm, select the cameras, adjust their parameters, retrieves a sequence of ESs to go from the current ES to the goal, and then move to the next ES. CanadarmTutor detects all the atomic actions like camera changes and entering/leaving an ES. It was not possible to go into finer details like how to choose the joint(s) to move from an ES to another. As [9] stated, modeling complete possible solutions for a given goal is not realistic in ill-defined domain. That is exactly the case in CanadarmTutor.

The CBM approach may represent a good alternative to the cognitive tutor approach [3, 9]. However, in the CanadarmTutor context, it would be a difficult work for domain experts to describe relevance and satisfaction conditions. In fact, there would be too many conditions and many possible ideal solutions for each problem; the domain is too much complex for this approach.

Contrary to these approaches, we are proposing a solution to create a more general, flexible, albeit sometimes partial, BG-like structure by inferring association rules between actions or action sequences, providing meta-knowledge to the ITS. In fact, both novice and expert domain users can provide primitive action sequences required to achieve typical tasks in the application domain. These sequences, whether good or buggy, may then be used to teach procedural knowledge associated with the task, thereby continually enhancing the system's intelligence. They can be used for supporting valuable tutoring services without the need of a clear and complete representation of problem space. We believe that, this approach represents an effective and complementary solution for tutoring ill-defined domain.

3 The Framework

The framework that we propose goes through four processes to learn rules: 1) Given log files containing users' plans, the first step consists in generating frequent sequential patterns; 2) Those patterns are used for creating a meta-context where each plan is linked with the frequent patterns that appear in it; 3) Using the meta-context, a generic base of association rules is produced; it shows how frequent sequential patterns are related; 4) The generic base and the set of frequent patterns are transformed into a knowledge base that will be used by the tutoring system.

3.1 Finding Frequent Sequential Patterns Using PrefixSpan

The problem of mining sequential patterns is stated as follows [10]. Let D be a transactional database containing a set of transactions (here also called plans) and a set of sequence of items (called actions in our context). An example of D is depicted in table 1. Let A = {a_1, a_2,…, a_n} be a set of actions. We call a subset X $\subseteq$ A an *actionset* and |X|, its size. A sequence s = (s_1, s_2, … , s_m) is an ordered list of actionsets, where $s_i \subseteq$ A, i $\in$ {1,…,m}, and where m is the the size of s (also noted |s|). A sequence s_a = (a_1, a_2,…, a_n) is contained in another sequence s_b = (b_1, b_2,…, b_m) if there exists integers $1 \leq i_1 < i_2 < \ldots < i_n \leq m$ such that $a_1 \subseteq bi_1$, $a_2 \subseteq bi_2$, . . . , $a_n \subseteq bi_n$. The relative support of a sequence s_a is defined as the percentage of sequences s $\in$ D that contains s_a, and is denoted by *supD*(s_a). The problem of mining sequential patterns is to find all the sequences s_a such that *supD*(s_a) $\geq$ *minsup* for a database D, given a support threshold *minsup*.

Consider the dataset of table 1. The size of the plan P2 is 6. Suppose we want to find the support of the sequence s_a = (1 {9 31}). From Table 1, we know that s_a is contained in the sequences for plan 1 and plan 3 but is not in the sequence for plan 2. Hence, the support of s_a is 2 (out of a possible 7), or 0.28. If the user-defined minimum support value is less than 0.28, then s_a is deemed frequent.

Table 1. A Data Set of 7 Successful plans

PlanID	Sequences of actions
P1	1 2 25 46 48 {9 10 11 31}
P2	1 25 46 54 79 {10 11 25 27}
P3	1 2 3 {9 10 11 31} 48
P4	2 3 25 46 11 {14 15 16 48} 74
P5	2 25 46 47 48 49 {8 9 10}
P6	1 2 3 4 5 6 7
P7	25 26 27 28 30 {32 33 34 35 36}

A subsequence or pattern, P, is closed if there exists no superset of P with the same support in the database. A closed pattern induces an equivalence class of pattern sharing the same closure, i.e. all the patterns belonging to the equivalence class are verified by exactly the same set of plans. Those patterns are partially ordered, e.g. considering the inclusion relation. The smallest elements in the equivalence class are called minimal generators, and the unique maximal element is called the closed pattern.

Many algorithms have been proposed to efficiently mine sequential patterns or other time-related data [10], [11], [12]. We chose PrefixSpan [12] as it is one of the most promising approach for mining large sequence databases having numerous patterns and/or long patterns, and also because it can be extended to mine sequential patterns with user-specified constraints. PrefixSpan is a projection-based, sequential pattern-growth approach that recursively projects a sequence database into a set of smaller projected databases. Sequential patterns are grown in each projected database by exploring only locally frequent fragments. Table 2 shows some sequential patterns extracted by PrefixSpan from the data in table 1 using a minimum support of 25%.

Table 2. Examples of sequential patterns extracted by PREFIXSPAN

Sequential patterns	Sequence patterns' labels
1 25 46 48	S1
1 25 46 {10 11}	S2
1 {9 10 31}	S6
1 {9 11 31}	S7
1 {9 10 11 31}	S8
1 46 {10 11}	S13

Although a sequential pattern may not allow reaching a tutor goal in a problem-solving situation, two or more sequential patterns linked together might do so. Thus the patterns found by Prefixspan will be linked by discovering associations between them. Association rules is a powerful technique originally applied for market basket analysis that mine associations between items from a list of transactions. Since the number of extracted rules can be very high and include many redundancies, minimal and non redundant representation of association rules called generic base have been proposed such. Among previous studies on mining of generic bases, we chose IGB [13] as it efficiently extracts more compact generic bases without information loss, i.e. it has a valid and complete axiomatic system allowing the derivation of all the association rules.

3.2 Extracting Generic Rules between Patterns Using IGB

IGB [13] is a new informative generic base. Its rules are correlations between minimal premise and maximal conclusion (in term of items number). It was shown that this kind of rules is the most general (i.e., conveying the maximum of information). They are two types of generic rules: (1) factual rules having an empty premise; and (2) implicative rules having a non empty premise.

IGB base is generated by a dedicated algorithm which takes as input the meta-context of initial plans, the minimum support *minsup* (as defined in PrefixSpan), and the minimum confidence, *minconf*. The meta-context of initial plans (see example in Table 3) is the set of plans rewritten with the frequent sequential patterns obtained with PrefixSpan.

IGB algorithm checks for each non empty closed pattern P if its support is greater or equal to *minconf*. If it is the case, then the generic rule $\emptyset \rightarrow P$ is added to IGB base. Else, it iterates on all frequent closed actionsets P0 subsumed by P. For those having support at

Table 3. Part of the crisp meta-context of frequent sequences built from dataset in table 1

PlanID	Frequent sequential patterns
P1	S1 S2 S4 S5 S6 S7 S8 S9 S10 S95 S97 S98 S113 S116 S118
P2	S1 S5 S6 S7 S9 S98
P3	S1 S2 S3 S4 S5 S6 S8 S10 S95 S97 S98 S113 S116 S118
P4	S2 S3 S6 S7 S9 S10
P5	S2 S4 S5 S7 S9 S10 S95
P6	S1 S2 S3
P7	S7

least equal to *supD* (P)/*minconf*, the algorithm iterates on the list of minimal generators associated to P0. During this iteration, we look for the smallest minimal generator gs, such that there does not exist a generator g0 subsumed by gs which is already inserted in the list L of smallest premises. Then, IGB algorithm iterates on all elements of the list L to generate rules of IGB which have the following form: gs → (P – gs).

By dividing the sub-sequence occurrence by the plans' occurrence, we obtain the relative support associated to the sub-sequence. Let us consider a minsup of 2 (25%), meaning that a valid sequence should occur in at least 2 input-plans, we can obtain the meta-context which part is shown in table 3. Each sub-sequence can appear in a plan with a certain confidence which is its relative support (in table 3, we consider a crisp context where dichotomic values (0 or 1) are assigned when a subsequence appears or not in a plan). Using this meta-context as input, IGB computes a set of generic meta-rules, part of which is shown in table 4. These meta-rules combined with frequent sequential patterns will constitute the knowledge that will be used to support tutoring services.

Table 4. Examples of generic meta-rules extracted by IGB

Meta-rules	Support	Confidence	Expanded meta-rules
S10 ===> S9	4	0.8	…
S9 ===> S7	4	0.8	1 {10 31} ===> 1 {9 11 31}
S9 ===> S5	4	0.8	…
S5 ===> S10	4	0.8	…

4 Supporting Tutoring Services Using Learned Rules

As mentioned before, tutoring systems should provide useful tutoring services to assist the learner, such as coaching, assisting, guiding, helping or tracking the students during problem-solving situations. To offer these services, a tutoring system needs some knowledge related to the context. The knowledge base namely procedural task knowledge (PTK) obtained from the framework serves to that end. The next paragraphs present some examples of services that can be supported.

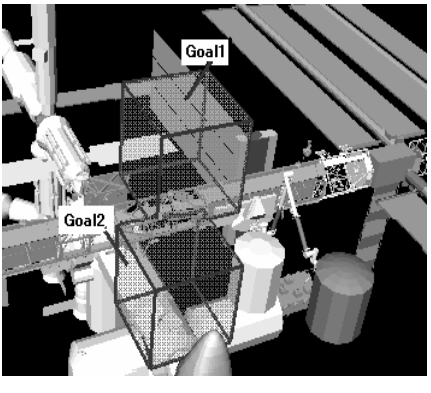
Assisting the User to Explore Possible Solutions. Domain expert users can explore, validate or annotate the PTK. The validation consists of removing all meta-rules with a low confidence, meaning that those rules can not significantly contribute to help the student. Annotation consists of connecting some useful information to meta-rules lattice depicting semantic steps of the problem as well as hints or skills associated to a given step. A meta-rule lattice annotated in this way is equivalent to [2]'s BN, except that BNs are built from scratch by domain experts. For student users, exploring PTK will help them learn about possible ways of solving problem. They can be assisted in this exploration using an interactive dialog with the system which can prompt them on their goals and helps them go through the rules to achieve these goals. This kind of service can be used when the tutoring system wants to prepare students before involving them in real problem-solving situation.

Tracking the Learner. Plan recognition is very important in tutoring systems. PTK is a great resource to this process. Each student's action can be tracked by searching the space defined by meta-rules lattice so that we can see the path being followed by detecting patterns that the learner follows.

Guiding the Learner. When solving a problem, a classic situation is when the student asks the tutor what to do next from the actual state. PTK allows the tutor to produce the next most probable actions that the student should execute and prompt him on that, taking into account uncertainty related to rules' confidence and other possible patterns that match with the current state.

5 Experiments and Results

We have set up two scenarios consisting each of moving the load to one of the two cubes (figure 1a). A total of 15 users (a mix of novices, intermediates and experts) have been invited to execute these scenarios using the CanadarmII robot simulator. A total of 155 primitive actions have been identified. Figure 1b shows part of an example log file from a user's execution of the first scenario.



...
EnterCorridor(C1)
{SelectCamera(Monitor1,CP8),
SelectCamera(Monitor2,CP10),
SelectCamera(Monitor3,CP9)}
SelectJoint(WE)
bigMove(WE,decrease)
LeaveCorridor(C1)
smallMove(WE,decrease)
SelectJoint(SP)
bigMove(SP, decrease)
SelectCamera(Monitor1,CP2)
smallMove(SP,decrease)
EnterCorridor(C2)
mediumMove(SP,decrease)

Fig. 1. (a): Environment setup for the two plan's database. (b): An entry of experimental scenarios.

We obtained a database with 45 entries each corresponding to a given usage of the system. A value indicating the failure or success of the plan has been manually added at the end of each entry. The framework presented in section 3 was applied. A unique number was assigned to each action. After coding each entry of the traces database using PrefixSpan data representation, we obtained a binary file containing plans' data for the two scenarios. This file was sent as input to the rest of the process.

After executing PrefixSpan, the first stage of the experiment consisted of finding sequential patterns from the input data, we obtained a total of 76 significant patterns (with a support greater than .5) for the first scenario, and 82 for the second scenario. At the second stage, we created a binary context where each row represents a plan

data and each column stands for a set of patterns. The goal at this stage was to mine association rules between sequential patterns.

Because all the sequential patterns shared the same first few actions and because they are closed patterns, initially the IGB approach did not find any rules. To overcome this difficulty, we filtered the patterns to remove the first common actions to all plans. Then, we regenerated the sequential patterns using PrefixPan. Using the IGB approach with a minimum support of 0.2, we obtained a PTK of 37 meta-rules. One such meta-rule (rule #31) connects the two following patterns. The first pattern (pattern #25) is to select the CP6 camera, which gives a close view of the arm in its initial position, slightly decrease the yaw of the CP6 camera to have a better view, select the WE joint and decrease a little bit its rotation value. The second pattern (pattern #55) was to select the SP joint and decrease its rotation value. These two patterns constituted a possible safe and effective strategy to move the arm toward the goal. Moreover, the confidence level of the rule (0.8), its relative support (0.25), and its expertise level annotation ("intermediate") indicated respectively that the second pattern is a common follow-up to the first pattern, that the rule is widespread among users, and that this rule is usually employed by intermediate users.

These rules were then coded and integrated in a new version of CanadarmTutor that uses this knowledge base to support tutoring services. To recognize a learner's plan, the system proceeds as follows. The first action of the learner is compared with the first action of each frequent pattern in the PTK. If the actions do not match for a pattern, the system discards the pattern. Each time the learner makes an action, the system repeats the same process. It compares the actions done so far by the learner with the remaining patterns. When a complete pattern has been identified, the software looks for association rules that link the completed pattern to other patterns that could follow. Because the generated PTK is partial, it is possible that at any given moment a user action does not match with any patterns. If this situation arises, the algorithm tries two possibilities, which are ignoring the last user action or ignoring the current action to match for each pattern. This heuristic rule makes the plan recognizing algorithm more flexible and has shown to improve its effectiveness. Furthermore, a marking scheme has been implemented to detect and avoid association rules cycles. One utility of the plan recognizing algorithm is to assess the expertise level of the learner (novice, intermediate or expert) by looking at the rules and patterns s/he applied.

The plan recognizing algorithm also plays a major role in the CanadarmTutor tutoring service for guiding the learner. It allows determining the possible actions from the current state according to the PTK. This functionality is triggered when the student selects "What should I do next?" in the interface menu. The algorithm returns a set of possible actions with the associated pattern(s) or rule(s). The tutoring service then selects the action among this set that is associated with the rule or pattern that has the highest utility value, and that is the most appropriate for the estimated expertise level of the learner. Whereas the utility value of a pattern is the pattern's relative support, the utility of a rule is calculated as the product of the relative support and the confidence of the rule. For example, the utility value of rule #31 is 0.2 (0.8 * 0.25). Utility values and expertise levels ensure that in every case the likely most useful action is proposed to the user. In the rare cases where no actions can be identified, the system asks the FADPRM path planner to generate a path to go from the current configuration to the goal.

Figure 2 illustrates a hint message given to a learner upon request during scenario 1. The guiding tutoring service recognized that the student carried out pattern #25, that rule #31 matched with the highest utility value, and that rule #31 correspond with the estimated expertise level of the learner. Therefore, the system suggested pattern #55, which is selecting the SP joint and decreasing its rotation value. By default, two steps are showed to the learners in the hint window depicted in figure 2. However, the learner can click on the "More" button (fig. 2) to ask for more steps or click on the "another possibility" button to ask for an alternative. The sentences in natural language depicted in figure 2 to describe the actions of pattern #55 are an example of tutoring resources that can be used to annotate the PTK

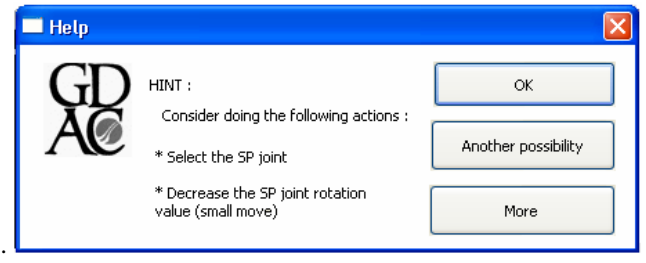


Fig. 2. A hint generated by the guiding tutoring service

An empirical test with this version has been conducted with the same users of the system's version relying on FADPRM. We found that the system behavior in terms of guiding the user during the two scenarios significantly improved compared to the behavior observed in the version relying solely on the path planner. The system can now recommend good and easy-to-follow actions sequences. The system can also recognize users' plans and anticipate failures or successes, thus infer user profiles by detecting the path they follow. The PTK produced by our framework is sometimes too large and contains non useful rules (because of the amount of sequential patterns). However, this is not harmful for the tutor behavior but it may slow the performance as the system need to go through this huge knowledge base each time the user executes an action. We are now working to improve the quality of the PTK. We are also looking for a way of managing unsuccessful plans data. We believe that this could allow better learning guidance, as the tutor could easily identify sequence patterns that lead to failure or success.

6 Conclusion

We proposed a knowledge discovery framework to learn procedural knowledge associated to a task. We showed how the framework contributes to enhance an intelligent tutoring system's domain knowledge in an ill-defined procedural domain. The experiment showed that this enables CanadarmTutor to better help learners.

Prefixspan and IGB are parameters of our framework proposal, and thus can be replaced by other convenient tools. Since these tools and the input and output of the

framework are domain independent, the framework can be potentially applied to any ill-defined procedural domains where the problem can be stated in the same way.

For future works, it would be interesting to find some ways of filtering the resulting meta-rules and integrating unsuccessful paths. The work of Kum et al. [17] provides some suggestions on a model that may help to filter sequential patterns. Different proposals have also been made on qualitative measures of association rules. We will also carry out further tests to clearly measure the benefit of the approach in terms of tutoring assistance services.

Another important aspect that we will focus on is binding users' skills with discovery sequences. In fact, we found that, it will be interesting to compute a subset of skills that characterized a pattern by finding common skills demonstrated by users who used that pattern. This will allow a thorough cognitive diagnosis of missing and misunderstanding skill for the users who demonstrate part of that pattern and therefore, help them to acquire correct skill to be able to solve the goal.

Acknowledgments. Our thanks go to the Canadian Space Agency, the NSERC (Natural Sciences and Engineering Research Council) for their logistic and financial support. The authors also thanks the current and past members of the GDAC and PLANIART research teams who have participated to the development of the CanadarmTutor. We also thank the authors of PrefixSpan and IGB for providing their programs.

References

[1] Aleven, V., Ashley, K., Lynch, C., Pinkwart, N.: Proc. of the Intelligent Tutoring Systems for Ill-Defined Domains Workshop. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053. Springer, Heidelberg (2006)

[2] Aleven, V., McLaren, B.M., Sewall, J., Koedinger, K.: The Cognitive Tutor Authoring Tools (CTAT): Preliminary evaluation of efficiency gains. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 61–70. Springer, Heidelberg (2006)

[3] Mitrovic, A., Mayo, M., Suraweera, P., Martin, B.: Contraint-based tutors: a success story. In: Proc. of the Industrial & Engineering Application of Artificial Intelligence & Expert Systems, pp. 931–940 (2001)

[4] Nkambou, R., Belghith, K., Kabanza, F.: An Approach to Intelligent Training on a Robotic Simulator Using an Innovative Path-Planner. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 645–654. Springer, Heidelberg (2006)

[5] McLaren, B., Koedinger, K.R., Schneider, M., Harrer, A., Bollen, L.: Bootstrapping Novice Data: Semi-Automated Tutor Authoring Using Student Log Files. In: Proc. of the Workshop on Analyzing Student-Tutor Logs. ITS 2004 (2004)

[6] Blessing, S.B.: A Programming by Demonstration Authoring Tool for Model-Tracing Tutors. In: Murray, T., Blessing, S., Ainsworth, S. (eds.) Authoring Tools for Advanced Technology Learning Environments: Toward Cost-Effective Adaptive, Interactive and Intelligent Educational Software, pp. 93–119. Kluwer Academic Publ. (2003)

[7] Jarivs, M., Nuzzo-Jones, G., Heffernan, N.T.: Applying Machine Learning Techniques to Rule Generation in Intelligent Tutoring Systems. In: Proc. of ITS 2006, pp. 541–553 (2006)

[8] Fournier-Viger, P., Nkambou, R., Mayers, A., Dubois, D.: Automatic Evaluation of Spatial Representations for Complex Robotic Arms Manipulations. In: Proc. of ICALT 2007, pp. 279–281 (2007)
[9] Mitrovic, A., Koedinger, K.R., Martin, B.: A Comparative Analysis of Cognitive Tutoring and Constraint-Based Modeling. In: User Modeling 2003, pp. 313–322 (2003)
[10] Agrawal, R., Srikant, R.: Mining Sequential Patterns. In: Proc. of the Int. Conf. on Data Engineering, pp. 3–14 (1995)
[11] Zaki, M.J.: SPADE: An Efficient Algorithm for Mining Frequent Sequences. Machine Learning Journal 42(1-2), 31–60 (2001)
[12] Pei, J., Han, J., et al.: Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach. IEEE Trans. Knowledge and Data Engineering 16(10), 1–17 (2004)
[13] Gasmi, G., Yahia, S.B., Mephu Nguifo, E., Slimani, Y.: A new informative generic base of association rules. In: Proc. Ninth Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 81–90 (2005)
[14] Kum, H., Chang, J.H., Wang, W.: Benchmarking the effectiveness of sequential pattern mining methods. Data and Knowledge Engineering (DKE) 60(1), 30–50 (2007)
[15] Simon, H.A.: Information-processing theory of human problem solving. In: Estes, W.K. (ed.) Handbook of learning and cognitive processes, vol. 5, Human information (1978)
[16] Lynch, C., Ashley, K., Aleven, V., Pinkwart, N.: Defining Ill-Defined Domains; A literature survey. In: Proc. of the Intelligent Tutoring Systems for Ill-Defined Domains Workshop ITS 2006, pp. 1–10 (2006)
[17] Fields, A.M.: Ill-structured problems and the reference consultation: The librarian's role in developing student expertise. Reference services review, Emerald 34(3), 405–420 (2006)

More Accurate Student Modeling through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing

Ryan S.J.d. Baker, Albert T. Corbett, and Vincent Aleven

Human-Computer Interaction Institute, Carnegie Mellon University
{rsbaker,corbett,aleven}@cmu.edu

Abstract. Modeling students' knowledge is a fundamental part of intelligent tutoring systems. One of the most popular methods for estimating students' knowledge is Corbett and Anderson's [6] Bayesian Knowledge Tracing model. The model uses four parameters per skill, fit using student performance data, to relate performance to learning. Beck [1] showed that existing methods for determining these parameters are prone to the *Identifiability Problem:* the same performance data can be fit equally well by different parameters, with different implications on system behavior. Beck offered a solution based on Dirichlet Priors [1], but, we show this solution is vulnerable to a different problem, *Model Degeneracy*, where parameter values violate the model's conceptual meaning (such as a student being more likely to get a correct answer if he/she does not know a skill than if he/she does). We offer a new method for instantiating Bayesian Knowledge Tracing, using machine learning to make contextual estimations of the probability that a student has guessed or slipped. This method is no more prone to problems with Identifiability than Beck's solution, has less Model Degeneracy than competing approaches, and fits student performance data better than prior methods. Thus, it allows for more accurate and reliable student modeling in ITSs that use knowledge tracing.

1 Introduction

Modeling students' knowledge is a fundamental part of intelligent tutoring systems. Key aspects of modern intelligent tutoring systems, such as deciding which problems to give students [cf. 6], are reliant upon accurately estimating each student's knowledge state at any given time. Giving students appropriate amounts of practice on each skill promotes complete and efficient learning [4]; both over-practice and under-practice can be avoided through having student knowledge models that are as accurate and dependable as possible.

In recent years, Corbett & Anderson's Bayesian Knowledge Tracing model [6] has been used to model student knowledge in a variety of systems, including tutors for mathematics [9], computer programming [6], and reading skill [2], and is statistically equivalent to the two-node dynamic Bayesian network used in many other learning environments [10]. Bayesian Knowledge Tracing keeps a running assessment of the probability that a student currently knows each skill, continually updating that estimate based on

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 406–415, 2008.

student behavior. Cognitive Mastery Learning built on top of Bayesian Knowledge Tracing has been shown to significantly improve student learning [6].

However, a recent paper by Beck and Chang [2] gives evidence that current methods for developing Bayesian Knowledge Tracing models for specific skills are vulnerable to a statistical problem, the ***identifiability*** problem, where models with equally good statistical fit to performance data may make very different predictions about a student's knowledge state, and correspondingly may assign very different numbers of problems to a student. To address this problem, Beck and Chang [2] proposed constraining model parameters by finding a prior probability across all skills. However, as we will show in this paper, Beck and Chang's solution is vulnerable to a different statistical problem, which we term ***model degeneracy***, where it is possible to obtain model parameters which lead to paradoxical behavior, such as the probability the student knows a skill dropping after three correct answers in a row. In this paper, we propose both theoretical and empirical definitions of this problem.

These two problems, at their core, arise from how these models handle uncertainty – in particular, how these models address the possibility of a student slipping (knowing a skill, but giving a wrong answer) or guessing (giving a correct answer, despite not knowing the skill). Within this paper, we propose a new way to assess uncertainty within knowledge tracing, using machine learning to make contextual estimations of the probability that a student has guessed or slipped. We show that this method leads to a significantly closer fit between models and student performance than prior methods, has lower model degeneracy than these approaches, and that there is reason to believe that this method will not suffer from the identifiability problem.

1.1 Bayesian Knowledge Tracing

Corbett and Anderson's Bayesian Knowledge Tracing model [6] computes the probability that a student knows a given skill at a given time, combining data on the student's performance up to that point with four model parameters. In the model's canonical form, each problem step in the tutor is associated with a single cognitive skill. The model assumes that at any given opportunity to demonstrate a skill, a student either knows the skill or does not know the skill, and may either give a correct or incorrect response (help requests are treated as incorrect by the model). A student who does not know a skill generally will give an incorrect response, but there is a certain probability (called **G**, the Guess parameter) that the student will give a correct response. Correspondingly, a student who does know a skill generally will give a correct response, but there is a certain probability (called **S**, the Slip parameter) that the student will give an incorrect response. At the beginning of using the tutor, each student has an initial probability ($\mathbf{L_0}$) of knowing each skill, and at each opportunity to practice a skill the student does not know, the student has a certain probability (**T**) of learning the skill, regardless of whether their answer is correct.

The system's estimate that a student knows a skill is continually updated, every time the student gives a first response (correct, incorrect, or a help request) to a problem step. First, the system re-calculates the probability that the student knew the skill before the response, using the evidence from the response (help requests are treated as evidence that the student does not know the skill), using the first two equations of Figure 1. Then, the system accounts for the possibility that the student learned the

skill during the problem step, using the third equation of Figure 1. Within the Cognitive Mastery algorithm used in most Cognitive Tutors [6], the student is assigned additional problems on skills that the system does not yet believe that the student has learned (e.g. skills that the student has less than 95% probability of knowing).

$$P(L_{n-1}|Correct_n) = \frac{P(L_{n-1}) * (1 - P(S))}{P(L_{n-1}) * \big(1 - P(S)\big) + \ (1 - P(L_{n-1})) * (P(G))}$$

$$P(L_{n-1}|Incorrect_n) = \frac{P(L_{n-1}) * P(S)}{P(L_{n-1}) * P(S) + \ (1 - P(L_{n-1})) * (1 - P(G))}$$

$$P(L_n|Action_n) = \ P(L_{n-1}|Action_n) + \big((1 - P(L_{n-1}|Action_n)\big) * P(T))$$

Fig. 1. The equations used to predict student knowledge from behavior in Bayesian Knowledge Tracing

The four parameters in Bayesian Knowledge Tracing are fit, for each skill, using data from students using that skill within an intelligent tutor. The goal during parameter fitting is to figure out which combination of parameters best predicts the pattern of correct and incorrect responses in the existing data, and then to use that model to make predictions about new students' knowledge as they use the tutor.

Challenges in Estimating Parameters for Bayesian Knowledge Tracing Models

Recently, Beck and Chang [2] showed that common methods of fitting Bayesian Knowledge Tracing models suffer from the ***identifiability*** problem; different combinations of the four parameters can fit the same data equivalently well, but yield very different estimates of the probability that the student knows the skill at any given time. This is of practical importance, as different model parameters may require very different amounts of practice before inferring that the student has reached mastery.

A second challenge for fitting models in Bayesian Knowledge Tracing approach is what we term ***model degeneracy.*** The conceptual idea behind using Bayesian Knowledge Tracing to model student knowledge in intelligent tutors is that knowing a skill generally leads to correct performance, and that correct performance implies that a student knows the relevant skill; hence, by looking at whether a student's performance is correct, we can infer whether they know the skill. A model deviates from this theoretical conception, and thus is ***theoretically degenerate***, when its guess (**G**) parameter or slip (**S**) parameter is greater than 0.5. A slip parameter over 0.5 signifies that a student who knows a skill is more likely to get a wrong answer than a correct answer; similarly, a guess parameter over 0.5 implies that a student who does not know a skill is more likely to get a correct answer than a wrong answer.

It is also possible to conceive of empirical tests that show that a model violates the linkage between knowledge and performance; we term a model that fails such a test ***empirically degenerate.*** We propose two tests for empirical degeneracy. First, if a student's first N actions in the tutor are correct, the model's estimated probability that the student knows the skill should be higher than before these N actions. Second, if the student makes a large number M of correct responses in a row, the model should assess that the student has mastered the skill. The exact values of N and M are

arbitrary – within this paper, we choose N=3 and M=10 as reasonable cut-off points for the two tests. In other words, if a student's first three actions in the tutor are all correct, but the model's estimated probability that the student knows the skill is lower than before these three actions, we say that the model failed the first test of empirical degeneracy. If a student gets a skill correct ten times in a row without reaching skill mastery, we say that the model failed the second test of empirical degeneracy.

Three Prior Approaches to Model Fitting in Bayesian Knowledge Tracing

The simplest ***baseline approach*** to fitting a Bayesian Knowledge Tracing model is to allow each of the four parameters to take on any value between 0 and 1. We fit parameters for this approach using Bayes Net Toolkit-Student Modeling (BNT-SM) [1].

An alternate approach is to bound the guess and slip parameters (the ***bounded guess and slip approach***). Generally, in existing tutors, the guess parameter is bounded to be between 0 and 0.3, and the slip parameter is bounded to be between 0 and 0.1, based on the most common number of candidate actions, and pragmatically, in order to err in the direction of requiring less practice for mastery. Though this approach was not explicitly designed to prevent model degeneracy, it makes theoretical degeneracy impossible. We fit parameters for this approach using Microsoft Excel.

A third way to fit a Bayesian Knowledge Tracing model is the ***Dirichlet Priors*** approach proposed in [1, 2]. A Gaussian probability distribution is found for how often different values of each parameter are seen across skills, and then the parameters of all skills are constrained by these prior probabilities. This approach biases all skills towards parameters that fit the whole data set well, with skills that have less data biased more strongly than skills that have large amounts of data. The prior probabilities lead to a single model always being the best-fitting model among the space of potential models, for each skill. We fit parameters for this approach using BNT-SM.

2 Analyzing Degeneracy in Previous Approaches

Prior work has already shown that the baseline and the bounded guess-and-slip approaches are vulnerable to the identifiability problem; the Dirichlet priors approach gives a single prediction, offering a response to the identifiability problem [1, 2]. In this section, we use data from the Middle School Tutor [9], an intelligent tutor which covers a wide span of mathematical topics covered by 6^{th}-8^{th} grade students (approximately 12-14 years old), to analyze whether these three model-fitting approaches are prone to problems with model degeneracy, and examine their accuracy. 232 students used the Middle School Tutor during the course of the 2002-2003 school year, making 581,785 transactions (either entering an answer or requesting a hint) on 171,987 problem steps covering 253 skills in 37 tutor lessons/units. 290,698 additional transactions were not included in either these totals or in our analyses, because they were not labeled with skills, information needed to apply Bayesian Knowledge Tracing.

Table 1 shows the level of theoretical and empirical model degeneracy for each of the three approaches. 76% of skills in the Dirichlet priors model and 75% of skills in the baseline model were theoretically degenerate; as a direct consequence of bounding

Table 1. The number (proportion) of degenerate skills in each model

Modeling Approach	Theoretically Degenerate Skills	Skills where a student who gets first three actions correct has lower P(**L**) afterwards (Empirical degeneracy test 1)	Skills where a student cannot reach mastery with 10 correct answers in a row (Empirical degeneracy test 2)
Baseline	189 (75%)	4 (2%)	57 (23%)
Bounded Guess and Slip	0 (0%)	0 (0%)	12 (5%)
Dirichlet Priors	192 (76%)	4 (2%)	57 (23%)

guess and slip, 0% of skills in the bounded guess and slip model were theoretically degenerate. The difference between the bounded guess and slip model and each of the other two models was statistically significant, using the test of the significance of the difference between correlated proportions with McNemar's standard error estimate [7], Z=13.86, Z=13.75, two-tailed p<0.0001. The Dirichlet priors and baseline model were not significantly different from one another, Z=0.83, two-tailed p=0.41. Across models, failures of the first test of empirical degeneracy were rare; only 2% of the skills in the Dirichlet priors and baseline models failed this test, and 0% of skills in the bounded guess and slip model failed. However, 23% of the skills in the Dirichlet priors and baseline models failed the second test of empirical degeneracy. Fewer (5%) of the skills in the bounded guess and slip model failed the second test of empirical degeneracy, in both cases Z=5.58, two-tailed p<0.0001.

3 Contextual Estimation of Guess and Slip

In this section, we will discuss a new approach to Bayesian Knowledge tracing, which removes one of the framework's assumptions, to address these modeling issues. In all three prior approaches, each of the four parameters is held constant across contexts, for any given skill. (One other prior approach changed parameter values depending on whether help was used or not [5], and anticipates our approach, although that approach's contextualization was far simpler than what is proposed here).

In the new approach we propose, we contextually estimate whether each individual student response is a guess or a slip, rather than using fixed guess and slip probability estimates across all situations. In this section, we describe our method for predicting whether individual actions are guesses or slips. We will then discuss how these predictions are integrated into a new approach to Bayesian Knowledge Tracing. Our method is as follows:

- We take a set of correct responses in the log files. For each correct student response, we apply a Bayesian analysis to estimate the probability the student knew the applicable rule or guessed, based on the student's performance on successive opportunities to apply the rule. A similar procedure is used to assess whether each non-correct response stemmed from the student not knowing the skill, or from knowing the skill but slipping.

- We use machine learning to identify features of an action that characterize whether that action was a guess or a slip. These features do not use any information from subsequent actions; hence, they can be used to predict whether an action is a guess or a slip immediately after it occurs.
- In modeling student problem-solving, we use the machine learned models to dynamically estimate the probability that a response is a guess or a slip. We employ these dynamic performance estimates in the Bayesian Knowledge Tracing algorithm to update the probability that the student knows the skill.

The first step is to label a set of existing student actions with the probability that these actions involve guessing or slipping, to serve as inputs to a machine learning algorithm. The set of student actions to be labeled is drawn from the set of first actions on the 64 skills for which the Dirichlet Priors model is not theoretically degenerate. We chose to use skills that are not theoretical degenerate to avoid training our models to include model degeneracy, and used Dirichlet Priors in order to avoid creating an equivalence class of potential models (i.e. the identifiability problem). We then use estimates from this model in order to create the contextual guess and slip models.

We label student actions (***N***) with the probability that they represented a guess or slip, using information about the two actions afterwards (***N+1***, ***N+2***). Using information about future actions gives considerable information about the true probability that a student's action at time ***N*** was due to knowing the skill – if actions ***N, N+1***, and ***N+2*** are all correct, it is relatively unlikely that ***N***'s correctness was due to guessing.

The probability that the student guessed or slipped at time ***N*** (i.e., the action at time ***N,*** which we term $\boldsymbol{A_n}$) is directly obtainable from the probability that the student knew the skill at time ***N***, given knowledge about the action's correctness:

$$P(A_n \text{ is guess} \mid A_n \text{ is correct}) = 1 - P(L_n) \qquad P(A_n \text{ is slip} \mid A_n \text{ is incorrect}) = P(L_n)$$

We can calculate the probability that the student knew the skill at time ***N***, given information about the actions at time ***N+1*** and ***N+2*** (which we term $\boldsymbol{A_{+1+2}}$). We do so by using Bayes' Rule to combine 1) the probability of the actions at time ***N+1*** and ***N+2*** ($\boldsymbol{A_{+1+2}}$), given the probability that the student knew the skill at time ***N*** ($\mathbf{L_n}$); 2) the prior probability that the student knew the skill at time ***N*** ($\mathbf{L_n}$); and 3) the initial probability of the actions at time ***N+1*** and ***N+2*** ($\boldsymbol{A_{+1+2}}$).

In equation form, this gives: $P(L_n \mid A_{+1+2}) = \frac{P(A_{+1+2} \mid L_n) * P(L_n)}{P(A_{+1+2})}$

The probability of the actions at time ***N+1*** and ***N+2*** is computed as

$$P(A_{+1+2}) = P(L_n) * P(A_{+1+2} \mid L_n) + (1 - P(L_n)) * P(A_{+1+2} \mid \sim L_n)$$

The probability of the actions at time ***N+1*** and ***N+2,*** in the case that the student knew the skill at time ***N*** ($\mathbf{L_n}$)***,*** is a function of the probability that the student guessed or slipped at each opportunity to practice the skill. C denotes a correct action; ~C denotes an incorrect action (an error or help request).

$$P(A_{+1+2} = C, C \mid L_n) = P(\sim S)^2 \qquad P(A_{+1+2} = C, \sim C \mid L_n) = P(G)P(\sim S)$$

$$P(A_{+1+2} = \sim C, C \mid L_n) = P(G)P(\sim S) \qquad P(A_{+1+2} = \sim C, \sim C \mid L_n) = P(G)^2$$

The probability of the actions at time ***N+1*** and ***N+2,*** in the case that the student did not know the skill at time ***N*** (**L_n**), is a function of the probability that the student learned the skill between actions ***N*** and ***N+1***, the probability that the student learned the skill between actions ***N+1*** and ***N+2***, and the probability of a guess or slip.

$$P(A_{+1+2} = C, C| \sim L_n) = P(T)P(\sim S)^2 + P(\sim T)P(T)P(G)P(\sim S) + P(\sim T)^2P(G)^2$$

$$P(A_{+1+2} = C, \sim C| \sim L_n) = P(T)P(\sim S)P(S) + P(\sim T)P(T)P(G)\big(P(S)\big) + P(\sim T)^2P(G)P(\sim G)$$

$$P(A_{+1+2} = \sim C, C| \sim L_n) = P(T)P(S)P(\sim S) + P(\sim T)P(T)P(\sim G)P(\sim S) + P(\sim T)^2P(\sim G)P(G))$$

$$P(A_{+1+2} = \sim C, \sim C| \sim L_n) = P(T)P(S)^2 + P(\sim T)P(T)P(\sim G)P(S) + P(\sim T)^2P(\sim G)^2$$

Once the actions are labeled with estimates of whether they were guesses or slips, we use these labels to create machine-learned models that can accurately predict at run-time whether a given action is a guess or slip. The original labels were developed using future knowledge, but the machine-learned models predict guessing and slipping using only data about the action itself (no future data).

For each action, we distilled a set of 23 features describing that action; the features used in the final models are shown in Table 2. We then used Linear Regression, within Weka [11], to create 2 models predicting the probability of guessing or slipping. Linear Regression gave slightly better performance under 10-fold cross-validation than a Support Vector Machine or Multilayer Perceptron – r=0.44 within the guess model, and r=0.38 within the slip model.

Then, when we have models that can predict the probability that any action was a guess or a slip, we can label the first action of each opportunity to use a skill with predictions as to how likely it is to be a guess and slip. Then, parameter values can be

Table 2. The machine learned models of guessing (left) and slipping (right). In the unusual case where output values fall outside the range {0,1}, they are bounded to 0 or 1.

Feature	**P(G)=**	**P(S)=**
Action is a help request		+ 0.066
Percent of past opportunities where student has requested help on this skill		- 0.047
Percent of past opportunities where student has made errors on this skill		- 0.004
Response is a string	+ 0.049	- 0.02
Time taken	+ 0.002	- 0.0002
Time taken (SD faster (-) or slower (+) than average across all students)	- 0.024	+ 0.01
Time taken in last 5 actions (calculated in SD off average across students)	- 0.003	+ 0.002
Total number of times student has gotten this skill wrong on the first try	- 0.002	+ 0.0002
Total time taken on this skill so far (across all problems)	+ 0.001	- 0.001
Number of last 5 actions which involved same interface element	+ 0.014	- 0.026
Number of last 8 actions which involved help request	+ 0.042	- 0.019
Number of last 5 actions which were wrong	+ 0.036	- 0.033
At least 3 of last 5 actions involved same interface element & were wrong	+ 0.067	+ 0.013
Number of opportunities student has already had to use current skill	+ 0.003	- 0.001
Constant term	+ 0.066	+ 0.442

fit for P(**T**) and P($\mathbf{L_0}$), for each skill, using curve-fitting. At this point, we have a model that makes predictions about student knowledge each time they attempt to use a skill for the first time on a given problem step. This model also involves considerably fewer parameters than previous models – whereas all three prior models had exactly 4 parameters per skill, this model has 2 parameters fit per skill, and 27 parameters fit across all skills, for an average of 2.11 parameters per skill.

4 Evaluating the Contextual Guess and Slip Model

Are models created by the contextual guess and slip method identifiable? The initial skill models used to create the labels for the linear regression process were generated by the Dirichlet priors method, and thus represent an optimal and unique parameter set for that method [2]. Linear Regression itself has a single optimal solution [3]. Finally, with only two parameters to fit, the contextual guess and slip model of each skill has a unique best solution for each pair of parameters P(**T**) and P($\mathbf{L_0}$). Hence, since each step in the model fitting process has a single optimal solution, the contextual guess and slip method is as identifiable as the Dirichlet Priors method.

What about model degeneracy? We can test for the two types of empirical model degeneracy using the student log data. A model fails the first test of empirical model degeneracy when a student gets the first three actions correct on a specific skill but then has lower P(**L**) afterwards. There were 2558 cases in the data where a student got the first three actions correct on a specific skill; in only 1 of the 2558 cases did the student have a lower P(**L**) afterwards – and, in that case, the student got the skill incorrect on the next 7 opportunities. The proportion of failure of the first test (1/2558 = 0.0004%) is significantly lower than the proportion of failure of this test in the baseline or Dirichlet Priors models, in each case $t(251) = -2.00$, two-tailed $p=0.05$, for a paired t-test (comparing model performance within each skill), but is not significantly higher than the proportion of failure of the first test for the bounded model, $t(251) = 1.00$, two-tailed $p=0.32$.

A model fails the second test of empirical degeneracy if a student gets ten actions correct in a row but does not reach mastery. There were 758 cases in the data where a student got the first ten actions correct on a specific skill; in 13 of the 758 cases the student afterwards had a P(**L**) below mastery (0.95). This proportion (1.7%) is significantly lower than the baseline, Dirichlet Priors, and bounded models, respectively, $t(251) = -8.42$, $t(251) = -8.42$, $t(251) = -3.37$, in all three cases two-tailed $p<0.001$.

Hence, there is evidence for limited degeneracy in the Contextual Guess and Slip model, but this model is substantially less degenerate than the baseline or Dirichlet Priors models, and appears to be less degenerate than the bounded model as well.

There are two ways to measure the accuracy of the four knowledge tracing models. The first is to compare actions at time ***N*** to the models' predictions of the probability that actions at time ***N*** will be correct – P($\mathbf{L_n}$)*P(~**S**)+ P(~$\mathbf{L_n}$)*P(**G**). This method accurately represents exactly what each model predicts; however, this method biases in favor of the Contextual Guess and Slip model, since that model uses information associated with the answer being predicted to estimate the probability of guessing and slipping. An alternate measure which is appropriate for all four models is to compare actions at time ***N*** to the models' predictions of the probability that the student knew the skill at time ***N***, before the student answered. This method under-estimates accuracy

Table 3. Each model's accuracy across the 171,989 first actions. Comparisons use model prediction of knowledge state after previous attempt at skill. Standard errors given in parentheses.

Modeling Approach	A'	r
Baseline	0.66 (0.001)	0.29
Bounded Guess and Slip (Corbett's method)	0.61 (0.001)	0.25
Dirichlet Priors (Beck's method)	0.65 (0.001)	0.26
Contextual Guess and Slip	0.75 (0.001)	0.43

for all models (since it does not include the probability of guessing and slipping when answering), but is preferable because it does not favor any model. We use A' (the probability that the model can distinguish a correct response from an incorrect response [8]) and correlation as the measures of model accuracy.

The full pattern of results is shown in Table 3. The Contextual Guess and Slip method achieves the highest value of A', 0.75. The second-best model is the Baseline model, with A' of 0.66. The Contextual Guess and Slip model's A' achieves 27% of the possible improvement over the Baseline model, a statistically significant difference in fit, $Z=2.86$, two-tailed $p<0.01$ (an adjusted standard error is used for A' to control for type II error stemming from non-independence). The Contextual Guess and Slip method also achieves the highest correlation, 0.43, 48% higher than the second-best model, again the Baseline model ($r=0.29$), $t(171984)=69.12$, two tailed $p<0.0001$. (This test is under-conservative, as it assumes independence; if we collapse across students, an overly conservative test, the result remains significant, $t(231)=7.95$, two tailed $p<0.0001$). Hence, for both measures of model accuracy, Contextual Guess and Slip performs substantially better than prior knowledge tracing methods.

5 Conclusions

In this paper, we have proposed a new way to contextually estimate the probability that a student obtained a correct answer by guessing, or an incorrect answer by slipping, within Bayesian Knowledge Tracing. The method we propose is less vulnerable to model degeneracy than previous methods of student knowledge modeling, and is as good as the best of previous approaches at dealing with challenges to identifiability. In addition, our method leads to substantially higher accuracy than prior methods – improving A' by 27% of potential gain, and improving correlation by 48%. Plus, the method seems quite generalizable; the machine-learned models of guess and slip was trained on only 64 skills, but functioned effectively within all 253 skills it was tested on. An interesting area for future research will be studying how widely the guess and slip models can be transferred with no re-training at all, and still function effectively. Similarly, it will be important to replicate this result in data from another Cognitive Tutor, and in other intelligent tutors [cf. 5].

Though the contextual estimation of guess and slip has proven more successful than earlier student knowledge modeling, we see this paper as just the beginning of a new, more contextually sensitive approach to student modeling. First of all, it is probably possible to increase the accuracy of the contextual estimates of slip even further, by incorporating data about second and subsequent attempts within a given opportunity to practice a skill (an error followed very rapidly by the correct answer is

probably much more likely to be a slip than an error followed by three more slow errors and a help request). Second, it is possible to combine overall estimation of the probability of guesses and slips (as used here) with information about individual skills, potentially raising accuracy further still. Third, it is possible to estimate the probability of learning a skill P(**T**) at any given time in the same contextual fashion as used here. We look forward to new possibilities for substantially more sensitive and accurate estimation of student knowledge. And, in the long term, more sensitive and accurate estimation of student knowledge will have considerable pay-offs: it will enable more accurate assignment of learning materials to students, optimize the amount of practice on each skill [cf. 4], and may even enable different types of remediation for different types of errors (such as giving specific remediation for slips).

Acknowledgements

We would like to thank Project LISTEN and Joseph Beck for offering the BNT-SM toolkit which made this comparison of models feasible. This work was funded by NSF grant REC-043779 to "IERI: Learning-Oriented Dialogs in Cognitive Tutors: Toward a Scalable Solution to Performance Orientation", and by the Pittsburgh Science of Learning Center, National Science Foundation award SBE-0354420.

References

1. Beck, J.: Difficulties in inferring student knowledge from observations (and why you should care). In: Educational Data Mining: Supplementary Proceedings of the 13th International Conference of Artificial Intelligence in Education, pp. 21–30 (2007)
2. Beck, J.E., Chang, K.-m.: Identifiability: A Fundamental Problem of Student Modeling. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511. Springer, Heidelberg (2007)
3. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
4. Cen, H., Koedinger, K.R., Junker, B.: Is Over Practice Necessary? Improving Learning Efficiency with the Cognitive Tutor. In: Proceedings of the 13th International Conference on Artificial Intelligence and Education (2007)
5. Chang, K., Beck, J., Mostow, J., Corbett, A.T.: Does Help Help? A Bayes Net Ap-proach to Modeling Tutor Interventions. In: Proceedings of the AAAI 2006 Workshop on Educational Data Mining (2006)
6. Corbett, A.T., Anderson, J.R.: Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modeling and User-Adapted Interaction 4, 253–278 (1995)
7. Ferguson, G.A.: Statistical Analysis in Psychology and Education. McGraw-Hill, New York (1971)
8. Hanley, J.A., McNeil, B.J.: The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. Radiology 143, 29–36 (1982)
9. Koedinger, K.R.: Toward evidence for instructional design principles: Examples from Cognitive Tutor Math 6. In: Proceedings of PME-NA XXXIII (the North American Chapter of the International Group for the Psychology of Mathematics Education) (2002)
10. Reye, J.: Student Modeling based on Belief Networks. International Journal of Artificial Intelligence in Education 14, 1–33 (2004)
11. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)

Interoperable Competencies Characterizing Learning Objects in Mathematics

Erica Melis[2], Arndt Faulhaber[1], Anja Eichelmann[3], and Susanne Narciss[3]

[1] University of Saarland
[2] German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
[3] Technische Universität Dresden, Germany
melis@dfki.de, arndt.faulhaber@dfki.de

Abstract. Cognitive task analysis has been used in ITSs to predict students' performance, improve curricula and to determine appropriate feedback. Typically, the learning factors/knowledge components have been determined only for the use in *one* ITS or curriculum and therefore, general frameworks were not applied. Moreover, the result is sometimes rather unsystematic and not reusable across domains. However, for making learning environments interoperable and comparable and to be able to reuse learning objects, the competency hierarchies have to be usable for different learning environments and across domains. In this paper, we propose an approach to competencies represented as pairs of knowledge and cognitive process whose ontologies extend and revise existing taxonomies. A goal is to make these competencies a quasi-standard that enables interoperability and reuse. Moreover, we briefly describe, how the competency ontology can be employed for different purposes.

1 Introduction

The cognitive analysis of domains and tasks is a common requirement for designing an ITS and/or its content as well as for designing the structure of student models. The competencies resulting from this analysis (also called learning factors, knowledge elements, skills, or knowledge components) have been used to characterize tasks, exercises and exercise steps with the goal to choose appropriate exercises, and to improve a curriculum, e.g. in [1, 2, 3]. In pedagogical psychology, cognitive task analysis plays a role for diagnosing and evaluating students' learning progress [4].

As a general framework for such an analysis, Bloom devised his well-known taxonomy of educational objectives [5]. The cognitive objectives he addressed are general and domain independent and a relation to domain-specific settings is missing. More recent work in this field has extended Bloom's taxonomy [6].

As opposed to Bloom, the learning factor/knowledge component analysis for ITSs typically focuses on domain knowledge [2] and targets a set of knowledge elements for the usage in *one* ITS, sometimes even without a hierarchical structure.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 416–425, 2008.

[1] For instance, the knowledge components *integers* and *addition* in exercises of the ASSISTment system [2] denote elements of a domain ontology without explicitly addressing the cognitive or meta-cognitive processes needed in an exercise for integers or addition. However, for making learning environments interoperable and comparable and to be able to reuse learning objects and competency annotations, the competency hierarchies have to be usable across different learning environments and (partially) across domains.

In recent years, the appropriateness of the competencies resulting from the knowledge component analyses has been questioned wrt. the actual elements and their granularity [2, 7]. The main goal of the questioning was to improve the design of curricula and the prediction of students' performance. Now, we question the knowledge component/competency representation. The need for this arises from the aim of sharing content, reuse components, and services for learning environments and from the use of (standardized) metadata in web-based learning systems.

In this paper, we propose an approach that extends existing work, fits the needs of cognitive task analysis in various domains (e.g. fractions and calculus) and that is implemented in the ActiveMath platform. The main contributions are an ontology which refines a recent competency hierarchy that pair knowledge with cognitive processes, (partial) translations of previous taxonomies into the new ontology, and a demonstration of usages of the resulting competencies.

2 Previous Competency Systems

Learning objects such as exercises as well as their steps can be characterized (in technical terms: annotated) with the competencies required to solve a problem/succeed with a step.

For the ASSISTment Project Heffernan et al. have empirically determined so-called knowledge components (called skills) to characterize exercises and their steps [7]. These knowledge components include: *integers, addition, rounding, ordering numbers, reduce fractions, equivalence of fractions and decimal percents, equiliteral triangle, evaluating functions, finding percents, statistics, making sense of expressions and equations, order of operations, graph shape, reading graph, divide decimals, ...* and have been defined by subject matter experts and by analyzing the Massachusetts curriculum. All of these knowledge components point to knowledge (a concept, rule, or procedure of the domain). Some of these knowledge components could also be interpreted as pointing to knowledge *and* a cognitive process.

Bloom [5] describes a hierarchy of educational goals that include:

- `Knowledge`: remembering; memorizing; recognizing; recall identification; recall of information

[1] Typically, the learning factors/knowledge components have been determined empirically by expert judgement for exercises or by analyzing curricula.

[2] Personal communication with Neil Heffernan.

- `Comprehension`: interpreting; translating from one medium to another; describing in one's own words; organization and selection of facts and ideas
- `Application`: problem solving; applying information to produce some result; use of facts, rules and principles
- `Analysis`: subdividing something to show how it is put together; finding the underlying structure of a communication; identifying motives; separation of a whole into component parts
- `Synthesis`: creating a unique, original product that may be in verbal form or may be a physical object; combination of ideas to form a new whole
- `Evaluation`: making value decisions about issues; resolving controversies or differences of opinion; development of opinions, judgments or decisions

Anderson et al. [6] extends Bloom's taxonomy and pairs cognitive processes with knowledge to represent *competencies*. They introduce two different dimensions, the dimension of cognitive processes and the dimension of knowledge. Pairs of cognitive processes and knowledge elements form objectives (that constitute the basic building blocks of curricula). This was motivated by analyzing objectives listed in curricula, usually consisting of phrases such as "The student will learn to differentiate between rational numbers and irrational numbers" ([6], p. 5). They point out that such phrases typically are composed of a verb describing the intended cognitive process and one or more nouns referring to knowledge the students are supposed to acquire.

The competencies used for the PISA studies [8] include `think, argue, model, solve, represent, language, tools`. These top competencies have subcompetencies, e.g. `model` has the subcompetenices *decode* and *encode*.

The PISA competencies arise from the international discussion in mathematics education (with influence from OECD and NCTM). Approximatively, they represent learning objectives on a higher level. An advantage that is stressed is their independence of content and independence of students' age. The competencies are:

- `Think mathematically` includes the abilities to understand and handle mathematical concepts, their scope, and to understand and distinguish between different kinds of mathematical statements.
- `Argue mathematically` includes the abilities to develop and assess chains of arguments, to know what a mathematical proof is, to describe and reason about solutions, to uncover basic ideas in a line of arguments, and to understand reasoning and proof as fundamental aspects of mathematics.
- `Solve problems mathematically` includes the abilities to identify, pose and specify problems, to self-constitute problems, to monitor and reflect on the process of problem solving, to endue strategies and heuristics, and to solve different kinds of problems.
- `Model mathematically` includes the abilities to translate special areas and contents into mathematical terms, to work in the model, to interpret and verify results in the situational context, and to identify differences between the situation and the model.

- `Use mathematical representations` includes the abilities to understand and utilize different representations of mathematical objects, phenomena, and situations, to find relations between different representations, and to choose the appropriate representation for the special purpose.
- `Language` includes the abilities to use parameters, terms, equations and functions to model and interpret, to translate from symbolic and formal language into natural language and vice versa, and to decode and interpret mathematical language and understand its relations to natural language.
- `Communicate` includes the abilities to explain solutions, to use a special terminology, to work in groups, e.g. explain at an adequate level and understand and verify statements of others.
- `Use tools and aids` includes the abilities to know about the existence of various tools and aids for mathematical activities, their range and limitations, and to reflectively use them.

3 New Definition of the Competency Taxonomy

We define *elementary competencies* as pairs of a cognitive process and a knowledge element. We consider it a as a pair c of cognitive process p and a knowledge element k, $c = (p, k)$. Cognitive processes are defined as in [6] while knowledge elements represent facts, topics, concepts, theorems, rules/procedures and *Grundvorstellungen* [12] - i.e., elements of the knowledge dimension as available in ActiveMath and in the ontology of instructional objects OIO [10].

Composite competencies are defined as a set of multiple elementary competencies. In comparison to the work of van Assche [11], where a competency is defined as a tuple $c =< v, \{t_1, \ldots, t_n\} >$ (t_i are topics – our knowledge elements). Our definition, however, facilitates the representation within the learner model by reducing dimensionality without loss of expressivity.

3.1 Knowledge Ontologies / Extended Domain Ontologies

Knowledge elements can be related to each other, and therefore, a domain can be represented by an ontology. The knowledge includes concepts (e.g. fraction, integer, numerator), rules (e.g. addition of fractions with unlike denominators or subtraction), and *Grundvorstellungen*[3]. Grundvorstellungen in the fraction domain are part-whole, ratio, operator, quotient, and measure which provide different interpretations of a fraction in application contexts [13]. Other Grundvorstellungen exist for addition, multiplication and division of fractions too. The Grundvorstellungen have corresponding elements (nodes) in the educational domain ontology.

[3] We use the German term here since we could not find an appropriate translation for this term which was coined by German educationalists. A possible translations may be “interpretation/meaning of a concept”.

3.2 Ontology of Cognitive Processes

According to Flavell [15], meta-cognition is composed of meta-cognitive knowledge and meta-cognitive experiences or regulation. Meta-cognitive knowledge includes acquired knowledge about cognitive processes. Flavell divides meta-cognitive knowledge into three categories: knowledge of person variables, task variables and strategy variables. Correspondingly, Anderson et al. [6] differentiate between *self-knowledge*, *knowledge about cognitive tasks*, and *strategic knowledge*, and place these meta-cognitive aspects exclusively into the knowledge dimension.

emOur ontology of cognitive processes modifies and extends [6] by adding meta-cognitive processes that aim at representing the meta-cognitive regulation processes.

Although meta-cognitive competencies may be seen as more global/general proficiencies of a learner, the evidences always occur in a knowledge context whose influence has still to be investigated empirically. It may be that the ability to apply meta-cognitive processes varies depending on the proficiency in the respective domain and possibly even between different knowledge elements. Adhering to our definition of competency as pair of cognitive process and knowledge, the absence of meta-cognitive processes leads to meta-cognition related competencies, which can only express a global proficiency regarding meta-cognitive aspects and does not allow for a differentiation of how well a student is able to consciously apply meta-cognitive operations to specific knowledge elements or within a certain domain, e.g. questions such as "how proficient is the student in detecting errors within problems of the fraction domain?", cannot be answered. Therefore, we add meta-cognitive processes to the process dimension.

The first and second columns in Table 1 show the proposed hierarchy of the cognitive processes. Several cognitive processes are combined into categories:

- `Remember`: consists of the most basic retrieval operations performed on knowledge, i.e. the *recognition* of knowledge and its *recall* from memory.
- `Represent`: includes the abilities to *interpret* knowledge (e.g. "a fraction consists of two numbers: the numerator divided by the denominator"), to *illustrate* - to find an instance of a given concept, to *transform* from one representation to another, and to *summarize* (generalize) by inferring common principles or by identifying the main aspects of some information.
- `Solve`: includes the ability to *estimate* a result without calculating its exact value (e.g. estimate whether the addition of two fractions results in more or less than one), to *apply algorithm*s with all their steps, and to *apply tool*s appropriately (e.g. a calculator to add fractions).
- `Analyze`: summarizes abilities needed to break information into parts and to determine how these parts relate to each other and to the general picture. This category consists of the abilities to *check* information for inconsistencies or problems, to *differentiate* between important and unimportant information, to *organize* information according to some criteria, and to *attribute* a bias, value or intent to some presented material.

- `Model`: combines abilities needed to understand and create models in a specific domain. Included are the capability to *decode* information presented and transform it into a mathematical model (e.g. a textual description of ratios of some persons' ages, decoding could be putting the ratios into an equation), to *encode* a mathematical model into a situational context or its transformation into another domain. Furthermore, the category includes to *generate* hypotheses, and to *produce* new models by combining hypotheses to achieve a certain goal.
- `Communicate`: is concerned with explaining and discussing knowledge. It contains the processes of how to *describe* one's own knowledge, to *argue* about different aspects of some knowledge, and to *prove* certain facts.
- `Meta-cognition`: consists of meta-cognitive processes, i.e. processes that aim at reflecting and controlling cognitive processes. Such processes include to *reflect* upon one's own knowledge and thinking processes. Furthermore, *help seeking*, and to *search for information* to fill gaps in or extend one's own knowledge, to *detect errors* in one's own or the work of others, to *plan* tasks by dividing them into steps and order them according to their sequence of execution/implementation, to *self-monitor* one's own actions and behavior (e.g. by analyzing progress and differentiating between more effective learning strategies and less effective ones, and hence choose an appropriate one), and finally, to *self-explain*.

Some of the processes, such as *estimate*, could be placed into several of the categories, since they can be applied at different levels. The reason to place estimate into the `Solve` category is a specific interpretation, i.e., in the sense of applying rules of thumb or heuristics in order to get an idea of what the actual result may be. Alternatively, *estimate* may be interpreted as the process of ordering some aspects and inferring certain points.

4 Usage of the Ontologies

From the beginning, knowledge elements from domains have been characterized by metadata in ActiveMath. Exercises are annotated with metadata that specify which concepts they train and what cognitive processes are involved in attaining the correct solution. Cognitive processes have been present in ActiveMath as metadata for exercises, exercise steps and other learning objects. The metadata scheme evolved over time:

In the first version of the ActiveMath platform [16], Bloom's taxonomy was used. Later, we introduced the PISA-competencies for the LeActiveMath application of the ActiveMath-platform. The knowledge dimension is implicitly defined in the content and its metadata. This revision of competencies was driven by the influence of the pedagogical partners involved in the LeActiveMath project. Currently, ActiveMath relies on the competencies described above.

4.1 Student Modeling

ActiveMath's student model [17] can be parameterized to use different competency taxonomies. The structure of the student model includes nodes k_i for knowledge components. In case of using the proposed taxonomy, it stores pairs of cognitive processes and (domain) knowledge. This is done by relating each knowledge (super-)node k_i to a cognitive process p_j. The knowledge elements are dynamically extracted from the learning objects included in the content.

The student model derives competency values from evidences: exercises (whose metadata specify the knowledge and cognitive processes needed) and the respective performance/action of the student. Exercises-to-concept and concept-to-concept relations are dynamically extracted from the learning content in order to derive the competencies that have to be updated. In case of a competency involving multiple knowledge elements, evidences about proficiencies are equally attributed to all elementary competencies, as long as no further information is available to differentiate the attribution to a specific competency.

Evidences can be propagated along *prerequisite relations* in the student model. The propagated evidences are regarded as indirect evidences and are overridden as soon as direct evidence is available. In order to estimate proficiencies at a more general level, the hierarchical nature of the competency taxonomy is exploited, so that a more general competency reflects how the subcompetencies are mastered (and to what degree).

4.2 Selection of Content

Based on the estimations of the student model, the Tutorial Component selects appropriate learning objects, which serve the improvement of goal competencies [18]. One of the limitations of adaptively assembling courses is the availability of learning objects with perfectly matching knowledge and cognitive processes metadata. However, if no learning object is found that has exactly the metadata requested by the Tutorial Component, the hierarchical structure of both ontologies can be used to relax the search for learning objects and select a learning object that serves the training of a more general (or similar) cognitive process and knowledge.

Similarly, the two hierarchies (for knowledge and for cognitive processes) can be employed to facilitate mappings of exercises from different learning object repositories for course generation [19].

The relaxed search for knowledge and cognitive process metadata can lead to *approximate* mappings. Additionally, approximate mappings can be used for the alignment of different curricula as proposed in [11].

5 Relation to Existing Competency Taxonomies

One of the goals of the proposed competency taxonomy is to subsume and integrate existing competency systems, such as the PISA competency hierarchy and

the revised Bloom taxonomy. Therefore, we define (partial) mappings between our cognitive process hierarchy, the revised Bloom taxonomy and the PISA competency hierarchy.

Table 1 relates our process hierarchy to the PISA (mathematics) competencies and to cognitive processes in [6]. In some cases, finding a corresponding PISA-competency is rather difficult. One reason for the difficulty is that the PISA-competencies often include multiple processes in a single competency and sometimes also elements that we would place into the knowledge dimension. Thus, for the mapping, we compared the aspects of PISA competencies regarding the actual cognitive processes involved. Elements that are placed into parentheses capture only part of the scope of a proposed cognitive process.

Table 1. Mappings of the proposed cognitive processes to revised Bloom and PISA

Proposed category	**Proposed process**	**PISA**	**Revised Bloom**
Remember	Recognize	n/a	Recognize
	Recall	n/a	Recall
Represent	Interpret	Represent/Think	Interpreting
	Exemplify	Represent	Exemplifying
	Transform	Represent	Interpreting
	Summarize	Think	Summarizing
Compare	Find commonalities	Think	Comparing
	Find differences	Think	Comparing
	Classify	Think	Classifying
	Infer	Think	Inferring
	Order	Think	Comparing
Solve	Estimate	Solve	Inferring
	Apply algorithm	Solve	Executing
	Apply tool	Tools	n/a
Analyze	Check	Solve	Checking
	Differentiate	Think/Model	Differentiate
	Organize	n/a	Organize
	Attribute	(Think/Argue)	Attribute
Model	Decode	Model	Interpreting
	Encode	Model	Interpreting
	Generate	Model/Think	Generating
	Produce	n/a	Producing
Communicate	Describe	Communicate	Explaining
	Explain	Communicate	Explaining
	Critique	Argue	Critiquing
	Prove	Argue	Inferring
Meta-cognition	Reflect	(Solve/Model)	n/a
	Help seeking	n/a	n/a
	Search for information	n/a	n/a
	Detect errors	(Solve)	Checking
	Plan	(Solve)	Planning
	Self-monitor	(Solve)	Checking
	Self-explain	(Argue/Communicate)	(Explaining)

Paquette [9] (kindly brought to our attention by a reviewer) presents a top-level ontology for competencies, and a competency taxonomy pairing *generic skills* with *ressources*, corresponding to cognitive processes and knowledge elements respectively. His taxonomy combines and extends several previous approaches such as Bloom's. A mapping to the original approaches is provided.

6 Future Work

Since the proposed competency taxonomy leads to a fine granularity of competencies, ways to exploit the hierarchy of the taxonomy need to be developed in order to overcome the problem of sparse data. Therefore, we plan to explore the influence of the hierarchy on learner model estimations by comparing estimations with real students' performance. The additionally inferred data may provide a means for fine-grained and accurate estimations and, e.g. provide teachers with detailed information about students' weaknesses and strengths enabling them to revise their courses, e.g. to add an extra repetition of a poorly understood topic.

7 Conclusion

Most previous ITS have only used the knowledge dimension for characterizing their exercises and exercise steps as well as for building a structure for student models.

In order to make ITS-content and its metadata reusable and systems interoperable on learning objects, we propose a framework for competencies that can be used across domains and for many ITSs. This ontology framework includes two taxonomies, one for (domain) knowledge and one for cognitive processes. We define a taxonomy for cognitive processes which extends and modifies the taxonomy of Anderson et al. and we also extend the types of knowledge included in domain ontologies by Grundvorstellungen which are an important ingredient for real world problems.

We compare and translate several well-known competency taxonomies with the new ontological framework and briefly indicate why the hierarchical, two-dimensional framework is useful.

Acknowledgement

This publication is a result of the ATuF project (contract ME 1136/5-1) funded by the German National Science Foundation (DFG).

References

[1] Croteau, E., Heffernan, N., Koedinger, K.: Why are algbra word problems difficult? using tutorial log files and the power law of learning to sleect the best fitting cognitive model. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 240–250. Springer, Heidelberg (2004)

[2] Martin, B., Mitrovic, A.: Using learning curves to mine student models. In: Ardissono, L., Brna, P., Mitrović, A. (eds.) UM 2005. LNCS (LNAI), vol. 3538, pp. 79–88. Springer, Heidelberg (2005)
[3] Pardos, Z., Heffernan, N., Anderson, B., Heffernan, C.: The effect of model granularity on student performance prediction using bayesian network. In: User Modeling. LNCS (LNAI), pp. 435–443. Springer (2007)
[4] Narciss, S.: Informatives Tutorielles Feedback. Habilitationsschrift, Technische Universität Dresden, Fak. Mathematik und Naturwissenschaften, Dresden (Mai (2004)
[5] Bloom, B. (ed.): Taxonomy of Educational Objectives: The Classification of Educational Goals: Handbook I, Cognitive Domain, Longmans, Green, Toronto (1956)
[6] Anderson, L., Krathwohl, D., Airasian, P., Cruikshank, K., Mayer, R., Pintrich, P., Wittrock, M.: A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxomnomy of Edicational Objectives. Longman, New York (2001)
[7] Pardos, Z., Heffernan, N., Anderson, B., Heffernan, C.: Using fine-grained skill models to fit student performance with bayesian networks. In: Educational Datamining Workshop, at ITS 2006, pp. 5–12 (2006)
[8] OECD: Learning for Tomorrows World – First Results from PISA 2003. Organization for Economic Co-operation and Development (OECD) Publishing (2004)
[9] Paquette, G.: An ontology and a software framework for competency modeling and management. Educational Technology & Society 10(3), 1–21 (2007)
[10] Ullrich, C.: The learning-resource-type is dead, long live the learning-resource-type!. Learning Objects and Learning Designs 1(1), 7–15 (2005)
[11] Assche, F.V.: Linking content to curricula by using competencies. In: Massart, D., Colin, J.N. (eds.) 1^{st} International Workshop on Learning Object Discovery & Exchange (LODE 2007), Crete (2007)
[12] Malle, G.: Grundvorstellungen zu Bruchzahlen. mathematik lehren 123, 4–8 (2004)
[13] Kieren, T.: On the mathematical, cognitive, and instructional foundations of rational numbers. In: Lesh, R. (ed.) Number and Measurement: Papers from a Research Workshop, Columbus, OH, ERIC/SMEAC, pp. 101–144 (1976)
[14] Charalambous, C., Pitta-Pantazi, D.: Drawing on a theoretical model to study students understandings of fractions. Educational Studies in Mathematics 64(3), 293–316 (2007)
[15] Flavell, J.: Metacognition and cognitive monitoring: A new area of cognitive-developmental inquiry. American Psychologist 34, 906–911 (1979)
[16] Melis, E., Goguadze, G., Homik, M., Libbrecht, P., Ullrich, C., Winterstein, S.: Semantic-aware components and services of activemath. British Journal of Educational Technology 37(3), 405–423 (2006)
[17] Faulhaber, A.: Building a new learner model for activemath combining transferable belief model and item response theory. Master's thesis, Saarland University (2007)
[18] Ullrich, C.: Adaptive Course Generation Service Based on Hierarchical Task Network Planning. PhD thesis, Universität des Saarlandes, Department of Computer Science, Saarbrücken, Germany (2007)
[19] Kärger, P., Ullrich, C., Melis, E.: Integrating learning object repositories using a mediator architecture. In: Nejdl, W., Tochtermann, K. (eds.) EC-TEL 2006. LNCS, vol. 4227, pp. 185–197. Springer, Heidelberg (2006)

Comparing Classroom Problem-Solving with No Feedback to Web-Based Homework Assistance

Leena Razzaq[1], Michael Mendicino[2], and Neil T. Heffernan[1]

[1] Worcester Polytechnic Institute
Worcester, Massachusetts 01609
{leenar,nth}@cs.wpi.edu
[2] West Virginia University
Morgantown, West Virginia 26506
mmendic1@mail.wvu.edu

Abstract. It is common for math teachers to give students time in class to practice problem solving skills. Some studies have shown that intelligent tutoring systems (ITS) can be superior to traditional classroom instruction. In this study, we compare problem solving with little or no feedback in the classroom to problem solving using a web-based ITS for homework. The system provides students with coached practice that is meant to scaffold "learning by doing" while students practice their problem solving skills. We found evidence that using the web-based ITS to practice problem solving at home was better than the classroom problem solving with an effect size of 0.5.

Keywords: Problem-solving, intelligent tutoring systems, symbolization, web-based homework, word problems.

Introduction

Many studies have shown that learning can be improved by learning technologies with varying effect sizes depending on the system and the metrics used to measure learning. It has been shown that traditional computer-assisted instructional systems (CAI) can lead to better learning when compared to traditional classroom instruction [1, 2]. Kulik & Kulik's [1] studies indicate that CAI systems can lead to about 0.3 to 0.5 standard deviation effect sizes over classroom instruction and suggests that classrooms using CAI systems can learn more and learn faster than classrooms using traditional classroom instruction alone.

Evidence of the benefits of newer intelligent tutoring systems (ITS) over classroom instruction exists as well [3, 4]. Koedinger et al. [4] compared a commercially available ITS (Cognitive Tutors) to a classroom control and suggested a 1.0 standard-deviation effect size for experimenter-designed metrics, while for external metrics (The Iowa Algebra Aptitude test and a subset of the Math SAT) the study found an effect size of 0.3. This study may suffer from a confound of the effect of the ITS with a new textbook prepared to go along with the curriculum. It is unclear how to compare these effect sizes with the Kulik & Kulik [1] effect size of about 0.4 as we don't know if the metrics in the Kulik & Kulik studies are more generally like externally

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 426–437, 2008.

designed measures or experimenter defined measures. In another study, VanLehn et al. [5] compared an ITS not to classroom instruction, but to doing homework in a traditional paper-and-pencil manner. They found results similar to the Cognitive Tutor results mentioned above with effect sizes of about 1.0 standard deviation for their own measures, and about 0.4 for what they consider analogue to externally designed measures. Additionally, other studies have shown that ITS can produce superior learning results when compared to CAI systems [6, 7, 8].

It is common for math teachers to give students time in class to practice problem solving skills where they receive little or no feedback on their work. We question whether it is more effective to instead do web-based intelligent tutoring homework to practice problem-solving skills. In this study, we compare two conditions: problem-solving practice in the classroom and problem-solving practice for homework using an ITS. The ITS, called Ms. Lindquist, was developed by Heffernan & Koedinger [9] and is described in the next section.

We conduct this study in a mathematics classroom while teaching the skill of writing algebra expressions for word problems, a skill we call symbolization. We report on an experiment, with one teacher and a total of 28 students. The study involved analyzing the amount of learning gains by students as measured by experimenter designed pre- and post-tests the days before and after the treatments.

Web-Based Homework Assistance

Web-based systems that allow students to do their homework online such as Blackboard (www.blackboard.com), WebCT (www.webct.com), Homework Service (https://hw.utexas.edu/bur/overview.html) and WeBWorK (http://webwork.rochester.edu) are becoming widely used at the college level. Use of homework assistance systems at the K-12 level, such as Study Island (studyisland.com) and PowerSchool (powerschool.com), is less common, but they are gaining popularity among teachers.

Advantages of web-based homework assistance systems are immediate feedback to students and automatic grading for instructors. Automatic grading can be helpful to teachers by saving time for those who do not have time to grade all of their students' paper-and-pencil homework carefully by hand, which in turn can prompt students to take homework more seriously because they know it will be graded and the grade will be recorded. Students can get immediate feedback on their answers to problems and sometimes hints or intelligent help towards solving problems.

Although there are benefits to using these web-based homework assistance systems, there can be disadvantages, as well. Many of these systems require students to enter a single answer for each problem and they do not consider or take note of students' work. Students may also try to do more math in their heads and do less scrap work which can help them to be more organized. Teachers may spend less time looking at their students' work and figuring out exactly where they are having difficulties. Because these systems often do not consider student work, it may be easier for students to cheat. Additionally, the digital divide, which may be more prevalent at the K-12 level than at the college level, could also prevent teachers from taking advantage of web-based homework assistance.

Research has shown positive results for using intelligent tutors instead of doing traditional paper-and-pencil homework. "MasteringPhysics" is a web-based physics homework tutor developed at MIT that uses mastery learning to help students reach mastery when solving physics homework problems. Students can request hints on problems and can receive feedback on common student errors. Warnakulasooriya & Pritchard [10] found that twice as many students were able to complete a set of problems in a given time with the help provided with MasteringPhysics when compared to students that worked on the problems without help (administered by MasteringPhysics but without hints or feedback).

Quantum Tutors (http://www.quantumsimulations.com/) is a web-based system that is commercially available for students to do homework in the sciences and math. Students can choose topics to work on, enter their own problems and choose from a list of questions they may have on particular problems. For instance, students working on percents can choose "I need to find the percentage one number is of another" and solve problems provided by the system or enter their own values to solve. They can also choose from a list of questions such as "Why would I want to convert a percent to a fraction?" In a press release, Quantum Tutors describes a week-long study done in 2005 (http://www.quantumsimulations.com/news15.html), where students using Quantum Tutors for homework in a high school chemistry course outperformed a control group that did paper-and-pencil homework on a post-test by just over a full letter grade. The difference between groups became larger as the problems increased in difficulty.

The previously mentioned VanLehn work [5], the Andes system, is an intelligent tutoring system that provides support for problem-solving for physics homework. Andes requires students to complete whole derivations step-by-step and offers feedback after each step. Students can also ask for hints on each step to find out the nature of their errors (What's Wrong Help) or to get help on what the next step is (Next Step Help). Andes was used and evaluated in introductory physics classes from 1999 – 2003 at the U.S. Naval Academy. VanLehn et al. [5] presented evidence that students who used Andes for homework got significantly higher exam scores than students in control groups who did paper-and-pencil homework. Other studies of web-based physics homework vs. paper-and-pencil homework did not find significant differences between the two [11, 12].

Ms. Lindquist. Heffernan and Koedinger [9] developed an ITS, called Ms. Lindquist, which uses dialog to help students write algebra expressions by modeling both student behavior and tutorial behavior. Ms. Lindquist combines a cognitive model of student behavior in the domain of *symbolization* with a tutorial model of strategies observed in a human tutor. The cognitive student model has a set of production rules that models the problem solving skills needed to write algebraic expressions. The tutorial model is based on the observation of an experienced human tutor during an hour-long tutoring session and thus captures tutorial strategies that were observed to be effective for human tutors. The tutor, a middle school mathematics teacher with four years of mathematics teaching experience, had two years of one-on-one tutoring experience as well (through both University tutoring centers and private tutoring). This tutor charged clients 40 dollars an hour. The tutor worked with a seventh grade student whom she had not previously tutored on a list of symbolization problems. The session was recorded on video and then transcribed. Strategies that the tutor used often and

Table 1. An experienced tutor's transcript and Ms. Lindquist's attempt to model it

Experienced Human Tutor Exact Transcript	The Ms. Lindquist Tutor Similar simulated input.
240. Student: [reads the "bike-trip" problem] Cathy took a "m" mile bike ride. She rode at a speed of "s" miles per hour. She stopped for a "b" hour break. Write an expression for how long the trip took.	T1: [The "bike-trip" problem was presented.] Please write an expression for the total time the trip took.
241. S: um … [Writes "s/m+b"]	S1: s/m+b
242. Tutor: How do you calculate the amount of time it takes you? If you're, if you're, if you're riding at … Let's make it simple. If you are riding at 20 miles per hour, OK, and you go 100 miles, how many hours did that take you?	T2: No. Let me ask you an easier question. Please compute the time actually on the bikes if the speed was 20 miles per hour, and the distance ridden was 100 miles?
243. S: Um 5	S2: 5
244. T: 5. And how did you get that 5? How did you use the numbers 100 and …? [trails off]	T3: Correct. What math did you do to get that five?
245. S: 100 miles divided by miles per hour.	S3: 100/20
246. T: So you took the miles and divided it by the [garbled, but probably "speed"]	T4: Correct. Now write your answer of 100/20 using the variables given in the problem (i.e., put in 'm' and 's')?
247. S: Miles divided by 's' plus 'b' equals time. [writes m/s+b]	S4: b+m/s
248. T: Right.	T5: Correct.

that were easy to implement were chosen to implement in Ms. Lindquist's tutorial model. Table 1 shows a portion of the human tutor's transcript and Ms. Lindquist's attempt to model it.

Ms. Lindquist was the first intelligent tutor that had both a model of student thinking and a model of tutorial planning. The system is of the "coached practice" variety that is meant to scaffold "learning by doing" while students practice their problem solving skills. It is different from typical Cognitive Tutors [4] in that it takes its cues more from the dialogs that human tutors have with students and is more flexible in the interactions it has with students. For example, it can acknowledge that part of an answer is correct and then engage a student in a "sub dialog" to help him or her to improve the incorrect path. It "breaks" problems down for students by asking questions

and re-phrasing questions, but does not give students answers. Whereas Cognitive Tutors typically cover a full-year of algebra, Ms. Lindquist deals only with symbolization problems.

Ms. Lindquist was evaluated in several experiments [13] that showed positive learning results for the ITS. For instance, one analysis focused on 76 middle school students who used Ms. Lindquist as part of a class assignment. The students in the experimental condition received one of Ms. Lindquist's tutorial strategies when they needed help. The students in the control condition were simply told the answer if they answered incorrectly and moved on to the next problem. This experiment controlled for time. The interaction between condition and learning gain was statistically significant with an effect size of 0.56 standard deviations in favor of Ms. Lindquist, even though the students in the control group did significantly fewer problems than those in the experimental group.

Fig. 1 shows a screenshot of Ms. Lindquist tutoring a student on a symbolization problem. Ms. Lindquist can be found at algebratutor.org where students and educators can log in and be tutored on symbolization. Tutoring strategies are randomly chosen for each student.

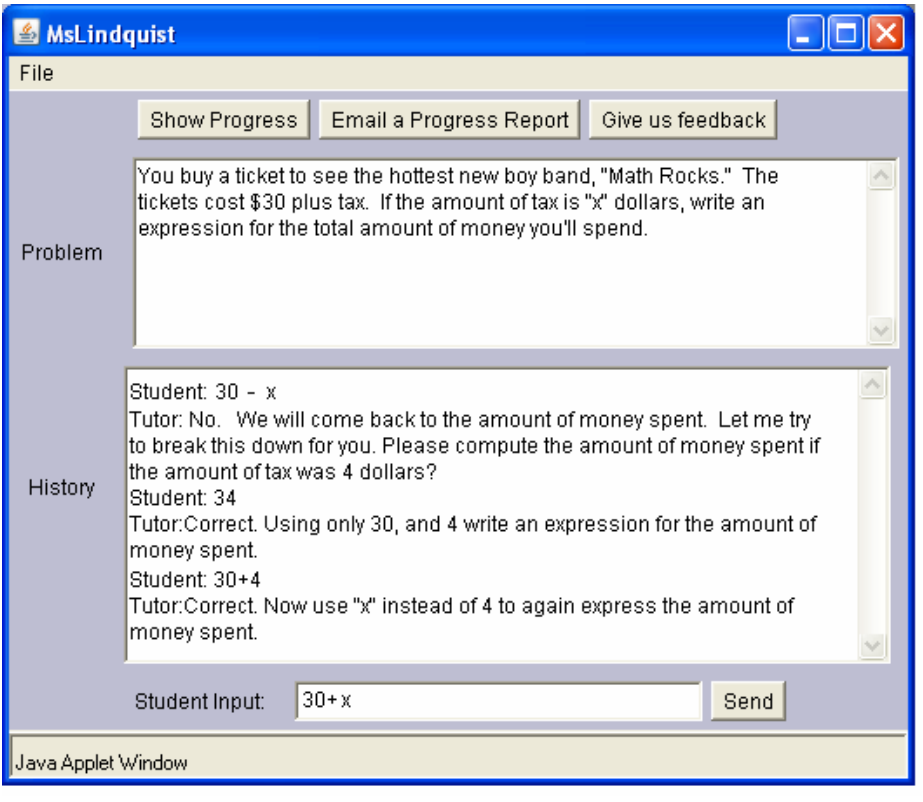


Fig. 1. A screenshot of Ms. Lindquist

Method

Our research question for this experiment is:

Research Question. For students who have Internet connections at home, does 30 minutes of classroom problem-solving do better or worse, in terms of student learning gains than ~30 minutes of homework problem solving using web-based homework assistance.

Measures of interest included student learning gains measured in school, gains with the computer system itself, time on task, and student reported satisfaction with the computer system.

Setting and Participants. The setting for this study was two regular algebra classrooms in a high school in West Virginia, and students' home computers. Students in both classes were offered extra credit to do the experimental condition as a homework assignment. Obviously, only students with Internet access could participate in the study.[1]

Fourteen students from each class agreed to participate in the experimental condition which meant they agreed to work at home, for at least thirty minutes, on the web-based system. Thus, the participants for this study were twenty-eight students (20 female, 8 male, ages 14-16 years) out of a possible 45 students from both classes. All students were classified as typically achieving students; that is, none were identified as learning disabled. The second author taught both classes during the experiment and while he was not the students' regular math teacher, he is a highly qualified, math through Algebra 1, special education teacher and was well-known to the students.

The students had been introduced to the topic of symbolization approximately a month before this study; they were not studying this topic in parallel with the experiment. This topic was chosen because it is the only topic that Ms. Lindquist teaches.

Table 2. Overall experimental design

Day	Time	Group 1	Group 2
1	~ 10-20 minutes	Pretest / Introduction to Computer-system	Pretest / Introduction to Computer-system
2	30 minutes	Classwork	Web-based Homework
3	~ 10-15 minutes	Mid test	Mid test
4	30 minutes	Web-based Homework	Classwork
5	~ 10-15 minutes	Post-test	Post-test

[1] This was not a planned circumstance, as the original plan was to have students use computer labs in school. However, because the school recently installed new security software that prevented the web site from functioning correctly, the instructor instead sought volunteers to engage in the computer condition at home for extra class credit. Unfortunately, this excluded students who did not have internet at home from the study.

Design. A counterbalanced design was used in which all groups received all conditions, but in a different order. Specifically, one group participated in the *classwork* condition first while the other group participated in the *homework* condition first, thus ensuring a different sequence of instruction for each group.

A pretest was administered to both groups to ensure initial comparability on the dependent measures. The students were given a mid-test after participating in the first condition (according to which group they were in) and a post-test after completion of the experiment. The data for this study were analyzed using SPSS. Analysis of Variance, t-tests and descriptive tests were used. Table 2 displays the overall design of the study.

Procedures. The study was conducted over a five day period and included a pretest, mid-test, and post-test administered on days one, three and five of the experiment, before and after each condition. The pretest, mid-test and post-test all contained the same nine problems and were administered on paper. The questions on these tests were symbolization problems similar to those worked on in the *classwork* and *homework* conditions. The following problems are samples of those that appeared on these tests:

1) Mary starts a car washing business at the local gas station. She spends $30 to buy supplies. She then charges $10 to wash a car and scrub clean all four tires. Write an expression showing how much money she has made after she has washed "c" cars.
2) Mary's mom bought her a CD player that cost $200. She has to pay her mom $15 per month until it is paid off. Write an expression for the amount she owes her mom after "n" months.
3) Martin got a Christmas bonus at work that was worth 'b' dollars. He first paid the rent which was 'r' dollars. He then split the remaining money between his three children. How much money did each child get?

The experimental conditions occurred on days two and four in a counterbalanced manner. Group 1 participated in the *classwork* condition on day two and did the *homework* condition on day four, while Group 2 did the *homework* condition on day two and the *classwork* condition on day four. For the *homework* condition both groups were taught how to create an account and log on to the system and were instructed to spend at least thirty minutes on the computer system from the time they were logged on without stopping.

To be clear, students who did not volunteer to do the extra homework were not in the classroom; students who did volunteer to do the extra homework required were "pulled out" of their normal classroom for the *classwork* part of this experiment. Obviously, this was a more motivated group of students.

The *classwork* activities were divided into two main parts: 1) introduction with in-class examples, and 2) guided practice. Students were given a worksheet with twenty-five problems. The instructor demonstrated how to translate word problems into algebraic expressions by first displaying problems on an overhead projector and reading the problems to the class. The instructor then discussed several traditional textbook

methods used to translate word problems to algebraic expressions including matching "clue" words with mathematical operations and procedures and using problem-solving plans such as: explore the problem, plan the solution, solve the problem, and examine the solution. The instructor demonstrated five problems that took approximately twenty minutes. During the remaining thirty minutes of the class, students completed their worksheets and the instructor was available to all students and assisted them in the order in which students requested help. The students did not get feedback on every problem and those who did not request help got no feedback at all. Some of the problems in the worksheet are as follows:

1) Missy starts a business selling fishing bait. She spends $40 buying supplies like containers, bags, and minnows. She sells a container of bait for $3. If she sells "c" containers of bait, how much profit will she end up making?
2) Aunt Bee won "d" dollars in the lottery. She spent $40 on groceries and spilt the rest up in presents for her six children. How much did each child get?
3) Jane is "j" years old. Peg is "p; years old. Mary is Peg's age minus Tom's age. Tom is 4 years younger than Jane. Peg is "x" years older than Mark. Write an expression for Mary's age.

Students in the *homework* condition were asked to log in to Ms. Lindquist and work for at least 30 minutes on symbolization problems. However, this experiment does not control for time since we could not control how much time students spent on their homework.

Results

We assume that the groups were fairly balanced since we found no statistically significant difference ($t = -0.655$, $p < 0.518$) at pretest between groups. The mean pretest score for the computer first group was ($m = 3.86$, $sd = 1.29$) and for the computer second group, the mean pretest score was ($m = 4.21$, $sd = 1.58$).

Overall, students showed large learning gains ($F = 6.58$, $p < 0.016$) as measured by repeated-measures ANOVA. For both groups, we looked to see if the pretest to posttest gains were reliably different from zero, and in both cases they were, suggesting that students learned from doing *classwork* and from doing web-based *homework*.

When comparing web-based *homework* gains and *classwork* gains, we found statistically significant differences ($t = 2.044$, $p = 0.051$) in favor of the web-based homework condition ($m = 1.41$, $sd = 1.00$) over the classwork condition ($m = 0.8707$, $sd = 0.80$), suggesting that students achieved a one-half problem learning gain from the computer condition overall. The effect size for this difference was 0.54 with confidence intervals of -0.12 – 0.94. We ignore the order in which students participated in each condition in this analysis (the order of condition is considered in the next analysis shown in Table 4). The results of this analysis are summarized in Table 3.

Table 3. Summary of results of comparing classwork to web-based homework

N	effect size	p value	Mean classwork gain score	Mean homework gain score
28	0.54	0.051	0.8707	1.41

Finally, we looked to see if there was a difference in learning gains when taking into account the binary factors of order and computer condition. We found that for both groups (computer first and computer second) the difference was not statistically significantly different ($F = 2.508$, $p < 0.126$). These results are summarized in Table 4.

Table 4. The Average Learning Gain split out by whether the students did the web-based homework first or second. (Dependent Variable: Computer Gain)

Group	N	Mean	Standard Deviation
Overall	28	1.3571	1.0261
Computer first	14	1.5000	0.94054
Computer second	14	1.2143	1.1217

Anecdotal Data on Motivation: Students were directed to work for at least 30 minutes on the web-based homework, however some of the computer log files were lost due to technical difficulties, so we were not able to determine whether all students did their homework or not. We were able to recover 70% of the log files and we have no reason to believe that the files that were lost would differ greatly from those recovered. From the recovered log files, we could see that students worked for an average of 25 minutes on Ms. Lindquist. Four students worked for over 30 minutes, one student for an hour and twenty minutes. We hoped that the students would be motivated and want to spend more time on the computer doing their homework and some did. However, the average time spent by both groups on problem-solving seems comparable and the *classwork* condition actually spent more time overall when we count the time spent by the instructor at the beginning of the class on the introduction and symbolization examples (50 minutes total).

There was evidence that students had a positive attitude toward the tutoring dialog and also thought it was helpful. For example, students were asked to respond to the following questions embedded within the computer-delivered instruction:

1) "Rate from 1-10 how much trouble you had on this section. Use 1 for easy and 10 for hard."
2) "Did you find the computer-delivered instruction helpful?"
3) "Before you quit, would you please give us feedback on Ms. Lindquist. Type your feedback anywhere in this window".

All but three students who did the web-based homework answered "yes" they thought the feedback was helpful and gave an average difficulty rating of 3.3. Several students answered question three with the following comments:

1) "I think this is a good program to help students in math and he helps them understand it better",
2) "It was fun and helped me understand it better",
3) "This program is a very good way to tutor people about algebra, but even though it tells you what you have done wrong it sometimes becomes frustrating therefore it may be a good idea to have the student write questions about specific problems that they may have because an actual person helping is always better than a computer program"
4) "This program is a significant program in teaching algebra. Some problems are difficult, but are possible to figure out".

Discussion

In this experiment, we found that students in both *classwork* and *web-based homework* conditions had learning gains between pre- mid- and post-tests. The web-based homework condition outperformed the classwork condition with an effect size of 0.54. Given that students in the web-based homework condition learned more than students doing problem-solving practice in class, perhaps the ITS might well be beneficial in the context of assigned homework. However, we are not sure if our results are due to better "intelligent" pedagogy or to students receiving immediate feedback on their work.

We speculate that our results might be affected by the fact that we asked students to volunteer for this experiment and do extra credit work at home making this experiment's population probably more motivated (as indicated by being willing to do extra-credit work at home). Our results could also be affected by the fact that only students with access to computers and internet at home could participate in this experiment.

Some states in the U.S., such as Virginia, Maine, and Indiana are implementing "one-to-one computing" programs [14] in schools where each child gets his/her own laptop to use during school and sometimes they are allowed to take the laptops home, as well. In fact, the Maine Learning Technology Initiative (2002-2004) supplied every seventh and eighth grade student in Maine and their teachers with laptop computers, allowing 40% of the middle schools students to take their laptops home. While we await research studies on the effects of one-to-one computing on teaching and learning we have seen reports that students in one-to-one computing programs are more engaged, motivated and interact better with teachers [15, 16]. At the same time, widely published opposition cites the high cost, potential access to inappropriate material and lack of proven impact on student achievement [17, 18] as reasons to abandon one-to-one computing programs in schools. Even so, the numbers of U.S. schools that are adopting one-to-one computing programs continue to increase every year [18].

As the digital divide narrows and more states become committed to one-to-one computing programs, opportunities for students to do their homework online increase. We conducted another study comparing web-based math homework to paper-and-pencil math homework for fifth grade students which also showed favorable results for the web-based homework condition [19]. We think that implications of these studies, and others like them could be important to policy- makers, when considering

whether to adopt one-to-one computing programs (especially considering the falling price of laptops and claims that they can now be produced for prices as low as $200 each [20].)

Future Work. We believe that the results of this study may be affected by the fact that only some students could participate either because of the motivation to earn extra credit or because they did not have access to the web-based ITS. For future work, we would like to repeat this study with more students that are not limited to having computers and internet at home. To counter this, we would direct students who do not have computers at home to use the computers in the library or the school's computer lab after school. This way we would also be able to see if we could get similar results with students who are not as motivated as the students in this study were by requiring that all students in the class participate by doing their homework using a web-based ITS.

References

1. Kulik, C.C., Kulik, J.A.: Effectiveness of Computer-Based Instruction: An Updated Analysis. Computers in Human Behavior 7, 75–94 (1991)
2. Kulik, J.A.: Meta-Analytic Studies of Findings on Computer-Based Instruction. In: Baker, E.L., O'Neil Jr., H.F. (eds.) Technology Assessment in Education and Training, pp. 9–33. Erlbaum Associates, Hillsdale (1994)
3. Anderson, J.R., Corbett, A., Koedinger, K., Pelletier, R.: Cognitive Tutors: Lessons Learned. Journal of the Learning Sciences 4(2), 167–207 (1995)
4. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A.: Intelligent tutoring goes to school in the big city. In: Proceedings of the 7th World Conference on Artificial Intelligence in Education, Association for the Advancement of Computing in Education, Charlottesville, pp. 421–428 (1997)
5. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The Andes physics tutoring system: Lessons Learned. International Journal of Artificial Intelligence and Education 15(3), 1–47 (2005)
6. Carroll, J.M., Kay, D.S.: Prompting, feedback and error correction in the design of a scenario machine. International Journal of Man-Machine Studies 28, 11–27 (1988)
7. Corbett, A., Anderson, J.R.: Locus of feedback control in computer-based tutoring: Impact on learning rate, achievement and attitudes. In: Jacko, J., Sears, A., Beaudouin-Lafon, M., Jacob, R. (eds.) Proceedings of ACM CHI 2001 Conference on Human Factors in Computing Systems, pp. 245–252. ACM Press, New York (2001)
8. Mathan, S., Koedinger, K.R.: Recasting the feedback debate: Benefits of tutoring error detection and correction skills. In: Hoppe, Verdejo, Kay (eds.) Artificial Intelligence in Education, Proceedings of AI-ED 2003, pp. 13–18. IOS Press, Amsterdam (2003)
9. Heffernan, N.T., Koedinger, K.R.: An Intelligent Tutoring System Incorporating a Model of an Experienced Human Tutor. In: International Conference on Intelligent Tutoring System 2002, Biarritz, France, pp. 596–608 (2002)
10. Warnakulasooriya, R., Pritchard, D.E.: Learning and problem-solving transfer between physics problems using web-based homework tutor. In: Conference Proceedings: EdMedia -World Conference on Educational Multimedia, Hypermedia & Telecommunications, pp. 2976–2983 (2005)

11. Pascarella, A.M.: CAPA (Computer-Assisted Personalized Assignments) in a Large University Setting. Unpublished Doctoral Dissertation, University of Colorado, Boulder, CO (2002)
12. Bonham, S.W., Deardorff, D.L., Beichner, R.J.: Comparison of student performance using web and paper-based homework in college-level physics. Journal of Research in Science Teaching 40(10), 1050–1071 (2003)
13. Heffernan, N.T., Croteau, E.: Web-Based Evaluations Showing Differential Learning for Tutorial Strategies Employed by the Ms. Lindquist Tutor. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) Proceedings of 7th Annual Intelligent Tutoring Systems Conference, Maceio, Brazil. LNCS, pp. 491–500. Springer (2004)
14. Bonifaz, A., Zucker, A.A.: Lessons learned about providing laptops for all students. Education Development Center, Newton (2004), http://www.neirtec.org/laptop Retrieved October 15, 2007
15. Silvernail, D.L., Lane, D.M.M.: The impact of Maine's one-to-one laptop program on middle school teachers and students (Report #1). Gorham, ME: Maine Education Policy Research Institute, University of Southern Maine Office (2004)
16. Bebell, D.: Technology promoting student excellence: an investigation of the 1st year of 1:1 computing in New Hampshire middle schools. Technology and Assessment Study Collaborative (2005, May) Retrieved from November 20 2007, http://www.intasc.org
17. Vascellaro, J.E.: Saying no to school laptops. The Wall Street Journal (August 31, 2006), Retrieved November 20, 2007 from, http://online.wsj.com/article_email/SB115698378733250090-lMyQjAxMDE2NTM2MTkzODEzWj.html
18. Hu, W.: Seeing No Progress, Some Schools Drop Laptops. New York Times (May 4, 2007), Retrieved November 25, 2007 from, http://www.nytimes.com/2007/05/04/education/04laptop.html
19. Mendicino, M., Razzaq, L., Heffernan, N.T.: A Comparison of Traditional Homework with Computer Supported Homework. Journal of Research on Technology in Education (accepted)
20. Bray, H.: $100 laptops? Not really, but $200 isn't bad. Boston Globe (November 19, 2007), Retrieved November 20, 2007 from, http://www.boston.com/business/technology/articles/2007/11/19/100_laptops_not_really_but_200_isnt_bad/

Harnessing Learner's Collective Intelligence: A Web2.0 Approach to E-Learning

Hicham Hage and Esma Aïmeur

Département d'informatique et de recherche opérationnelle
Université de Montréal
{hagehich,aimeur}@iro.umontreal.ca

Abstract. Today, learners have access to a wide range of sources where they regularly search for, find and use learning resources outside the scope of regular course material. Although E-learning systems offer a variety of tools and functionalities, there are no specific provisions for learners to easily store and share these valuable resources. In this work we present SHAREK, a Web2.0 inspired approach to allow learners to store and share their resources. Specifically, SHAREK combines Artificial Intelligence techniques, such as Recommendation Systems and Information Retrieval with Web2.0 technologies, including RSS and tagging to allow easy sharing of resources and knowledge. Finally, we report on the implementation and validation of SHAREK.

Keywords: Web2.0, Collaborative Learning, Knowledge Sharing.

1 Introduction

E-learning emerged over 20 years ago, and consisted solely of text like a book on a screen, and was ineffective and unpopular with learners. Today, E-learning has become richer with multimedia content and more interactive. With E-learning, education is shifting from being Tutor Centered where the tutor is the center and has access to the resources, and becoming more *Learner Centered* [1] where the student is the center and the focus of the learning process and has access to a multitude of resources. Although learner centered education is not a novel idea, E-learning and ITS (Intelligent Tutoring Systems) [2] are major contributors to the development and advancement of learner centered education.

As such, and with the availability of a multitude of resources (the World Wide Web, forums, Wikis, libraries, …), learners regularly search for, find and use learning resources, other than the regular class material. Indeed, in a recent survey we conducted (further details in section 4), when asked how often they refer to resources external to the course material, 93% of the students said they do so regularly.

Nonetheless, these precious resources are discarded after their use. Such resources and knowledge are important and valuable to other learners. Within the same survey, when asked how probably they would share these resources with classmates and friends, again 93% of the students were favorable to sharing. In current E-learning and ITS systems, there are provisions for collaborative learning [3, 4]. Systems such as SPRITS [5], Comtella [6] and iHelp [7] promote knowledge and resource sharing.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 438–447, 2008.

iHelp and SPRITS are designed for sharing knowledge and Comtella for sharing academic articles, yet learners access a wider variety of resource types. As such, we introduce SHAREK (**SHA**ring **RE**sources and **K**nowledge) for learners to share their knowledge and resources. Inspired by the Web2.0 [8] approach at harnessing collective intelligence, SHAREK allows learners to augment the learning material proposed by the tutors by adding or attaching, to a course or lecture, learning resources that they have created or found. In contrast to existing systems, such as Comtella, iHelp or SPRITS, SHAREK aims to share various forms of knowledge (such as a report, a research paper, or an assessment) and resources (such as a video, a presentation, or an animation). Moreover, SHAREK combines several Web2.0 techniques including RSS (Really Simple Syndication), tags, as well as Collaborative Filtering recommendation [9] and Information Retrieval [10] techniques to help manage, share and locate the shared resources within SHAREK.

The paper is organized as follows. Section 2 offers a brief introduction to Web2.0. Section 3 introduces SHAREK and details the components. Section 4 highlights the testing procedure and results. Section 5 concludes the paper and provides an overview of future work.

2 Web2.0

Although the term Web2.0 suggests a new version of the World Wide Web, it does not refer to an update or any technical specifications, but rather to changes in the ways software developers and end-users perceive and use the web. The term Web2.0 refers to a perceived second generation of web-based communities and hosted services (such as blogs, Wikis, etc.) which aim to facilitate creativity, and to promote collaboration and sharing between users. Table 1 [11] formulates a sense of what is Web2.0 by example:

Table 1. Web1.0 vs. Web2.0 [11]

Web1.0	*Web2.0*
Britannica Online	Wikipedia
Personal websites	Blogging
Publishing	Participation
Directories (taxonomy)	Tagging ("folksonomy")
Content Management Systems	Wikis

One of the driving principles of Web2.0 is harnessing collective intelligence. Indeed, in the context of Web2.0 users are not just recipients of information, but actively participate in the creation of such information, whether by building personal blogs, participating in Wikis, tagging, rating, sharing and/or referring websites.

A recently published report [12] indicates that 64% of online teenagers in the US, ages 12 to 17, engage in at least one type of content creation. Moreover, both YouTube and Wikipedia are listed among the top 10 most visited sites by Alexa [13]. Both sites rely heavily on user input. These are some of the encouraging indications

about the willingness of users to create and share information and knowledge, a major motivation behind SHAREK.

3 SHAREK (SHAring REsources and Knowledge)

To the best of our knowledge, there are no existing mechanisms in E-learning systems to *efficiently* harness and take advantage of learner's knowledge and the resources they locate and use. Supported by learner centered education, and the learners' access to a variety of learning resources, and inspired by Web2.0, we propose our system SHAREK. SHAREK's primary goals are to *harness* the collective intelligence and knowledge of learners obtained through accessing various learning resources, and *sharing* this knowledge and resources with other learners. Specifically, each time a learner accesses a course's learning content (specified by the tutor), he would be able to also access additional learning resources located and used by other learners and classmates. Fig. 1 illustrates the positioning of SHAREK within an ITS architecture based on LTSA [14].

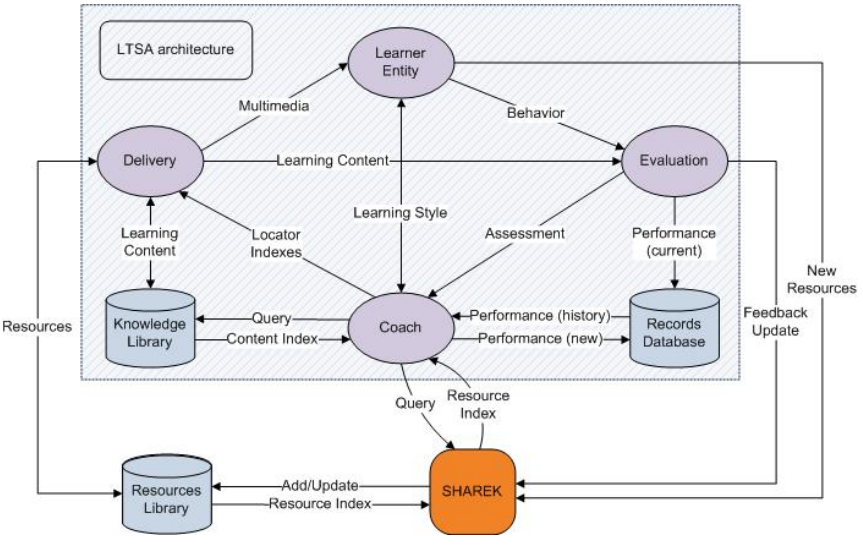


Fig. 1. SHAREK within an ITS

Originally within the LTSA architecture, the Coach reviews a set of information, such as performance history and objectives, and searches, via the *Query* request, the *Learning Resources* for the proper learning content. In this case, the coach would send two Query requests, one to the Learning Resources for the learning content, and another to SHAREK for the learner added learning resources. SHAREK will process the coach's query and send back the appropriate resources indexes, which in turn are sent, along with the locator index to *Delivery*. *Delivery* will retrieve both the content and the resources and send them as multimedia to the learner. *Evaluation* will send

any behavior feedback with regard to the resources (tags, rates, etc.) back to SHAREK which updates the *Resources Library*. Moreover, SHAREK handles the learners' requests to add new resources.

3.1 Adding Resources

The first and most important step of the SHAREK process is for learners to be able to add, or attach new resources to a course or a lecture. What follows is the data gathered about the different resources added by learners. The model presented here is inspired and based on the IEEE LOM (Learning Object Metadata) [15] standard. The data collected is divided into six categories. The **General** category contains information such as the resource identifier, a unique id assigned by the system, the title, language and a short description of the resource provided by the contributor when adding the resource. Moreover, the general category contains the Tags associated to the resource. Tags are keywords or terms associated with the resource to describe its content. Learner Ratings are also information stored under the general category. A rating is a score (on a scale of 1 to 5) a learner gives to a resource, with regard to its *relevance* to the lesson, its *utility*, and its *clarity* and *ease of use*. Moreover, the flags are stored in the general category. Learners can flag the resource as being *Inaccessible*, *Unrelated* to the lesson, *Redundant* or *Plagiarized*. Flags reflect within the system to advise learners accessing the resource, and tutors are automatically notified for them to take the appropriate actions.

The **Educational** category contains information such as the type of the resource, whether the resource is an exercise, an experiment, a lecture, etc. Moreover, the educational category contains the Related To, and Relation Type information. The first describes which part of the course, or which lesson the resource is related to, and the second describes the relationship of the resource to the lesson: whether to *support* the theory in the lesson, to *contradict* it, to *illustrate* the theory with an example, or to *evaluate* the learner's knowledge in the topic.

The **Technical** category contains information related as to what are the technical requirements to access the resource, the format of the resource, the resource size (if applicable) and its location. The content of *location* varies depending on the format of the resource: if the resource is an uploaded file, then the location indicates where this file is stored. Otherwise, if the resource is located online then the location will contain the URL address.

The **Contributor** category contains the information to identify the learner who contributed, or added the resource. The **Context of Contribution** category holds information about the context in which the resource was added, including the course id, the semester, the tutor and the date the resource was contributed. Although the contributor could be the author of the resource, there are many situations when he is not. Thus, the **Author** category contains information about the actual author of the resource.

3.2 Sharing and Accessing Resources

Within such a context, it is important to have a well designed scheme to help learners locate resources. Indeed, with the availability of a multitude of resources attached to a single lesson or lecture, how can a learner easily locate the most suitable resource?

One approach would be to use the best rated resources, but this approach alone is inadequate. Indeed, different learners prefer different resources. Moreover, when relying just on the ratings, new and suitable resources, which have not yet been rated, would be overshadowed by other, probably less adequate, resources that have been rated. As such, we propose combining several different approaches (Fig. 2), including the use of a *Collaborative Filtering* (CF) recommendation system, *RSS feeds* and a *Search* tool. In short, a CF recommender system accumulates user ratings of items, identifies users with common ratings, and offers recommendations based on inter-user comparison [16].

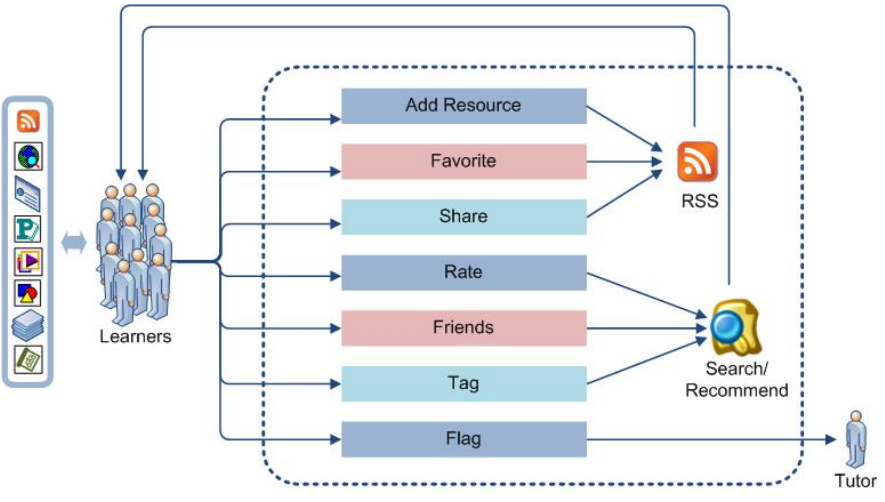


Fig. 2. SHAREK process

Within SHAREK, in the context of the lesson the learner is presented with the Top5 resources attached to the specific lesson. The Top5 are determined using the CF approach, and are presented alongside the learning area. The CF process is composed of two steps: first determine the neighborhood of the learner, which consists of k learners with the highest ratings' similarity. The similarity between the learner a and his *neighbor* u is derived using Pearson correlation coefficient. An important case to consider is the *cold start*: which is when the recommender system first starts and doesn't have enough data to produce reliable recommendations. For instance if the learner is new to the system (or has just a few ratings), the recommender system cannot efficiently determine the neighborhood, nor predict the learner's preference for resources. Thus, in this case, the Top5 will be selected with regard to their overall rating.

Although the recommender system does help locating the material learners appreciate, it is not enough on its own. Indeed, since the recommendation relies on learner ratings, resources newly added to the system might get discarded and overlooked. Therefore, we introduce our second approach, a scheme inspired by *social networking*. Each learner has a list of *friends*, to which he can add or remove other learners. The learner in question is then kept up to date on his *friends'* resources activity through RSS. Specifically, RSS is a family of Web feed formats used to publish

frequently updated content. An RSS document, which is called a *feed*, usually contains either a summary of content, from the associated web site or the full text. In short, RSS makes it easier to keep up with updates. As such, each learner has his own feed per course, automatically updated every time the learner *adds* a new resource, includes a resource into his *favorite*, or *shares* a resource with his friends. In order not to be overwhelmed with updates, a learner can choose, for each of his friends, one or more feeds to subscribe to: add a resource, favorite, or share.

Finally, the third approach consists of a *search tool* used by the learners to locate shared resources. Specifically, learners can search for resources by specifying one or more of the following criteria: language, rating, tags, date added, format, by the educational type, relation type, or even within resources favorite by friends.

3.3 Managing Resources

The freedom offered to learners within SHAREK should be balanced with an equivalent amount of control. Indeed, SHAREK is able to control unintended abuses of learners. One such abuse is the uncontrolled nature of tags. Undoubtedly, tagging has its strengths, and perhaps the most important is that it directly reflects the vocabulary of users. Indeed, tagging, with its uncontrolled nature, can adapt quickly to user vocabulary changes and needs. Yet, this strength is the source of a main disadvantage. In particular, having too many different tags for a single resource affects the quality of a search based on tags. For example, when searching for "bubble sort" on YouTube, the second video in the list retrieved is about Barack Obama (a candidate for the 2008 US presidential elections), and other returned videos include, among other, a bubble gum advertisement. As such, in order to still take advantage of the flexibility of tags, and increase the accuracy of retrieval, we propose to make use of the *tf-idf* weighing scheme [17] such as to allow the most relevant tags to *float* above the rest. The *tf-idf* weighing scheme is a well established approach within the IR (Information Retrieval) field. Specifically, the term frequency tf_{ij} represents the relevance of the term, or tag i to a document j (in our case a resource j), and idf_i represents the discriminating power of the tag i. As such, the most repeated tags with the most discriminating power can be determined for each resource. Currently within SHAREK, for each resource, only the five highest ranking tags, are considered while the other tags are not discarded, such that they might still *float* to the top.

4 Implementation and Validation

In order to validate our approach, we implemented an E-learning prototype platform that supports the tools and functionalities described in this paper. Specifically, the platform proposes three Data Structures lessons, one for each of the following sorting algorithms: *Bubble Sort*, *Merge Sort* and *Selection Sort*. Moreover, a minimum of 6 resources were originally attached to each lesson. The technical environment of the prototype is: PHP, JavaScript, AJAX, and MySQL.

After logging in for the first time, each learner is requested to complete a survey before having full access to the system. This survey collected background information about the learners, such as how often do they refer to learning material outside the regular

class content, where do they look for these resources (search engines, Wikis, library books, etc.), how likely are they to share resources with friends, classmates or tutors, how likely are they to use resources referred by friends, and how familiar are they with computer based training and Web2.0. After completing the survey, the learners can then access the lessons and their respective resources, where they can add new resources, use and rate existing resources, and test the different functionalities of the system. Finally, after having used and tested the system, the learners complete an evaluation form, answering questions about how easy and intuitive the system is, would they use such a system if it existed, and would such a system encourage them to share resources. Moreover, they were asked to evaluate the functionalities of SHAREK, and whether there was any missing functionalities they would like to add, and finally they were asked to give an overall evaluation of the system, and to provide any comments.

4.1 Results and Findings

There were a total of 93 learners who tested the system. The volunteers consisted mainly of graduate and undergraduate students at the Computer Science department of our university (Université de Montréal). We will first start by relating the findings with regard to the survey, then highlight the results of the evaluation.

The survey results mainly reinforce the hypothesis on which we based this work: learners do refer regularly to learning material outside the regular class content, and they are willing to share these resources with friends and classmates. Indeed, when asked, in **question #1** of the survey, to specify on a scale of 1 to 5 (1: never and 5: always) how often they refer to learning material other than what is recommended by the tutor, 93% responded they do so on a regular basis (Fig.). Moreover, when asked to rate how likely they would use a resource recommended by a classmate or friend, on a scale of 1 to 5 (1: never and 5: always) none of the respondents answered with less than 3, and 73% answered higher than 3, which indicates the willingness of learners to use resources recommended by colleagues and friends. Finally, when asked in **question #4** of the survey, to rate on a scale of 1 to 5 (1: never and 5: always) how probably they would share these resources with the Tutor, Friends or

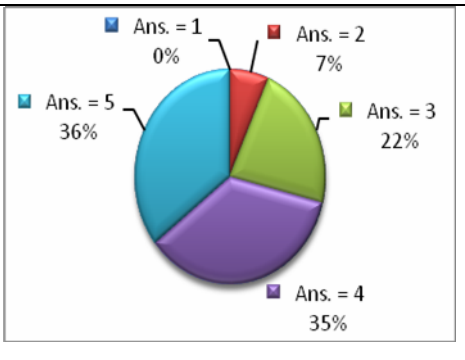


Fig. 3. Partition of answers to the survey's question #1

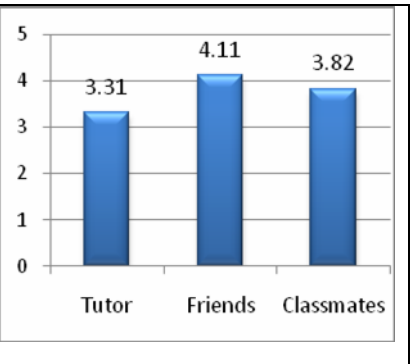


Fig. 4. Average of answers to the survey's question #4

Classmates, learners were the least eager to share resources with their tutor (actually 10% of the respondents answered Never), and most eager to share their resources with their friends. Fig. summarizes the averages of learners' responses. Note that female respondents averaged higher that their male counterparts.

With regard to the evaluation of SHAREK and its functionalities, when asked to rate, on a scale of 1 to 5 (1: Poor and 5: Excellent), the ease of use of SHAREK, the intuitiveness of the interface, and whether they would regularly use such a system, most respondents answered favorably, with ratings' averages around 4. Moreover, when asked in **question #2** of the evaluation, if a system such as SHAREK would encourage then to share resources, on a rate of 1 to 5 (1: never and 5: definitely), most respondents answered with a 3 and higher (see Fig. for more details). In addition, learners who answered with a low score (1 or 2), where learners who initially, in the survey, had also given low scores to their willingness to share resources. In addition, when asked to rate the functionalities provided by SHAREK (such as the RSS feed, search tool, tagging, rating, etc.), on a scale of 1 to 5 (1: poor and 5: excellent) the respondents ratings averaged at 4.1. Moreover, when the learners were asked if there are any additional functionalities that they would require within SHAREK, there were several interesting suggestions, such as a forum and other student communication tools, functionalities which are usually part of E-learning systems (such as Blackboard or ATutor), which SHAREK is intended to complement.

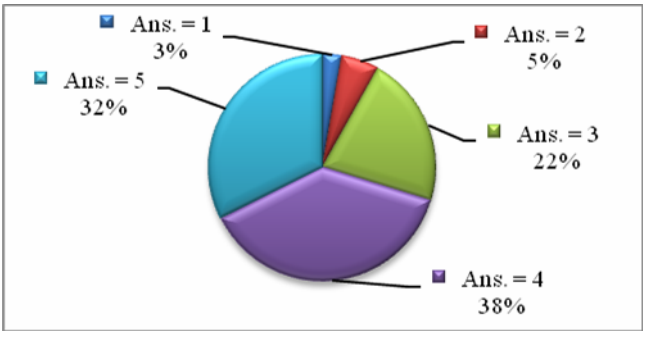


Fig. 5. Partition of answers to the evaluation question #2

With regard to the efficiency of the CF resource recommender system, and the tf-idf approach to float the most relative tags, preliminary findings are encouraging, unfortunately the results were insufficient. In fact, for such analysis to be accurate, we require further testing and data. In our case, all the volunteers used the system only once to evaluate it. Thus, the learners' profiles were *sparse* (learners rated a couple of resources only) in order to perform CF recommendations, and the resources were not accessed and tagged intensively (as they would within a regular course) in order to analyze the tags floating approach.

5 Conclusions and Future Work

Today, E-learning has become a standard, and there exist several virtual universities that offer online programs. Moreover, learners regularly locate and use learning resources from outside the courses' scope. Although these resources are beneficial, to the best of our knowledge there are no functionalities in current E-learning systems to *efficiently* harness and take advantage of this knowledge. In this paper we presented SHAREK (**SHA**ring **RE**sources and **K**nowledge), which allows learners to easily share learning resources within the context of a course. Moreover, SHAREK utilizes well established Recommendation and Information Retrieval techniques to help learners locate the most suitable shared resources. In order to validate our approach we implemented and tested a prototype. Preliminary results are encouraging: out of 93 learners who tested SHAREK, 93% related their readiness to share resources in the pre-test survey, and 90% of the respondents confirmed, after the testing procedure, that they would regularly use such a system and that it would encourage them to share resources.

Nonetheless, we are working at further enhancing SHAREK: we are investigating the inclusion of provisions for competitive group work, such as resources would be shared and accessible only to learners of the same group. In addition, we are looking into considering pedagogical aspects and the *learning profile*, into the recommendation process. Moreover, we are currently in the process of implementing a resource management tool in order to control the size of the resource repository. In particular, we expect the number of resources to grow uncontrollably, thus clogging SHAREK with unused resources, which will become cumbersome for the learners and could cause a cognitive overload. As such, we are developing a methodology, inspired from the confidence and support approach in Data Mining [18] to automatically archive and discard unused and inadequate resources. Finally, we plan to investigate further the expected role of tutors, and their reactions to their new responsibilities within SHAREK.

Acknowledgments. We would like to acknowledge and address our thanks to the FQRSC (Fonds Québécois de la Recherche sur la Société et la Culture) for their support.

References

1. Mccombs, B.L., Vakili, D.: A Learner-Centered Framework for E-Learning Teachers College Record, 107(8), 1582–1600 (2005)
2. Brooks, C.A., Greer, J.E., Melis, E., Ullrich, C.: Combining ITS and eLearning Technologies: Opportunities and Challenges. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053. Springer, Heidelberg (2006)
3. Israel, J., Aiken, R.: Supporting Collaborative Learning With An Intelligent Web-Based System. International Journal of Artificial Intelligence in Education 17(1), 3–40 (2007)
4. Takeuchi, M., Hayashi, Y., Ikeda, M., Mizoguchi, R.: A Collaborative Learning Design Environment to Integrate Practice and Learning Based on Collaborative Space Ontology and Patterns. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 187–196. Springer, Heidelberg (2006)

5. Aïmeur, E., ManiOnana, F.S., Saleman, A.: SPRITS: Secure Pedagogical Resources in Intelligent Tutoring Systems. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 237–247. Springer, Heidelberg (2006)
6. Vassileva, J.: Harnessing P2P Power in the Classroom. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 305–314. Springer, Heidelberg (2004)
7. Brooks, C.A., Panesar, R., Greer, J.E.: Awareness and Collaboration in the iHelp Courses Content Management System. In: European Conference on Technology Enhanced Learning, Crete, Greece, pp. 34–44 (2007)
8. Schauer, B.: Crucial DNA of Web 2.0 (2005), http://adaptivepath.com/ideas/essays/archives/000547.php Retrieved January 2008
9. Burke, R.: Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction 12(4), 331–370 (2002)
10. Singhal, A.: Modern Information Retrieval: A Brief Overview. IEEE Data Engineering Bulletin 24(4), 35–43 (2001)
11. O'Reilly, T.: What Is Web 2.0 (2005), http://www.oreillynet.com/ Retrieved January 2008
12. Lenhart, A., Madden, M., Macgill, A.R., Smith, A.: Teens and Social Media (2007), http://www.pewinternet.org/PPF/r/230/report_display.asp Retrieved January 2008
13. Alexa: The Web Information Company, http://www.alexa.com Retrieved January 2008
14. IEEE Learning Technology Standards Committee. Standard for Learning Technology - Learning Technology Systems Architecture (LTSA) (2003)
15. IEEE Learning Technology Standards Committee. Standard for Learning Object Metadata (2002)
16. Deshpande, M., Karypis, G.: Item-Based Top-N Recommendation Algorithms. ACM Transactions on Information Systems 22(1), 143–177 (2004)
17. Baeza-Yates, R., & Ribeiro-Neto, B.: Modern Information Retrieval. Addison Wesley Reading, MA (1999)
18. Cios, K.J., Pedrycz, W., Swiniarski, R.W., Kurgan, L.A.: Data Mining: A Knowledge Discovery Approach. Springer, London (2007)

Bridging the Gap between ITS and eLearning: Towards Learning Knowledge Objects

Amal Zouaq[1], Roger Nkambou[2], and Claude Frasson[1]

[1] University of Montreal, CP 6128, succ Centre-Ville, Montreal, QC, H3C3J7
[2] University of Quebec at Montreal, CP 8888, succ Centre-Ville, Montreal, QC, H3C3P8
{zouaq,frasson}@iro.umontreal.ca,
Nkambou.roger@uqam.ca

Abstract. This paper presents an ontology-based approach for the dynamic generation of learning knowledge objects (LKO). These objects have the particularity of implementing AIED techniques and they exhibit characteristics hitherto reserved for intelligent tutoring systems (ITS): they are knowledge-based, adapted to a learner model and they are composed dynamically according to a learning need and a particular instructional theory. Additionally, the paper tackles the issue of using eLearning resources by ITS and shows how this could be realized. Finally, the paper also demonstrates how LKOs are deployed and evaluates briefly the results of the proposed approach.

1 Introduction

In the context of eLearning, educational material provides access to many knowledge sources. However, there are several shortcomings with this kind of web-based education. First, the huge number of learning resources makes it difficult to find the appropriate resource to fulfill a learning need. The main burden of indexing, storing, extracting and organizing the learning material remains on human's shoulders [6]. The creation and standardization of metadata offers structuring and indexing, but again this task has to be performed by a human expert. Moreover there are several standardized metadata formats and they require a matching process to be used in a uniform way. Finally, an important shortcoming of educational material is that it cannot be exploited by ITS due to its lack of model.

To alleviate the user's tasks and provide access to better formalized knowledge structures, there is a need to set up (semi)automatic mechanisms to mine the learning object content. There is also an urgent need of rethinking the notion of learning object. From one side, learning object has to evolve to become more appropriate for the current needs of eLearning: adaptation, on-the-fly composition, knowledge modeling, instructional theory modeling and learning objective statement. On the other side, ITS should no longer neglect learning objects as potential knowledge sources.

This paper presents the results of "The Knowledge Puzzle Project" (**KPP**) in its effort to offer an alternative to static learning objects by generating dynamically learning resources called "Learning Knowledge Object" (LKO). These LKOs have several characteristics: they are knowledge-based, theory-aware [18], dynamically generated

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 448–458, 2008.

and they provide a tutoring service interface. Therefore, they draw several characteristics from both AIED and eLearning communities, hence contributing to bridging the gap between them [3].

The first part of this paper (section 2) discusses the current landscape of eLearning and AIED communities. It emphasizes on the current issues facing web-based education and initiates a reflection over the impact of ITS modules on the learning object concept and vice-versa. Section 3 describes a general semantic Web architecture for the on-the-fly generation of LKOs that benefit from the AIED techniques to model the domain knowledge, adapt their content to a given learner and encode instructional theories in a declarative way. Finally, section 4 presents an evaluation of the approach before a conclusion.

2 Computer-Based Instruction Landscape

Computer-based education is mainly divided into two communities: eLearning and AIED. ELearning has been widely adopted by organizations whereas AIED techniques are still generally confined to research projects in the universities.

2.1 ELearning Systems Landscape

Since eLearning relies heavily on web-based resources and on the web as the training platform, we have seen, over the past few years, the proliferation of learning objects in repositories and their wide adoption in the industrial world. This success is also the result of the failure of more sophisticated techniques such as intelligent systems and artificial intelligence in education in general to take place on a wide scale. However, there are many problems in the eLearning vision [4, 17]:

- *The metadata problem:* Metadata is used for the description and research of learning objects. The first problem is that metadata are designed with the idea that they are intended to humans and produced by humans [3]. The second problem is that these metadata, while necessary, are not sufficient. In fact, they describe the world around the learning object (the language used, the context of use, the author, etc.) but they remain very vague about its content.
- *The black-box problem:* A learning object is a black box: it is presented as an integrated content package containing all the necessary resources but its real content is inaccessible to a software agent.
- *The lack of explicit instructional theory problem:* Learning objects implement an instructional theory provided by their designer. However, this theory remains implicit and learning objects are therefore dependent on this theory and cannot modify it dynamically.
- *The lack of adaptability problem:* Finally, learning objects are suffering from their static model and the lack of adaptation of their content to a learner model. This is why various researches focused on this issue and has led to the adoption of various standards for modeling the learner, such as IMS LIP (Learner Information Package) [12], or IMS ePortfolio [11]. The creation of teaching scenarios standards such as IMS-LD (IMS Learning Design) [13] is also another effort to formalize the pedagogical aspect and adapt it to a given learner.

2.2 Intelligent Tutoring Systems Landscape

ITS rely on artificial intelligence techniques to improve the processes of computer-based teaching and learning. Typically, a traditional ITS architecture uses various knowledge representation techniques. One of the main problems of ITS is the lack of an agreed standard among the AIED community regarding domain modeling, learner modeling, and instructional scenario modeling. This lack of standardization is felt in the different modules of an ITS which must be rebuilt from scratch in every project. It is also clearly felt in the knowledge representations themselves: various representations, techniques and languages are used without clear distinction about when any of these formalisms should be used instead of another. The domain model is specially a heavy burden over the community shoulders when built manually, which is often the case.

Finally, with the growing number of learning objects and their availability on a large scale, ITS should benefit from these learning resources and should find a way to integrate them into their knowledge base.

3 Implementing the Semantic Web Architecture: The Knowledge Puzzle Experience

To overcome the limitations mentioned above, the research community has gradually refocused on the need to create more complex representations than simple content aggregations, the need to adapt training to individual learners and the need of knowledge representation and reasoning mechanisms able to exploit the knowledge base. The *educational Semantic Web* [1, 5, 17] represents a way to integrate all these efforts in a common architecture.

In the Knowledge Puzzle Project (KPP), we propose a Semantic Web Architecture which aims at formalizing the different models of the traditional ITS architecture using ontologies and at offering a set of services that exploit the ontological model. Another goal of the project is to provide tools to support the dynamic composition of learning resources that act themselves as small tutoring systems: they possess a domain model, a learner model, and a pedagogical model. They also exhibit different characteristics: they are active, domain-knowledgeable, independent, reusable and theory-aware [18]. We believe that this architecture makes the synergy of the strengths in the eLearning and the ITS field. First, it uses OWL as a standard language for knowledge representation and reasoning. Second it advocates the use of tutoring services to dynamically create learning objects implementing the traditional ITS architecture.

Contrary to classical eLearning approaches where learning objects repositories are considered as suppliers of learning objects, KPP exploits these repositories (or any kind of documents about the domain of interest) as raw materials for creating learning content. A number of tools and services are implemented to make use of these resources to produce domain knowledge and to annotate relevant items according to different contexts (domain, instructional roles, competences, instructional theories). These different contexts are explained below and organized around the three models of an ITS: the domain model, the learner model and the pedagogical model.

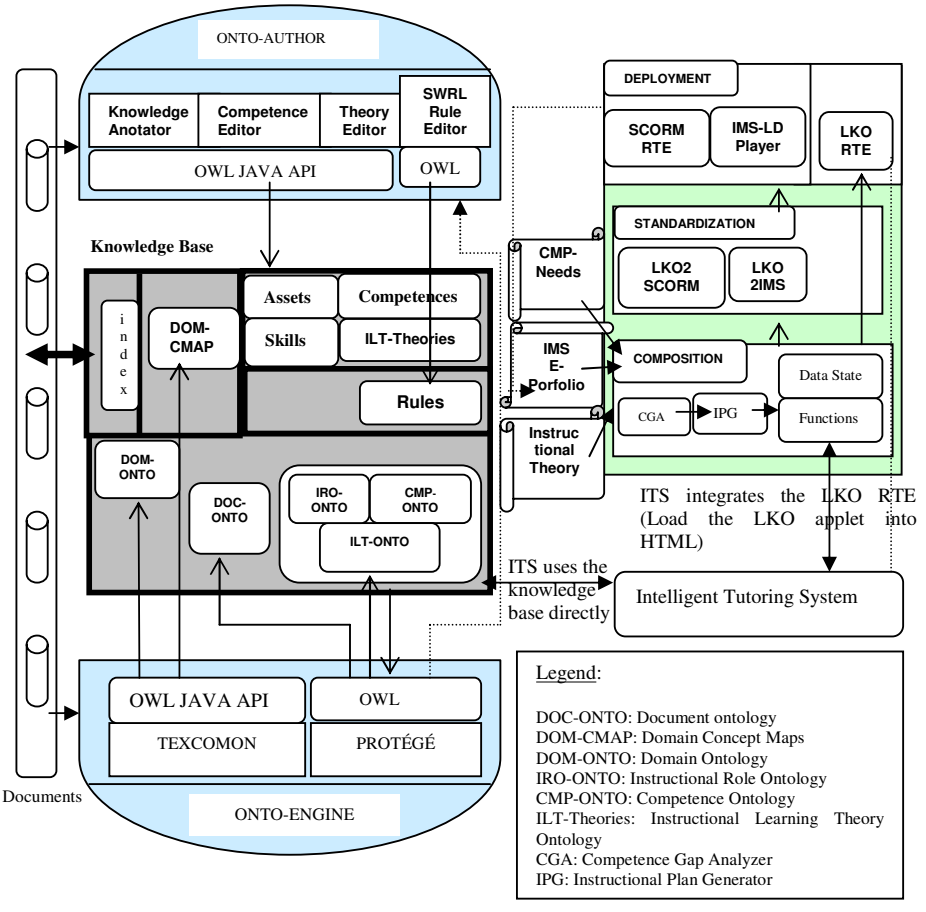


Fig. 1. Overview of the KPP Semantic Web Architecture

The following figure shows the KPP semantic web architecture. On the left, the different tools that feed the knowledge base are represented (Onto-Author, TEXCOMON). On the right, the exploitation of the knowledge base is shown through the deployment of LKOs in eLearning environments and ITS.

3.1 Setting up the Knowledge Base

The knowledge base is represented as a set of intertwined ontologies that are organized around the *domain ontology*. These ontologies are the *competence ontology*, the *instructional role ontology*, and the *instructional theory ontology*. Ontology instances are created through two authoring suite tools: TEXCOMON and ONTO-AUTHOR.

3.1.1 Setting up the Domain Model

In KPP, the domain model is a three-layer structure including a *domain ontology*, a *concept map* and an *index* on related physical learning resources..

A ***domain ontology*** is used to formally describe a domain. Most of the works [8, 13] have considered first building a domain ontology from scratch and then annotating the learning objects with relevant ontological concepts. However, due to the difficulty and cost of such a manual operation, the adoption of automatic methods for extracting the domain knowledge is now acknowledged within ontology engineering communities. The domain ontology is generated semi-automatically through an ontology learning tool (The *TEXCOMON Tool*) that formalizes the domain knowledge as a set of concepts (most of them are primitive classes), individuals, concept attributes and relationships between concepts. The process of learning this domain ontology has been described elsewhere [23]. Without repeating its description, it is important at this stage to underline the benefit gained from the employed methodology.

In fact, one of the main advantages of our approach is the resulting three-layer model that allows for the representation of text content through ***concept maps*** and the refining of these concepts maps into more formal domain ontology. Here, refining means the identification of the most important domain concepts according to the input documents. This importance is measured by the connection degree of a given concept with the others (*the out-degree parameter*). The domain model is thus represented by three structures in ascending order of abstraction: the texts (refered by an index), the concept maps and the domain ontology. It is then easy to navigate from one level to the other by searching a concept map around a given ontological concept and then retrieving text portions relevant to this map and vice versa (starting from a text to know the relevant domain concepts and relationships). In the last case, the domain ontology "automatically" reflects the domain knowledge covered by the input documents (or at least a part of it) and automatically references this knowledge. This is an interesting feature that helps the search of concepts or particular relationships between concepts by providing fine-grained resources (at the sentence-level, paragraph level, or whole text level). Course designers (human, software agents) as well as learners benefit both from these structures respectively to build a course or to understand a domain in an exploratory way.

3.1.2 Setting up the Learner Model

In ITS, the learner model is the key aspect that allows for the adaptation of instruction. We believe that the learner model should be expressed in an ontology in order to offer a better sharing of the model between training environments. For example, Dolog and Nejdl [7] have proposed a user model ontology based on IEEE Personal and Private Information (PAPI) and IMS Learner Information Package (LIP). Another ontology based on an RDF Schema is presented in TANGRAM [14]. An additional specification is emerging as a way to model learner knowledge over a long time period: the IMS ePortfolio specification [11]. This specification has been used in [9] as a way to initialize the learner model and to conserve the trace of mastered competencies. Since the adoption of this specification is rapidly growing in higher education and is also suitable for organizations, we propose to build an ePortfolio ontology to represent the learner model. In KPP, the learner model is expressed as a set of skills (where one or more skills describe a competence) to acquire over domain concepts through an OWLObjectProperty "*concept*". The terminology adopted (the *Bloom taxonomy* [2]) to describe the skills is independent from the domain model (example

of skill: *define, analyze …)* to enable its reusability and an easier sharing. This ontology stores the mastered skills of a given learner over domain knowledge.

3.1.3 Setting up the Pedagogical Model

One of the other drawbacks that face ITS is their inability to model instructional theories and strategies in a standard and reusable manner. The instructional strategies are generally encoded in the ITS program and are then very difficult to update and reuse. Some efforts and reflections have been made to remedy the situation. Hayashi et al. [10] present a framework that gives theoretical justification to standard-compliant learning/instructional scenarios. There were also some efforts to encode instructional strategies in the form of rules by extending the SWRL standard [21].

In KPP, the pedagogical model includes instructional theories and instructional roles ontologies.

The ***instructional theories ontology*** enables to present the same content with different instructional strategies. Each instructional theory is represented as a set of instructional steps. Each instructional step is fulfilled either through the presentation of an instructional role or through the execution of actions that are meant to represent basic tutor operations such as computing the learner score or generating or loading exercises. For expressing this link between a particular step and a given instructional role or a given instructional action, an instructional step is linked to a set of SWRL rules. Given a particular theory, these rules express the kind of processing the tutor should make at a particular step of the instruction (present a definition, generate an exercise, etc.). It is then relatively easy for a designer to edit or add new rules or create ad-hoc rules. For the moment, we still use the SWRL editor provided by the Protégé Environment but easier ways to edit instructional rules are currently undertook. Other initiatives regarding ontology-based instructional theory modeling are established by the research community such as the OMNIBUS project [17]. Since KPP instructional theory model is very simple, the final aim would be to integrate a richer ontology such as OMNIBUS instead of the one we currently use.

The ***instructional role ontology*** represents instructional roles widely used in education such as *definitions, examples, descriptions, exercises*, and so on. These roles are related to the domain ontology through an OWLObjectProperty "*concept"* that identifies the referred topic. Instructional roles allow for the identification of reusable pedagogical knowledge fragments in learning objects or documents. These fragments are needed for the establishment of an automatic generation process of learning resources. Hence, they sustain the reusability of fine-grained learning resources, which is one of the main aims of eLearning.

The *ONTO-AUTHOR* tool suite [22] is used to edit all the ontology instances. It comprises functionalities such as competence edition, instructional role annotation, instructional theory edition, etc. It allows establishing the pedagogical, structural, domain and competence relationships between the contents. The *ONTO-AUTHOR* tool suite uses the Protégé OWL API to communicate with the ontology layer.

3.2 Exploiting the Knowledge Base

As previously mentioned, KPP advocates a service-oriented architecture to exploit the knowledge base. Once the different models are set up, it is possible to organize a set

of services to effectively use the KB. Other projects have recognized the importance of the service-based vision in contrast with the resource-based vision such as the TELOS architecture [19] and the LUISA project [15]. In the same line of thought, KPP proposes the replacement of LORs by knowledge bases which should be exploitable by LCMS and ITS. One difference in the KPP vision is that we also consider a learning resource as a **small knowledge base surrounded by services** (the LKO). In fact, setting up the knowledge base as described above aims at providing the required resources to automatically generate LKOs. These LKOs can be usable by eLearning environment as well as by ITS.

The LKO generation process is composed of three layers represented as services: the composition service, the deployment service and the standardization service (see fig.1, right side).

3.2.1 The Composition Service

The composition service is in charge of aggregating learning Knowledge Objects according to a given instructional theory, a specific competence need (CMP-Needs) and a specific learner model (IMS e-Portfolio).

Once learning objectives are specified, the competence definition is matched against the learner profile using a *Competence Gap Analyzer (CGA).* For each skill in the targeted competence, the learner profile is checked to find out if this skill is already mastered or if required prerequisites have to be added. If this is the case then adjustments are made to the competence definition leading to a set of skills that are new to the learner (Adjusted Competence).

The Adjusted Competence is transferred to an *Instructional Plan Generator* (IPG) that is in charge of composing the actual Learning Knowledge Object according to the chosen theory. The IPG exploits the Instructional Learning Theory Ontology in order to find out the instructional events and conditions that will effectively guide the composition. Actually, KPP supports the execution of SWRL rules using the Jess rule engine. As previously indicated, the execution of the rules indicates to the IPG the type of resources that should be used to fulfill each step or the type of action that should be performed. Each action described in the rules is implemented as a java function (for example: generate an exercise related to a given concept).

Since an LKO is generated on-the-fly according to a particular training need, it requires, at generation time, an ontological data structure to store relevant instances and resources. This data structure is represented by the LKO ontology. The instructional Plan Generator fills the LKO ontology with the required resources.

3.2.2 An Ontology for Representing Data States in Learning Knowledge Objects

The execution of the IPG produces a Learning Knowledge Object that is composed of a data state and of a set of functions to manipulate it.

An LKO defines learning scenarios through topics from the domain ontology based on the definition of competences as learning objectives and guided by an instructional theory. The data state is an OWL data structure composed of the various resources necessary for the LKO. An *LKO* is represented by a set of *LKOActivities (CourseActivity or ExerciseActivity).* Each activity is targeted towards a specific skill and is linked to the skill concept. Each concept is in turn related to a set of *LKOSemanticRelations* that define the

concept map around the concept (its context). Each *LKOActivity* is composed of a set of *LKOSteps* that are conformant to a specific instructional theory. Each of the steps requires a *resource* to fulfill it.

The LKO functions are assembled into a standard interface that enables any LKO to act as a small "Intelligent Tutoring System". This standard interface offers a number of functions that are further described in the deployment service.

3.2.3 The Deployment Service

The LKOs are targeted towards any kind of training environment. KPP provides an LKO Runtime Environment (LKO-RTE). The LKO-RTE makes it possible to run the LKO as a standalone resource. The user interface gives access to relevant functions that are implemented within the LKO to support the variety of learning services (functions) described below.

Scenario control is set up by following the instructional theory that was used to generate the Learning Knowledge Object. Basically the LKO Runtime Environment is composed of a *course view*, a *concept map view* and *concept map exploration view*.

One deficiency of current textual learning objects is that they only foster one kind of learning. KPP's aim is to offer some kind of constructivist environment using the learning object content. Each concept related to the skills that compose the learning knowledge object is presented with its context. The concept map view shows the context of these different concepts and the relationships between them.

The *concept map exploration* enables a deeper understanding of the concepts at hand. In fact, it enables to explore the concepts connected to the current competence and it also allows exploring the surrounding concepts. Another possibility is to *explore the different instructional roles* related to a concept as well as their source documents. This can be helpful for a learner as well as for a course designer, because it gives a quick view of the available resources.

Besides concept map exploration, the learner is offered with *exercises* to test his understanding regarding the presented concepts. An exercise can be any learning resource that is edited in the ONTO-AUTHOR Tool Suite. Single or multiple choice exercises can also be generated by exploiting the concept maps. Here an exercise is defined as a set of right and erroneous assertions where the learner should select the right ones. The right assertions (relationships between concepts) come from the domain model. The erroneous ones are automatically created by exchanging relationships labels or concepts labels with other ones from the domain model that are not appropriate to the current learning situation.

Finally, at any time, the learner is offered with the possibility of asking for the *generation of a new learning knowledge object* related to some concept of the context. For example, the learner can ask for the generation of an LKO about the concept of LMS (Learning Management System).

3.2.4 The Standardization Service

The standardization layer serves as an interface to different standard eLearning environments and to ITS.

For each eLearning standard (SCORM and IMS-LD), the same methodology is employed: Once the LKO is generated according to a particular instructional theory, it is possible to export the generated LKO ontology as an OWL File "lko.owl". This file indicates the scenario to follow, the different resources related to each step and the

concepts and their context. An LKO applet takes this OWL file as input to display the generated LKO. This applet can be packaged in a SCORM content package with other required resources. The same procedure is used for IMS-LD where an LKO is considered as an activity executable in a Learning Design Player [20].

As far as ITS are concerned, a different approach may be necessary. If the ITS relies on web-based resources such as HTML pages, it is then possible to launch the LKO applet as a resource to fulfill the given competence need. Proprietary ITS should also follow this methodology to be able to launch an LKO. Otherwise, a standard ontology-based ITS (what we call the LKO Runtime Environment) is required. This environment uses the ontologies as its knowledge base and exploits also standard services such as the ones described above. However, one may have noticed that an essential component in the ITS architecture is missing: the expert module. The actual knowledge base focuses on declarative knowledge. The expert procedural knowledge should then be manually added with usual techniques in the form of rules that come on top of the domain model.

4 Evaluating the Semantic Web Architecture

We performed evaluations of the proposed approach by mainly focusing on the semantic validity of the domain concept maps and the domain ontology. In fact, since the domain model represents the basis of the proposed framework, it is really important to ascertain the interest of the ontology learning techniques to produce relevant results. We completed a three-level analysis: a structural, semantic and comparative analysis. The structural analysis considers the domain ontology as a graph and focuses on structural properties such as the out-degree of a concept, the centrality of a concept, etc. We found that the concepts were generally richly described and had sufficient amount of parents. The semantic analysis relies on human experts to judge the validity of the ontology. On average, the two experts rated the pertinence of the generated ontology as follows: 86.65% for primitive classes, 84.3 for hierarchical relationships and 80.08 for conceptual relationships. Finally, we completed a comparative evaluation (on the same corpus) with the *Text-To-Onto* Tool [16] where the analysis has shown a significant improvement of results with KPP ontologies especially in *conceptual relation learning*, often considered as one of the weakest points of similar text mining platforms.

An empirical evaluation of the LKOs is needed to complete these experiments. Our goal is to compare, on a given subject, LKOs and traditional learning objects with a set of learners. Moreover, the main part of our efforts has been devoted, until now, to the evaluation of the automatic extraction techniques. However, another concern is to evaluate the workload for the teaching staff to set up the knowledge base. In fact, since annotation and edition tools are very simple to use, we do believe that this should not take more effort (at least) than "traditional" platforms. From the domain knowledge side, it is important to underline that an important part of the work is done automatically but the ontology should still be validated and enriched, which requires some effort from the expert.

5 Conclusion

We presented a semantic web architecture that allows the incorporation of AIED techniques in learning objects. We also introduced a new definition of learning objects: these resources should not be static content packages but programs able to reason over a small knowledge base. This new definition is concretized in the Learning Knowledge Objects that exhibit various characteristics and implement the three main models of ITS' architecture: the domain model, the pedagogical model and the learner model. The architecture proposed here has many advantages: it lessens the burden of the manual creation of the domain model, which is one of the main obstacles to AIED, it allows guiding learners towards the right content, it clarifies the links between learning objects and between learning objects and domain concepts, and finally, it offers the ability to switch from an instructional theory to another by regenerating a new LKO. From the ITS side, we proposed a new vision based on the definition of the ITS architecture through ontologies. We also showed how ITS could exploit LKOs.

References

1. Aroyo, L., Dicheva, D.: The New Challenges for E-learning: The Educational Semantic Web. Educational Technology & Society 7(4), 59–69 (2004)
2. Bloom, B.: Taxonomy of educational objectives: The classification of educational goals: Handbook I, cognitive domain. Longman, New York (1956)
3. Brooks, C.A., Greer, J.E., Melis, E., Ullrich, C.: Combining ITS and eLearning Technologies: Opportunities and Challenges. In: Proceedings of ITS, pp. 278–287. Springer (2006)
4. Brooks, C., McCalla, G., Winter, M.: Flexible Learning Object Metadata. In: Proceedings of SWEL 2005 held in conjunction with AIED 2005, Amsterdam, pp. 1–8 (2005)
5. Devedzic, V.: Semantic Web and Education. Springer, Berlin (2006)
6. Devedzic, V.: Key Issues in Next-Generation Web-Based Education. IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews, 339–348 (2003)
7. Dolog, P., Nejdl, W.: Challenges and Benefits of the Semantic Web for User Modeling. In: Proceedings of AH2003 workshop at 12th WWW Conference, Budapest (2003)
8. Gasevic, D., Jovanovic, J., Devedzic, V.: Ontologies for Creating Learning Object Content. In: Proceedings of the 8th International Conference on Knowledge-Based Intelligent Information & Engineering Systems, pp. 284–291. Springer, Wellington (2004)
9. Guo, Z., Greer, J.: Electronic Portfolios as a Means for Initializing Learner Models for Adaptive Tutorials. In: Innovative Approaches for Learning and Knowledge Sharing, pp. 482–487. Springer, Berlin (2006)
10. Hayashi, Y., Bourdeau, J., Mizoguchi, R.: Standard-compliant Scenario Building with Theoretical Justification in a Theory-aware Authoring Tool. In: Proc. of AIED 2007, Marina Del Rey, pp. 37–44 (2007)
11. IMS ePortfolio Specification (2007), `http://www.imsglobal.org/ep/index.html`
12. IMS LIP (2001, 2003), http://www.imsglobal.org/profiles/
13. IMS-LD (2007), http://www.imsglobal.org/learningdesign/index.html

14. Jovanovic, J., Gasevic, D., Devedzic, V.: Ontology-based Automatic Annotation of Learning Content. International Journal on Semantic Web and Information Systems 2(2), 91–119 (2006b)
15. LUISA. (2007), http://luisa.atosorigin.es/www/ Retrieved March 10, 2008
16. Maedche, A., Staab, S.: Ontology Learning for the Semantic Web. IEEE Intelligent Systems 16(2), 72–79 (2001)
17. Mizoguchi, R., Hayashi, Y., Bourdeau, J.: Inside Theory-Aware Authoring System. In: Proc. of The Fifth International Workshop on Ontologies and Semantic Web for E-Learning (SWEL 2007), Marina del Rey, CA, USA, July 9, 2007, pp. 1–18 (2007)
18. Mizoguchi, R., Bourdeau, J.: Using Ontological Engineering to Overcome Common AI-ED Problems. Journal of Artificial Intelligence and Education 11, 107–121 (2000)
19. Paquette, G., Rosca, I., Mihaila, S., Masmoudi, A.: Telos, a service-oriented framework to support learning and knowledge management. In: Pierre, S. (ed.) E-Learning Networked Environments and Architectures: a Knowledge Processing Perspective, Springer
20. Reload Editor and Player (2007), http://www.reload.ac.uk/ldplayer.html Retrieved May 2007
21. Wang, E., Kim, Y.S.: Using SWRL for ITS through Keyword Extensions and Rewrite Meta-Rules. In: Proceedings of SWEL, Marina Del Rey, pp. 101–105 (2007)
22. Zouaq, A., Nkambou, R., Frasson, C.: A Framework for the Capitalization of e-Learning Resources. In: Proceedings of ED-MEDIA, pp. 1241–1247. AACE (2007a)
23. Zouaq, A., Nkambou, R., Frasson, C.: Building Domain Ontologies from Text for Educational Purposes. In: Proceedings of the 2nd European Conference on Technology-enhanced Learning, pp. 393–407. Springer, Berlin (2007b)

Semantic Cohesion and Learning

Arthur Ward and Diane Litman

University of Pittsburgh, Pittsburgh, Pa., 15260, USA

Abstract. A previously reported measure of dialog cohesion was extended to measure cohesion by counting semantic similarity (the repetition of meaning) as well as lexical reiteration (the repetition of words) cohesive ties. Adding semantic similarity ties improved the algorithm's correlation with learning among high pre-testers in one of our corpora of tutoring dialogs, where the lexical reiteration measure alone had correlated only for low pre-testers. Counting cohesive ties which have increasing semantic distance increases the measure's correlation with learning in that corpus. We also find that both directions of tie, student-to-tutor and tutor-to-student, are equally important in producing these correlations. Finally, we present evidence suggesting that the correlations we find may be with deeper "far transfer" learning.

1 Introduction

Researchers in Intelligent Tutoring Systems often study tutorial dialog for clues to the effectiveness [1] of human tutors. This research has focused on many aspects of the tutoring interaction, from tutorial support [2] to the occurrence of various dialog acts [3,4]. Because deep dialog act features such as question answering are difficult for tutoring systems to identify automatically, our own research has also focused on shallow dialog features (such as word count or turn length) which are more automatically computable [5]. Unfortunately these shallow features tend to have poorer correlations with learning than the deeper features [6]. We look for correlations between dialog act features and learning both because we want to be able to detect learning during tutoring, and because we want to be able to design effective tutorial dialog interventions later on.

Much work on tutorial interventions, however, suggests that their effectiveness is often dependent on the preparedness level of the student. For example, VanLehn et al. [7] present evidence that tutoring is only better than reading when the reading material is too difficult for that particular student. Kalyuga et al. describe a number of instructional techniques which work for low-knowledge students, but not for high-knowledge students [8]. Similarly, Conati and VanLehn [9] find that providing rich scaffolding for self-explanation helps low-knowledge students, but can actually impede learning for high-knowledge students. McNamara and Kintsch [10] find that increasing the coherence of text can aid recall for low, but not for high-knowledge readers.

In a previous paper [11], we found a similar interaction with student preparedness. In that work, we measured the cohesion of tutorial dialog in a way similar to the lexical reiteration cohesion baseline described in Section 4. We found that the amount

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 459–469, 2008.

of cohesion in our tutorial dialogs predicted learning for our below-mean, but not our above-mean pre-testers. In that paper [11], we speculated that maybe cohesion could predict learning for our high-knowledge students, but we were measuring the wrong kind of cohesion. Perhaps measuring the reiteration of senses (meanings), rather than of words, would correlate with learning in high-knowledge students. In this work we have implemented that idea, and report on the results.

We find that in one corpus of tutoring dialogs, adding a count of semantic similarity reiterations to our lexical reiteration measure does indeed improve its correlation with learning among above-mean pre-test students. We find that lowering a similarity threshold so that more semantically distant pairs are counted as cohesive ties improves this correlation. We also find that tutor-to-student and student-to-tutor cohesive ties are equally well correlated with learning. Finally, we present suggestive evidence that our correlations may be with the deeper learning measured by "far-transfer," as opposed to "near-transfer" questions.

2 Related Work

In Section 5, we discuss a method by which dialog cohesiveness can be calculated using a WordNet [12] based measure of semantic similarity. Many measures of semantic similarity based on the WordNet taxonomy have been described in the computational linguistics literature. These measures range from counting edges [13], to adjusting edge counts with other information such as depth in the taxonomy [14] or information content calculated from a corpus [15].

These systems are typically evaluated by comparing them to human judgments or by seeing how they perform in tasks such as spelling correction [16]. We differ from the semantic similarity work mentioned above in that we apply our measure to tutorial dialog, and evaluate it by how strongly it correlates with learning in our corpora of tutorial dialogs. Pending future work, we use the simplest reported measure of semantic similarity.

Other work examining the cohesiveness of tutorial dialog has been done by the AutoTutor group at the University of Memphis. In [17], they use the CohMetrix [18] cohesion analysis tool to analyze the cohesiveness of tutor and student dialog contributions along many dimensions. Our semantic measures are similar in spirit, but where they use LSA to gauge the distributional similarity between two turns, we use a WordNet similarity metric to locate specific pairs of similar words between turns. Their "argument overlap" metric is also very similar to our lexical reiteration measure. However, we look for correlations between dialog cohesion and learning, whereas [17] examines cohesion differences between tutoring and other types of discourse.

3 Tutoring Dialog Corpora

We test our model on two corpora of tutoring transcripts collected by the Itspoke intelligent tutoring system project [5] in 2003 and 2005. Itspoke is a speech enhanced version of the Why2 qualitative physics tutoring system [19]. In both

experiments, the Itspoke tutor taught qualitative physics to students who had never taken physics before. The tutor was identical in each case, except that the version used in 2005 had a larger language model to improve speech recognition during dialog. In this work, we use dialog transcriptions, rather than speech recognizer output. Students for the 2003 study were recruited by flyer, whereas students in 2005 were recruited via their "Introduction to Psychology" instructor, as well as by flyer.

In each experiment, the students would first read instructional material about physics, and take a pre-test to gauge their physics knowledge. The tutor would then present a problem in qualitative physics, which the student would answer in essay form. The computer tutor would interpret the essay and engage the student in a spoken dialog to teach a missing or incorrect point. This would repeat until the tutor was satisfied that all points were covered. Each student worked through five problems this way, then took a post-test.

The 2003 and 2005 pre and post tests contained 26 questions in common. The tests used in 2005 contained an additional 14 questions re-used from other experiments. A pilot tagging study suggests that the full 40 question set (40Q) used in 2005 contains about 50% "far" transfer questions that were non-isomorphic to the tutored problems. However, the 26 question set (26Q) is about 27%, and the 14 question set (14Q) is over 90% "far" transfer. A firm classification of the questions into "near" and "far" is left for a more formal tagging study. Here, we present results for the three 2005 question sets separately, and suggest that their differences may be because of differing proportions of "far" transfer questions.

Table 1. Test Scores

	2003		2005 40Q		2005 26Q		2005 14Q	
Group	**M**	**SD**	**M**	**SD**	**M**	**SD**	**M**	**SD**
All Pre	0.48	0.17	0.54	0.17	0.49	0.18	0.61	0.18
All Post	0.69	0.18	0.71	0.14	0.70	0.16	0.70	0.15
High Pre	0.67	0.14	0.68	0.09	0.65	0.10	0.75	0.09
High Post	0.79	0.13	0.82	0.09	0.82	0.10	0.78	0.14
Low Pre	0.38	0.06	0.41	0.10	0.35	0.09	0.45	0.10
Low Post	0.64	0.18	0.61	0.11	0.61	0.14	0.60	0.09

There were twenty students in the 2003 study, who completed a total of ninety-five dialogs. A mean pre-test split divided these students into 13 "low" pre-testers and 7 "high" pre-testers. There were 34 students in the 2005 study, who completed 163 dialogs.[1] A mean pre-test split using the 40Q or 26Q test results divides these students into 18 "low" and 16 "high" pre-testers. Using the 14Q set divides them into 16 "low" and 18 "high" pre-testers. In each experiment, pre-test splits were done relative to the question set being used. Mean (M) pre and post-test scores, with standard deviations (SD) are shown in Table 1 for each pre-test group (All students, High & Low pre-testers).

[1] Dialogs were not always collected for every one of a student's five problems, because the computer tutor would sometimes accept the initial essay without discussion.

4 Baseline Cohesion Measure - Lexical Reiteration

As mentioned in Section 1, we want to know if measuring cohesion at the "sense" level will improve our previous lexical cohesion measure. Our baseline, therefore, will be a lexical reiteration cohesion measure similar to the one used in [11]. This measures the cohesiveness of a dialog by counting the number of token and stem matches between utterances, after removing stop words. A stem is the "root" form of a word, which we find using a standard Porter stemmer. An illustration of this measure is shown in Table 2. The top two lines of the table show two consecutive utterances from one of our dialogs. Nine cohesive ties can be counted between these utterances at the token level. The tokens matched are shown in row three of the table, and in bold in the utterances. Cohesive ties can also be counted at the stem level, by counting a tie whenever one utterance and the next contain words with the same stem. An example of this is shown in row four of Table 2, where the tokens "force" and "forces" have matched by stem. In both of our measures, cohesive ties are counted between all consecutive pairs of utterances, ie: both from tutor-to-student and student-to-tutor. This measure is a close approximation to the "exact word repetition" type of cohesion described by Halliday and Hassan [20] in *Cohesion in English* .

Table 2. Token, Stem, and Semantic Similarity (Sem) Matches

Speaker	**Utterance**
Student	Before the **release** of the **keys**, the man's and the keys velocity are the same. **After** the release the **only force** on the keys and **man** is downward force of earth's **gravity**, **so** they are in freefall. We **can** ignore the **forces** that the air exerts on these objects since they are dense. Therefore, at every **point** in time the keys will remain in front of the man's face during their whole trip down.
Tutor	**So** you **can** compare it to your response, here's my summary of a missing **point**: **After** the **release**, the **only force** on the **person**, **keys**, and elevator is the **force** of **gravity**. Kindly correct your essay. If you're finished, press the submit button.
Level	**Cohesive Ties Counted between Utterances, at each level**
Token	so-so, release-release, point-point, only-only, keys-keys, gravity-gravity, can-can, after-after, force-force
Stem	forces-force
Sem	man-person

We count the total number of cohesive ties for each dialog as described above. We then line normalize the count, dividing it by the total number of lines in the dialog. We do this to remove the possibility that the count of cohesive ties correlates with learning simply because the longer dialogs had more cohesive ties. However, neither the total number of tutor turns, student turns, tutor words, or student words are correlated with learning in spoken dialogs with our computer

tutor [5]. In the example shown in Table 2, we count a total of 10 cohesive ties at the token and stem levels, line-normalizing the count (as if this were an entire dialog) would give a score of $10/2 = 5$.

Finally, we sum the line normalized counts over all dialogs for each student, resulting in a per-student cohesion measure which we correlate with learning. As in previous work [11], we use partial correlations of post-test score with our cohesion count measures, controlling for pre-test score. We control for pre-test score because it is significantly correlated with post-test score in both our 2003 corpus ($r(18)$=.462, p=.04) and in our 2005 corpus (40Q:$r(32)$=.817; 26Q:$r(32)$=.741; 14Q:$r(32)$=.698, all $p<$ .001).

5 New Cohesion Measure - Semantic Similarity

We next extend the baseline measure described above by counting semantic similarity, as well as lexical reiteration cohesive ties. We count a cohesive tie at the semantic similarity level whenever one utterance and the next have different words with similar meanings, and we measure similarity using WordNet [12]. WordNet is a large semantic lexicon with several features that are useful here. First, it groups words into groups of synonyms called "synsets." Second, it organizes synsets into an "is-a" (hypernym/hyponym) taxonomy. If we know the sense in which a word is being used, and therefore its relevant synset, we can use WordNet to find its relationship to other synsets in the taxonomy. An example of these relationships can be seen in Figure 1, which reproduces a portion of the WordNet taxonomy. For one sense of "man," we can see in WordNet that man is a type of male, and that male is a type of animal. Man is also a type of adult, which is a type of person.

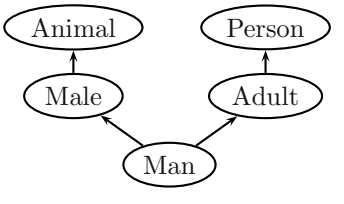


Fig. 1. Wordnet paths

We do not attempt to do word sense disambiguation before measuring the similarity of two words in WordNet. Instead, for each potential pair of words, we choose the senses in which the words are most similar.

We measure semantic similarity as a function of the distance between two concepts in the WordNet hierarchy. In this work, we use the simplest method of measuring this distance: Path Distance Similarity, as implemented in NLTK [21]. This measure calculates the similarity between two concepts as 1/1+N, where N is the number of edges in the shortest path between them. Scores range from zero to one, where zero means no similarity and one indicates exact synonyms. In Figure 1, the shortest path between "man" and "person" has two edges, and so their similarity is 1/1+2, or.333.

5.1 Identifying Semantic Ties

Finding semantic similarity ties is slightly more complicated than finding lexical reiteration ties because a word in one utterance may have non-zero similarities to

several words in the other utterance. Therefore, we use the following algorithm to find the best set of ties. For each word in utterance B, we look up the WordNet path similarity values between it and each word in utterance A. After collecting the set of all possible word pairs this way, we sort them by their similarity values. Starting at the high end of the list, for each pair we remove all lower pairs which have the same token in the same position. This process is illustrated in Table 3. To keep the example small, we have selected only two tutor and three student words from the example shown in Table 2. This produces six possible pairs, which are shown in columns two and three of Table 3, sorted by their similarity values. Starting at the top of the list, we consider first the pair: "man-person." We remove all instances below of "man" in position A and of "person" in position B. This step is shown under "Step 1" in Table 3. In step 2, we move down to the next remaining pair, "release-elevator." We remove all instances below that of "release" in position A and of "elevator" in position B. There are no pairs remaining to be considered in our example, so we stop and count two semantic cohesive ties: "man-person" with a similarity of .33, and "release-elevator" with a similarity of .13.

Table 3. Finding the best semantic ties

	Start		Step 1		Step 2	
Sim	**Tok A**	**Tok B**	**Tok A**	**Tok B**	**Tok A**	**Tok B**
0.33	man	person	**man**	**person**	man	person
0.13	release	person	release			
0.13	release	elevator	release	elevator	**release**	**elevator**
0.11	velocity	person	velocity		velocity	
0.09	man	elevator		elevator		
0.07	velocity	elevator	velocity	elevator	velocity	

This method can count cohesive ties with a broad range of similarity scores. We will investigate whether the stronger ties are more useful by instituting a threshold, and only counting cohesive ties for pairs with similarity values above the threshold. In the example shown in Table 3, a threshold of .3 would count the tie between "person" and "man" but not between "elevator" and "release."

A threshold $> .5$ counts cohesive ties only for word pairings which are listed in WordNet as being exact synonyms, and which therefore have a similarity score of one (note from the path similarity formula that scores between .5 and 1 are impossible). A threshold reduced to .3 allows cohesive ties with slightly more semantic distance in the pair, and a threshold of 0 allows all pairs found by our

Table 4. Example Semantic ties

Threshold		
> 0.5	**0.3**	**0**
5-five	motion-contact	remains-same
remain-stay	man-person	man-runner
speed-velocity	decrease-acceleration	force-magnitude
conclude-reason	acceleration-change	summarize-point
package-packet	travel-flying	submit-pull

algorithm. Examples of cohesive ties counted at each of these thresholds are shown in Table 4. In these examples we can see that the matches counted become more distant and less sensible as the threshold is reduced.

5.2 Semantic Similarity Measure

We count the number of cohesive ties between two utterances by first counting all the exact token matches between them, then counting ties based on stem matches as described in Section 4. After these lexical reiteration ties are identified, we look for semantic similarity ties among the remaining words.

An example of an additional cohesive tie counted at the semantic similarity level is shown in row five of Table 2. Here a tie between the tokens "man" and "person" has been counted which, as shown in Table 3, have a semantic similarity of .33. Adding this tie to the baseline measure from Section 4 brings our cohesive tie count to 10.33, and our normalized cohesion score for the example to 5.16.

6 Results

Table 5. Learning-Cohesion Correlations

pre-test Group	2003		2005 40Q		2005 26Q		2005 14Q	
	Cor	pVal	Cor	pVal	Cor	pVal	Cor	pVal
	Lexical Only							
All	0.474	**0.035**	0.273	0.118	0.185	0.295	0.289	0.098
Low	0.682	**0.005**	0.606	**0.013**	0.279	0.263	0.462	0.072
High	0.798	0.105	0.152	0.546	0.084	0.756	0.333	0.177
Lexical plus WordNet Similarity Threshold = .5 to .99								
All	0.470	**0.036**	0.276	0.114	0.187	0.289	0.298	0.087
Low	0.686	**0.005**	0.605	**0.012**	0.286	0.250	0.473	0.064
High	0.825	0.085	0.159	0.527	0.084	0.757	0.336	0.173
Threshold = .3								
All	0.470	**0.037**	0.277	0.112	0.182	0.303	0.308	0.076
Low	0.689	**0.004**	0.613	**0.011**	0.271	0.276	0.495	0.051
High	0.899	**0.038**	0.153	0.543	0.070	0.797	0.341	0.166
Threshold = 0								
All	0.451	**0.046**	0.286	0.100	0.183	0.301	0.337	0.051
Low	0.665	**0.007**	0.607	**0.012**	0.259	0.300	0.519	**0.039**
High	0.984	**0.002**	0.161	0.522	0.082	0.763	0.378	0.122

The top section of Table 5 shows results for our lexical measure alone. Here cohesive ties are counted for token and stem matches between utterances, but not for semantic similarity matches. In the 2003 corpus (cols 2 & 3), this measure produces significant correlations with learning for below mean pre-testers, and for the group of all students. It does not produce significant correlations with learning for above mean pre-testers. This pattern is similar to the one reported in [11] [2].

[2] The correlations shown here are slightly different from those reported in [11] because of small differences in implementation.

Next we examine results for the 2003 corpus after adding the semantic similarity measure to the previous lexical reiteration measure, shown in the lower three sections of Table 5. As the threshold is reduced, correlations for high pre-testers become significant and increasingly stronger. Fisher's z-test indicates that the improvement in high pre-tester correlations between the $> .5$ and 0 thresholds is significant ($p <= .0003$).

Results for the 2005 corpus are shown in the right three sections of Table 5. Unfortunately, the success of the semantic similarity measure among high pre-testers does not replicate in this corpus. Our measure correlates with learning only among low pre-testers. However, comparing results from different question sets gives us some insight into what sort of learning is correlating. Note that we get strong, significant correlations for low pre-testers in the 40Q question set (cols 4 & 5), which we have argued includes 50% far transfer questions. For the 26Q set (cols 6 & 7), which has fewer far transfer, we get no correlations. However for the 14Q set (cols 8 & 9), which is probably almost all far transfer, we see a significant correlation for the semantic measure at a threshold of zero, but not for the lexical measure. This suggests that our semantic measure may be correlating with the deeper learning measured by far transfer questions.

As described in Section 4, the results presented in Table 5 are bi-directional, ties are counted both when the student's utterance follows the tutor's, and the other way around. It is interesting, however, to consider whether one direction of tie is more important than the other. Results for uni-directional cohesive ties are shown in Table 6, for the 2003 corpus and for the 2005 corpus with the full question set. For brevity we present only significant correlations with a threshold of .3. Comparing these results to those shown in Table 5 for the same threshold indicates that both directions are equally responsible for our results among low pre-testers. This suggests the possibility of increasing learning by altering tutor word choice to manipulate dialog cohesion.

Table 6. Directional Correlations

pre test	**2003**		**2005 40Q**	
Group	**Cor**	**pVal**	**Cor**	**pVal**
Threshold = .3				
Tutor to Student				
Low	0.682	**0.005**	0.602	**0.014**
Student to Tutor				
Low	0.634	**0.011**	0.593	**0.015**

7 Discussion

As mentioned in Section 1, this work grew out of speculations we developed when trying to explain the results reported in [11]. We hypothesized that the lexical reiteration cohesive ties we were counting signaled inferences which led to learning among our low pre-testers. We wondered if being able to recognize cohesive ties between different words with similar meanings would allow us to detect the deeper inferences that might lead to learning among high pre-testers. For example, perhaps hearing (or producing) "person" when the dialog partner just used "man" is associated with deeper inference than simply re-using "man."

Results from both corpora suggest a relationship between inference and cohesion. In the 2003 corpus, counting semantic reiteration cohesive ties does improve

learning correlations for high pre-testers. Also, counting more semantically distant pairs, which may represent deeper inferences, improves this correlation.

In our 2005 corpus, correlations with "near transfer" learning seem to be weaker than correlations with "far transfer" learning, using a preliminary division of questions into "near" and "far". Also, correlations with "far transfer only" (14Q) learning improve with lower thresholds, becoming significant at a threshold of zero. This also supports a link between cohesion, inference and learning. Performance on "far transfer" tasks is often thought to be impeded if the original knowledge is encoded too simply [22]. Counting ties at low thresholds may partially measure the inferential elaboration which aids far transfer.

The comparative weakness of the 05 results, especially among high pre-testers is unexplained. However, the 05 corpus has a stronger pre-to-post test score correlation than the 03 corpus (.741(26Q) vs .462), suggesting that perhaps the tutorial dialog is responsible for correspondingly less of the learning gain. If so, we might also expect lower correlations between dialog features and learning. Other work [23] has also found these corpora to be quite different with respect to a suite of quantitative evaluation metrics.

8 Conclusions and Future Work

We have shown that measuring cohesion as the repetition of meanings, as opposed to only the repetition of words (or stems), improves the measure's correlation with learning among high pre-testers in one of our corpora of tutoring dialogs. Also, these correlations improve as we allow more semantically distant matches, which presumably require deeper inference to produce or understand.

Results from our other corpus suggest that counting semantic repetition cohesive ties predicts the deeper learning measured by "far transfer" problems.

In future work we hope to make a more formal division of questions into "near" and "far" to confirm the relationship between cohesion and far transfer learning. We are also currently extending this work to see if it will replicate in other corpora of tutoring dialogs. If these results replicate successfully, we will experiment with using tutor word choice to manipulate cohesion during tutoring.

Acknowledgments

This research is supported by the ONR (N00014-07-1-0039). We gratefully thank our anonymous reviewers, Bob Hausmann, Chas Murray, Mihai Rotaru and the ITSPOKE group for many helpful comments.

References

1. Bloom, B.S.: The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher 13, 4–16 (1984)
2. Merrill, D., Reiser, B., Ranney, M., Trafton, J.: Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. The Journal of the Learning Sciences 2, 277–306 (1992)

3. Graesser, A.C., Person, N., Magliano, J.P.: Collaborative dialogue patterns in naturalistic one-to-one tutoring. Applied Cognitive Psychology 9, 495–522 (1995)
4. Forbes-Riley, K., Litman, D., Huettner, A., Ward, A.: Dialogue-learning correlations in spoken dialogue tutoring. In: Proceedings 12th International Conference on Artificial Intelligence Education (AIED), Amsterdam, Netherlands (July 2005)
5. Litman, D.J., Rose, C.P., Forbes-Riley, K., VanLehn, K., Bhembe, D., Silliman, S.: Spoken versus typed human and computer dialogue tutoring. International Journal of Artificial Intelligence in Education 16, 145–170 (2006)
6. Forbes-Riley, K., Litman, D., Amruta Purandare, M.R., Tetreault, J.: Comparing linguistic features for modeling learning in computer tutoring. In: Proceedings 13th International Conference on Artificial Intelligence Education (AIED), Los Angeles (2007)
7. VanLehn, K., Graesser, A., Jackson, G., Jordan, P., Olney, A., Rose, C.: When are tutorial dialogues more effective than reading? Cognitive Science 30, 1–60 (2006)
8. Kalyuga, S., Ayres, P.: The expertise reversal effect. Educational Psychologist 38, 23–31 (2003)
9. Conati, C., VanLehn, K.: Further results from the evaluation of an intelligent computer tutor to coach self-explanation. In: 5th International Conference on Intelligent Tutoring Systems, pp. 304–313 (2000)
10. McNamara, D.S., Kintsch, W.: Learning from text: Effects of prior knowledge and text coherence. Discourse Processes 22, 247–287 (1996)
11. Ward, A., Litman, D.: Cohesion and learning in a tutorial spoken dialog system. In: Proceedings of the 19th International FLAIRS Conference, pp. 533–538 (2006)
12. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to wordnet: An on-line lexical database. International Journal of Lexicography (special issue) 3(4), 235–312 (1990)
13. Rada, R., Mili, H., Bicknell, E., Bletner, M.: Development and application of a metric on semantic nets. IEEE Transactions on Systems, Man and Cybernetics 19(1), 17–30 (1989)
14. Sussna, M.: Word sense disambiguation for free-text indexing using a massive semantic network. In: CIKM 1993: Proceedings of the second international conference on Information and knowledge management, pp. 67–74. ACM, New York (1993)
15. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, August 1995, pp. 448–453 (1995)
16. Budanitsky, A., Hirst, G.: Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In: Workshop on WordNet and Other Lexical Resources, in the North American Chapter of the Association for Computational Linguistics (June 2001)
17. Graesser, A., Jeon, M., Yan, Y., Cai, Z.: Discourse cohesion in text and tutorial dialogue. Information Design Journal 15, 199–213 (2007)
18. Graesser, A.C., McNamara, D.S., Louwerse, M.M., Cai, Z.: Coh-metrix: Analysis of text on cohesion and language. Behavior, Research Methods, Instruments, and Computers 36, 192–202 (2004)
19. VanLehn, K., Jordan, P.W., Rose, C.P., Bhembe, D., Boettner, M., Gaydos, A., Makatchev, M., Pappuswamy, U., Ringenberg, M., Roque, A., Siler, S., Srivastava, R.: The architecture of why2-atlas: A coach for qualitative physics essay writing. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 158–167. Springer, Heidelberg (2002)
20. Halliday, M.A.K., Hasan, R.: Cohesion in English. English Language Series. Pearson Education Limited (1976)

21. Loper, E., Bird, S.: Nltk: The natural language toolkit (2002)
22. Stark, R., Mandl, H., Gruber, H., Renkl, A.: Instructional means to overcome transfer problems in the domain of economics: empirical studies. International Journal of Educational Research 31, 591–609 (1999)
23. Ai, H., Litman, D.: Comparing real-real, simulated-simulated, and simulated-real spoken dialogue corpora. In: Proceedings of the AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems, Boston, MA (2006)

Dialogue Modes in Expert Tutoring

Whitney L. Cade[1], Jessica L. Copeland[1], Natalie K. Person[1], and Sidney K. D'Mello[2]

[1] Department of Psychology, Rhodes College, 2000 North Parkway, Memphis, TN 38112 USA
{cadwl,copjl,person}@rhodes.edu
[2]Institute for Intelligent Systems, The University of Memphis, 365 Innovation Drive, Memphis, TN 38152 USA
sdmello@memphis.edu

Abstract. Previous studies have examined human-to-human dialogue in expert tutoring on the speech act level, but these analyses fail to provide the context necessary for understanding how a series of speech acts relate to each other. This research examined tutorial dialogue in terms of sustained, pedagogically distinct phases, referred to as tutoring "modes", which gives context to the finer-grained analysis of "moves". Our accomplishments were twofold: we developed a new annotation scheme for tutorial dialogue that takes into account clusters of multiple dialogue moves, and we determined the extent to which these modes occurred in the tutoring sessions. We also examined likely sequences of modes, all of which are important factors when building an ITS that reproduces the efforts of expert human tutors.

Keywords: dialogue modes, speech acts, tutoring, expert tutors.

1 Introduction

It is widely acknowledged that accomplished (expert) human tutors are highly successful in promoting active knowledge construction via one-on-one tutoring sessions [1]. According to Bloom, [1] expert human tutors produce effect sizes (2 sigma) that are quantitatively higher than those obtained by unaccomplished tutors (0.4) [2] and Intelligent Tutoring Systems (1.0 sigma) [3].

However, as of yet, we know very little about the teaching methods of expert tutors because the tutors in a vast majority of human tutoring studies were peer tutors, cross-age tutors, or paraprofessionals, untrained in tutoring skills with moderate domain knowledge [2]. The few studies that focus on expert tutors included six or fewer tutors, with the majority including only one or two experts [e.g. 4-6], thereby calling into question the generalizability of the findings. Furthermore, in some cases the degree of expertise of the tutors is questionable. For example, the expert tutors in some studies are Ph.D.s with extensive teaching and/or tutoring experience [7-10], whereas in others the experts are graduate students that work in tutoring centers [11]. Therefore, there is a need to document the pedagogical and motivational strategies of expert tutors to inform the development of expert tutor-based Intelligent Tutoring Systems (ITSs).

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 470–479, 2008.

Determining the appropriate level of granularity to study the skills and techniques utilized is still an open question. When the dialogue of tutors, expert and non-experts alike, are studied, the analysis usually takes place at a very fine-grained level, the speech act level. For example, Graesser, Person, & Magliono [12] studied novice tutors using a coding scheme that specified pedagogical dialogue moves (speech acts). Chi et al. [13] classified statements made in tutoring sessions at the move level in order to study the dominance of student or tutor in the course of a session. While Cromley & Azevedo [14] identified problem-solving episodes (a larger unit of analysis), this was really only used to understand correct or incorrect answer responses.

Simply put, tutorial dialogue studied at the move level has been the trend of the ITS community since the mid-90s, with more focus on the observed speech acts or dialogue moves than their contextualization. For example, the coding scheme formulated by Shah et al. [15] sought to understand the motivation of the tutor and student by coding for the immediate learning goals of the student and tutor, but such a coding scheme still did not define the context with larger units of analysis.

Little is known about how tutors use specific sequences of dialogue moves to achieve particular pedagogical goals. Identifying these dialogue move sequences will certainly expand the understanding of expert tutorial dialogue beyond what is currently known about individual dialogue moves (or speech acts). As mentioned earlier, a number of researchers have examined tutorial dialogue at the move level and have at times reported co-occurrences of some of the dialogue moves. This research adds to the tutorial discourse literature in that it had yielded an annotation scheme to study tutor and student dialogue move sequences in terms of reoccurring, larger phases of a tutoring session, which we have termed "tutoring modes", or more briefly "modes". We have used this annotation scheme to code a large sample of expert tutoring sessions (N = 40) in order to examine the occurrence and relationships between these modes.

1.1 An Overview of the Coding Scheme

The precise meaning of a speech act during a tutoring session is related to the larger overarching teaching phase of the tutoring session. Therefore, a mode can be considered to be the overarching context or teaching phase during which learning occurs, as well as a means of examining existing tutoring models. For example, consider the modeling-scaffolding-fading paradigm [16]. Using the definitions of each mode below, we can study how and when the modeling, scaffolding, and fading phases occur in an expert tutoring session.

In expert tutoring sessions, modes are always initiated by the tutor; in line with previous research, it is not surprising to observe that tutors control the flow of the task and information [15, 12]. In 40 expert tutoring sessions, we identified eight mutually exclusive tutoring modes that encapsulate virtually all the dialogue that occurs in human expert tutoring sessions.

- **Introduction:** This mode captures the non-teaching dialogue that occurs as both tutor and student exchange greetings and attempt to establish the agenda for the tutoring session.
- **Lecture:** The tutor explicitly dispenses domain information to the student.

- **Highlighting:** During problem-solving activities, the tutor highlights what the problem is asking for and the information that it provides. The tutor can also break down the steps involved in a particular problem-solving method, create a "game plan" for working a problem, and redirect the student to the right path when they have forgotten parts of the method and its purpose.
- **Modeling:** In this mode, the tutor works out a problem for the student. While the student may "participate" in the problem-solving process (i.e., do minor calculations that the tutor instructs them to do), the tutor takes the lead in solving the problem and initiates all of the problem-solving steps.
- **Scaffolding:** The tutor and student work out a selected problem together, each contributing portions of the answer that result in a solution. The tutor intervenes when necessary so that the student can successfully arrive at a correct solution.
- **Fading:** The student works an entire problem with virtually no aid from the tutor, although the tutor may comment from time to time on their progress.
- **Off Topic:** This is a portion of the session where the tutor and student are not engaged in the tutoring lesson for a significant amount of time.
- **Conclusion:** Like the *Introduction*, this mode captures the social pragmatics as the session comes to a close, which has little to do with the lesson.

2 Methods

We recruited eight expert math and science tutors to participate in this study, using stringent criteria to ensure tutor quality. The tutors were licensed to teach at the secondary level, had five or more years of tutoring experience, were employed by a professionally tutoring agency, and were highly recommended by school personnel. Thirty students who worked with these tutors consented to participate; forty percent of the students were female, and sixty percent were male in grades 7 through 12. One of the students was receiving tutoring in order to obtain a GED. Each student participated in one or two tutorial sessions, while each tutor participated in at most 8 tutoring sessions.

Forty one-hour tutoring sessions were videotaped in various locations such as homes or libraries, depending on where the tutor and student agreed to meet. Tutors were given an informed consent and questionnaires to ascertain their views on learning prior to any tutoring sessions, and students were given an informed consent to be signed by their parents and themselves before their sessions. Cameras were positioned in front of the tutor and student, who faced the camera so that the sound, facial expressions, and actions of both parties could be recorded. No researchers were present in the rooms during the tutoring sessions. All 40 of the videotapes were digitized and transcribed.

Two knowledgeable judges examined a subset of the tutoring transcripts and identified dialogue sequences that appeared to have different overarching purposes. After multiple passes through the subset of tutoring transcripts and various category sortings [17], the final annotation scheme with the eight categories described above was settled upon. The researchers established a series of coding guidelines prior to coding; all 40 transcripts were coded in accordance with these guidelines.

3 Results

Inter-rater reliability for each of the eight mode categories was computed using Cohen's kappa; the unit of analysis was at the turn level. Several rounds of coding were necessary to achieve Kappa thresholds of .80 for each mode category. The reliability statistics are reported in Table 1.

A frequency analysis revealed that 650 modes occurred in the 40 sessions. Therefore, on average there were 16.25 modes per session. As expected, each session began with an *Introduction* and ended with a *Conclusion*. *Lecture* and *Scaffolding* modes dominated the sessions with average frequencies of 3.4 and 4.8, respectively. The average frequencies for the other mode categories were as follows: *Highlighting* (2.1), *Modeling* (1.475), *Fading* (1.075), and *Off Topic* (1.475).

Table 1. Kappa Values and Proportions of Occurrence of the Various Modes

Mode	Kappa	Unweighted		Weighted	
		Mean	Stdev	Mean	Stdev
Scaffolding	.878	0.300	0.108	0.523	0.247
Lecture	.810	0.211	0.141	0.217	.232
Highlighting	.800	0.116	0.094	0.041	.048
Off topic	.830	0.099	0.105	0.049	0.061
Modeling	.975	0.077	0.079	0.084	0.119
Introduction	1.00	0.071	0.026	0.036	0.039
Conclusion	1.00	0.067	0.032	0.02	0.039
Fading	.964	0.059	0.074	0.029	0.045

For each session, we computed the proportion of occurrence for each mode. Because some modes are inherently longer (span more turns) than others, we computed the proportions in two ways: unweighted by turns and weighted by turn. Table 1 lists the descriptive statistics for the mode proportions when not weighted by turn. A repeated measures ANOVA revealed that there were statistically significant differences in the occurrence of each of the modes, $F(7,273) = 31.999$, $Mse = .009$, $p<.001$, partial-eta squared = .451. *Lecture* and *Scaffolding* modes were the most frequently occurring modes, collectively comprising 51% of the observations. Bonferroni posthoc tests confirmed that the occurrence of *Scaffolding* was equal to that of *Lecture* and significantly higher than the other modes ($p<.01$). *Lecture* was not significantly different from *Highlighting* and *Scaffolding* but occurred more often than the other modes. *Highlighting* occurred more frequently than *Fading* ($p<.05$). The overall pattern indicated that *Scaffolding* and *Lecture* modes dominate the tutoring session.

The descriptive statistics for the mode proportions that were weighted by turns also appear in Table 1. A repeated measures ANOVA revealed that there were statistically significant differences in the occurrence of each of the modes when number of turns was taken into account, $F(7,273) = 59.945$, $Mse = .020$, $p < .001$, partial-eta squared = .606. The *Lecture* and *Scaffolding* modes comprised the most conversational turns,

74% of the observations. Bonferroni posthoc tests showed that the number of turns devoted to *Scaffolding* was significantly greater than those devoted to *Lecture*, and was significantly higher than the number of turns that occurred in the other modes as well ($p<.01$). *Lecture* was not significantly different from *Modeling* ($p>.05$), but had significantly more turns than the other modes categories ($p<.01$). The number of turns devoted to the other modes were not significantly different ($p>.05$).

We computed the conditional probabilities of one mode followed by another in order to establish likely transitions between modes. In order to correct for baserate biases we utilized a likelihood metric depicted in Equation 1 below (for a more detailed explanation of how the test works, see [18]).

$$L[C \rightarrow X] = \frac{\frac{\Pr[X \cap C]}{\Pr[C]} - \Pr[X]}{1 - \Pr[X]} \tag{1}$$

According to the transition likelihood metric (L), if $L[C \rightarrow X] \approx 1$, we can conclude that mode X always follows mode C above and beyond the prior probability of mode C. If, on the other hand, if $L[C \rightarrow X] \approx 0$, then X follows C at the chance level. Furthermore, if $L[C \rightarrow X] < 0$, then the likelihood of mode X following mode C is much lower than the base rate of mode X. Therefore, in order to detect significant mode transitions we compared the likelihood of each mode occurring to a hypothesized mean of 0 (base rate) using a one-sample t-test.

Table 2. Mode Transitions

Current Mode	**Next Mode**							
	Intro	Lecture	Highl-lighting	Modeling	Scaffold	Fading	Offtopic	Conclusion
Intro	nt.	+					nt.	nt.
Lecture	nt.	nt.			+		+†	
Highl.	nt.		nt.		+			
Model	nt.			nt.				
Scaffold	nt.	+	+		nt.			
Fading	nt.					nt.		
Offtopic	nt.						nt.	nt.
Concl.	nt.	nt.	nt.	nt.	nt.	nt.	nt.	nt.

\+ mode transition significantly greater than base rate (chance).
\- mode transition significantly less than base rate (chance).
nt. mode transition not tested, † marginally significant.

With eight modes, there are 64 possible combinations of mode transitions. However, not all mode combinations are possible in our coding scheme. For example, *Introduction* mode could never be immediately followed by *Conclusion* mode. Twenty-four impossible mode combinations were eliminated, leaving 40 meaningful mode transitions. We tested the significance of each of the 40 mode transitions by performing a one-sample two-tailed t-test that compared the transition likelihood to a

hypothesized mean of 0 (the chance level). A Bonferroni correction was applied to the significance threshold, making the effective significance level p<.00125 (0.05/40).

From these 40 tests, only a handful of transitions were significant, as seen in Table 2. The interpretations of these analyses appear in the Discussion section of this paper.

4 Discussion

Researchers in the human tutoring and ITS community have hypothesized that modeling-scaffolding-fading [16], Socratic tutoring [19], contingent teaching (A ZPD approach) [20], cognitive apprenticeship [21, 6], transmission and information-delivery models, inquiry teaching [22], situated learning [21], anchored instruction [23, 24], case-based reasoning [25], coaching [26], reciprocal teaching [27] are the theoretical models and strategies that are preferred by expert tutors. However, it is difficult to directly test these theories when a tutoring session is analyzed at the dialogue move level. With the new coding scheme developed here, we hope to have laid the foundation for a systematic evaluation of the theoretical models of human tutoring.

4.1 Mode Frequencies

The frequency of each mode in a session is an important factor in determining the distribution and prevalence of the modes, regardless of their length and the amount of time that they consume. The number of times that a tutor chooses to use a mode is akin to the number of times a mechanic chooses to use a tool, and both mechanics and tutors employ their tools at the precise moment that they are needed. With future research, we hope discover the precise triggers that warrant the need for a particular mode, but for now, the number of times a mode is employed will give us some idea of its value and urgency.

Modeling and *Fading* occur with similar frequency in the average session, although *Modeling* appears slightly more than *Fading*. The relatively low frequency of *Modeling* mode may reflect the tutor's desire for the student to engage during problem-solving rather than passively watch. Many of our tutors reported that they believe in the power of practice when proficiency and mastery are the goals of learning, and this seems to be reflected in both the high frequency of *Scaffolding* and the lower frequencies of *Modeling* and *Fading* modes. The tutors were aware obviously that their struggling students needed a great deal of help, and so they employed hands-on exercises that gave the students ample opportunity to engage in supervised practice. As a consequence, *Modeling* occurred relatively few times in the sessions, just enough to demonstrate particular methods. It may be the case that the time constraints of a one-hour tutoring session precluded the tutors from spending too much time in the *Fading* mode as well. That is, tutors preferred to spend their instructional time interacting with students and adequately covering as much material as possible rather than having students pay money to sit and work problems on their own.

The *Highlighting* mode also appears a fair number of times and by definition only occurs during problem-solving activities (i.e., they are usually embedded between *Scaffolding* modes). The *Highlighting* mode is a means for tutors to elucidate the constraints of particular problems, emphasize relevant information, and alert students to

appropriate problem-solving strategies. *Highlighting* differs from *Lecture* in that *Lecture* is more theory-driven in its explanations of domain topics, and refers to problem-solving in only a hypothetical sense. The occurrences of *Lecture* and *Highlighting* did not significantly differ from each other, which may be explained by the tutor's sense that he/she must convey as much information as possible, both as a knowledge base and in reference to specific problem-solving strategies.

Despite that our tutors believed in practice and problem-solving (as indicated on their pre-tutoring questionnaires), Lecture mode occurred about three times per session, making it the second-most frequent mode. The prevalence of Lecture mode is interesting given the taboo associated with information transmission models of instruction and the popularity on active student learning. While tutors may claim to favor active problem-solving, time constraints may once again prevent tutors from relying solely on problems to transmit basic knowledge. Considering the substantial knowledge deficits of the students, tutors may opt to address some of these deficits with carefully timed Lectures , maximizing the amount of material covered in the session.

Scaffolding was the most frequently occurring mode despite not significantly differing from *Lecture*. *Scaffolding* occurred approximately five times in each session. According to our criteria, *Scaffolding* constitutes any significant joint tutor-student involvement in problem-solving. This mode encompasses many degrees of tutor and student involvements, and this wide range makes it more likely to appear than other problem-solving modes (e.g., *Modeling* or *Fading*). The abundance of *Scaffolding* modes provides evidence that the tutors do indeed value student engagement in the problem-solving process.

4.2 Modes in Terms of Length

While frequencies give an indication of how often a mode is employed, they do not speak to the length of time an expert tutor spends in these modes. The number of turns in each mode gives us a more detailed understanding of how long these modes actually take, as well as how significant each mode is in teaching their student. *Scaffolding*, *Lecture*, and *Modeling* were the three modes that occupied the most turns.

Scaffolding requires significantly more dialogue turns than any other mode, which speaks to the importance and time-consumption of problem-solving activities during tutoring sessions. Over 50 percent of the dialogue exchanged between the tutors and students occurred in the *Scaffolding* mode. This may be because two people working on one problem, exchanging ideas and seeking confirmation from each other, *requires* more dialogue. In any event, the sheer volume of dialogue devoted to *Scaffolding* does indicate that this mode is a valued process in constructing knowledge, and perhaps also in correcting it. During *Scaffolding*, students' misconceptions are exposed, and the tutor can recognize the source of the error or confusion and calibrate their comments and instruction accordingly.

To do this, however, tutors need to utilize other tools, and their tools of choice appear to be *Lecture* and *Modeling*. *Lecture*, the "just in time" information which accounts for the next largest percentage of dialogue moves, is one of the most straightforward ways to fill the knowledge voids exposed in a session. *Modeling* works in a different capacity; students may grasp the basic tenets of the topic, but their knowledge needs practical implementation. A single modeled problem can lay the

foundation for several scaffolded exercises, resulting in fewer turns being devoted to the *Modeling* mode.

4.3 Mode Transitions

Just as important as mapping out the length and frequency of each mode is understanding the relationships between the modes. Establishing likely mode transitions allows us not only to predict a mode sequence given any mode, but also provides a sequenced structure for building an ITS. The likely sequences of modes reveal several interesting connections, as well as several interesting insignificant findings that the tutoring literature predicts should have been significant. Once the mode sequences that are rendered impossible by the rules of the coding scheme are eliminated, only positive significant relationships exist. *Introduction* followed by *Lecture* mode appears to be the most common start to a tutoring session, which makes sense in the context of the tutor's desire to create a knowledge base before progressing to the problem-solving phase of the session. *Scaffolding* and *Lecture*, the two modes that occur the most, have a cyclical relationship, where both *Lecture* then *Scaffolding* and *Scaffolding* then *Lecture* are significant mode sequences. We can conclude with some confidence (due to our transition likelihood metric that effectively factors out baserate biases) that the cyclical relationship between *Lecture* and *Scaffolding* points to some underlying reason for tutors to use these modes in conjunction with each other as a valuable tool of the trade. A similarly bidirectional relationship exists between *Scaffolding* and *Highlighting*, and this mode pairing makes sense given the context that surrounds *Highlighting*. *Highlighting* usually occurs when tutor and student are collaborating on a problem so that the tutor can redirect the student when needed. This type of coding will allow future analysis to pinpoint the circumstances that surround or trigger the process of aiding a confused student.

Of equal interest here is the negative evidence presented by the table, which can be used to raise questions about theoretical models of learning. For instance, many theories, such as Modeling-Scaffolding-Fading theory [16], would seem to predict that there should be some sequential relationship between those three problem-solving modes, but none exists. While this begs further research, it does bring up one of the advantages of a higher-order coding scheme; such a scheme can be the vehicle by which tutoring models may be tested in future research. Many models claim higher-order patterns in tutoring sessions, though analysis up to now have been at a much smaller grain size, and so studying current theories with dialogue moves, modes, and information from tutor questionnaires may help validate or disconfirm these conceptually-based notions.

5 Application

The culmination of all this information is ultimately its application to intelligent tutoring system design. Current systems can be updated and revised based on the information about the general pattern of a tutoring session, and how tutors react given a certain student's response. This may lead to a tutoring session that tailors itself to

each student's specific needs at that moment in time, which optimizes the program's external validity and student learning.

Acknowledgments

This research was supported by a grant awarded by the U. S. Office of Naval Research (N00014-05-1-0241). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research.

References

1. Bloom, B.S.: The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher 13, 4–16 (1984)
2. Cohen, P.A., Kulik, J.A., Kulik, C.C.: Educational outcomes of tutoring: A meta-analysis of findings. American Educational Research Journal 19, 237–248 (1982)
3. Corbett, A., Anderson, J., Graesser, A., Koedinger, K., van Lehn, K.: Third generation computer tutors: Learn from or ignore human tutors? In: Proceedings of the 1999 Conference of Computer-Human Interaction, pp. 85–86. ACM Press, New York (1999)
4. Gilkison, A.: Techniques used by expert and non-expert tutors to facilitate problem-based learning tutorials in an undergraduate medical curriculum. Medical Education 37, 6–14 (2003)
5. Hay, P.J., Katsikitis, M.: The 'expert' in problem-based and case-based learning: Necessary or not? Medical Education 35, 22–26 (2001)
6. Lajoie, S.P., Faremo, S., Wiseman, J.: Tutoring strategies for effective instruction in internal medicine. International Journal of Artificial Intelligence and Education 12, 293–309 (2001)
7. Evens, M.W., Spitkovsky, J., Boyle, P., Michael, J.A., Rovick, A.A.: Synthesizing tutorial dialogues. In: Proceedings of the 15th Annual Conference of the Cognitive Science Society, Boulder. Lawrence Erlbaum Associates, Hillsdale (1993)
8. Glass, M., Kim, J.H., Evens, M., Michael, J.A., Rovick, A.A.: Novice vs. expert tutors: A comparison of style. In: Proceedings of the Midwest Artificial Intelligence and Cognitive Science Conference, Bloomington (1999)
9. Graesser, A.C., Wiemer-Hastings, P., Wiemer-Hastings, K., Harter, D., Person, N.: TRG: Using latent semantic analysis to evaluate the contributions of students in AutoTutor. Interactive Learning Environments 8, 129–148 (2000)
10. Jordan, P., Siler, S.: Student initiative and questioning strategies in computer-mediated human tutoring dialogues. In: Paper presented at ITS Workshop on Empirical Methods for Tutorial Dialogue Systems, San Sabastian, Spain (2002)
11. Fox, B.: Cognitive and interactional aspects of correction in tutoring. In: Goodyear, P. (ed.) Teaching knowledge and intelligent tutoring, Ablex, Norwood (1991)
12. Graesser, A.C., Person, N.K., Magliano, J.P.: Collaborative dialogue patterns in naturalistic one-on-one tutoring. Applied Cognitive Psychology 9, 359–387 (1995)
13. Chi, M.T.H., Siler, S.A., Jeong, H., Yamauchi, T., Hausmann, R.G.: Learning from human tutoring. Cognitive Science 25, 471–533 (2001)
14. Cromley, J.G., Azevedo, R.: What Do Reading Tutors Do?: A Naturalistic Study of More and Less Experienced Tutors in Reading. Discourse Processes 40(2), 83–113 (2005)

15. Shah, F., Evens, M., Michael, J., Rovick, A.: Classifying student initiatives and tutor responses in human keyboard-to-keyboard tutoring sessions. Discourse Processes 33(1), 23–52 (2002)
16. Rogoff, B., Gardner, W.: Adult guidance of cognitive development. In: Rogoff, B., Lave, J. (eds.) Everyday cognition: Its development in social context, pp. 95–116. Harvard University Press, Cambridge (1984)
17. Corbin, J., Strauss, A.: Grounded theory research: Procedures, canons, and evaluative criteria. Qualitative Sociology 13, 3–21 (1990)
18. D'Mello, S.K., Taylor, R., Graesser, A.C.: Monitoring Affective Trajectories during Complex Learning. In: McNamara, D.S., Trafton, J.G. (eds.) Proceedings of the 29th Annual Cognitive Science Society, Cognitive Science Society, Austin, pp. 203–208 (2007)
19. Collins, A., Warnock, E.H., Aeillo, N., Miller, M.L.: Reasoning from incomplete knowledge. In: Bobrow, D.G., Collins, A. (eds.) Representation and understanding, pp. 383–415. Academic Press, New York (1975)
20. du Boulay, B., Luckin, R.: Modeling human teaching tactics and strategies for tutoring systems. International Journal of Artificial Intelligence in Education 12(3), 235–256 (2001)
21. Collins, A., Brown, J.S., Newman, S.E.: Cognitive apprenticeship: Teaching the craft of reading, writing, and mathematics. In: Resnick, L.B. (ed.) Knowing, learning, and instruction: Essays in honor of Robert Glaser, pp. 453–494. Lawrence Erlbaum Associates, Inc., Hillsdale (1989)
22. Dillon, T.J.: Questioning and teaching: A manual of practice. Teachers College Press, New York (1988)
23. Bransford, J.: Anchored Instruction. The Adventures of Jasper Woodbury (1989), http://peabody.vanderbilt.edu/projects/funded/jasper/intro/Jasperintro.html
24. Crews, T., Biswas, G., Goldman, S., Bransford, J.: Anchored Instruction. In: Anchored Interactive Learning Environments (1997), http://www.vuse.vanderbilt.edu/~biswas/Research/ile/papers/postscript/advplay.pdf
25. Feltovich, P.J., Spiro, R.J., Coulson, R.L.: The nature of conceptual understanding in biomedicine: The deep structure of complex ideas and the development of misconceptions. In: Evans, D.A., Patel, V.L. (eds.) Cognitive science in medicine: Biomedical modeling, pp. 113–172. MIT Press, Cambridge (1989)
26. Brown, J.S., Burton, R.R.: Diagnostic models for procedural bugs in basic mathematical skills. Cognitive Science 2, 155–192 (1978)
27. Palinscar, A.S., Brown, A.: Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. Cognition & Instruction 1, 117–175 (1984)
28. Robson, C.: Real word research: A resource for social scientist and practitioner researchers. Blackwell, Oxford (1993)

Seeing the Face and Observing the Actions: The Effects of Nonverbal Cues on Mediated Tutoring Dialogue

Federico Tajariol[1], Jean-Michel Adam[2], and Michel Dubois[3]

[1] R & D Orange Labs, France Telecom Groups, Lannion, France
tajariol@gmail.com
[2] Laboratory Informatics Grenoble, University of Grenoble, France
adam@imag.fr
[3] Laboratory of Social Psychology, University of Grenoble, France
Michel.Dubois@upmf-grenoble.fr

Abstract. Mediated communication technologies, conveying verbal and nonverbal cues, are more and more employed in learning activities. Nevertheless, their effects on teacher-student interaction have been not clearly stated yet. Through two experimental studies, we investigated on the effects of nonverbal communication cues (kinesic and ostensive-inferential) on synchronous mediated tutoring dialogue, in which a tutor and a student communicate through audio-video communication tools. The outcomes show that kinesic cues lead tutor to monitor more carefully learner's ongoing task and to encourage much more them, while ostensive-inferential cues improve learner's task performance and lead both tutor and student to focus better on tutoring speech acts.

Keywords: audio-video mediated communication, nonverbal communication, kinesic cues, ostensive-inferential cues, tutoring dialogue.

1 Introduction

Mediated communication technologies are more and more used in several human activities, also in teaching environments. These technologies allow distant participants, such a tutor and students, to see one another during the interaction, to work together on any shared documents [1], or just to observe the actions they make in their own working environment [2]. Even if many organizations and pedagogical institutions have already introduced these tools in their education services, the effects on the cognitive dimensions of the communication between teachers and students have not been clearly established yet [3].

The aim of this paper is to understand the effect of audio-video mediated technologies on a specific type of communication, which is the mediated tutoring dialogue. In the first section, we define tutoring dialogue as a joint communication activity and we show the role of nonverbal cues. We then describe our general method and we present two experimentations we conducted on the affordances of two non-verbal cues (kinesic and ostensive-inferential) on synchronous mediated tutoring dialogue.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 480–489, 2008.

2 Tutoring Dialogue and Non Verbal Cues

As human beings, we use both verbal and nonverbal languages to perform activities, as creating, teaching, etc. When we communicate with our partners, we cannot directly know their thoughts, feelings and intentions; we are just able to infer them by interpreting their utterances [4] and nonverbal behavior [5]. Nevertheless, communication is not a mere sending-receiving messages activity, but it is rather a cooperative action [6]. For example, in tutoring dialogue, both tutor and student coordinate their turns to ground on a mutual understanding, so that tutors may enable students to contextualize their own problem statements and to improve their knowledge [7]. Concerning the nonverbal cues, we may distinguish kinesic cues (e.g., facial expressions, postures) and ostensive-inferential cues (e.g., actions and deictic gestures) [8]. Kinesic cues ensure the conversational floor between tutor and student, and moreover, they inform each participant about feelings and intentions of the other person [8]. Ostensive-inferential cues facilitate the verbal referring process, helping participants to coordinate their actions and to anticipate the other's needs. In face-to-face tutoring dialogue setting, nonverbal cues are immediately available to both tutor and student: for instance, by observing students' facial expression tutors infer when to help them without disturbing needlessly [7]. In a video-mediated setting, according to the social presence theory [10, 11], it would be sufficient to put all nonverbal cues at tutor's and student's disposal to allow a suitable and efficient tutoring dialogue. Nevertheless, this does not seem the best solution. In fact, on the one hand, kinesic cues help distant participants to establish a mutual understanding [12, 13, 14] and ostensive-inferential cues support participants to perform a mediated activity more quickly [15]; on the other hand, nonverbal cues do not always help participants to better perform their activities [16, 17, 18]. However, amongst mediated-communication studies, just a few specifically concern mediated tutoring dialogue. This lack of studies would need priority status in the agenda research on mediated tutoring environments.

3 Research Problem

These considerations on the effects of kinesic and ostensive-inferential cues lead us to investigate on the following issues: which type of nonverbal cue improves the tutor-student learning interaction? Would it be better that the tutor and the student see each other (kinesic cues)? Would it be better that the tutor observes the student's actions to improve (her) his learning (ostensive-inferential cues)? To answer these questions, we conducted two experimental studies to understand the effects of kinesic cues (study 1) and ostensive-inferential cues (study 2) on tutoring dialogue.

4 General Method

The two experiments we will report followed a same experimental task, same apparatus, same procedure and same dependent measures.

4.1 Task, Apparatus and Procedure

Task & Apparatus. Tutors and students were involved in a sort of practical pedagogical work. The tutor had to help two students to learn basic commands of HTML language and to create an easy web page. The two students had to edit some sentences of a text in bold and italic, as well as building an internal link. The tutor had to help each student spontaneously or when a student asked for help, yet (s)he could help only one student at a time, as in a dyadic tutoring situation: while (s)he was communicating to a student, the other student could not hear their dialogue. The tutor and the two students settled down in three separate rooms, each one was equipped with a personal computer (central unit and monitor) and with a monitor for audio-video communication. Each computer was set with a web browser and a simple text editor. Anytime the tutor wanted to communicate with a student, (s)he chose the student by pressing a button on an *ad hoc* interface. Both students could ask the tutor to help them anytime, sending him/her a standard message by means of chat software.

Procedure. Each experimental session lasted nearly an hour, and it included three main phases. In the preparing phase (15 minutes), the tutor met the two students and the researcher explained them the aims of the experimentation. Then, the tutor and each student reached their own working room. Each subject filled in a consent form and was briefed on the main functions of the apparatus (e.g., the button to use to start a call, the folder containing the html files, etc.). Next, each student answered a pre-test to evaluate her/his HTML knowledge. Then the experimental phase started (35 minutes). Each student received a four-page HTML manual, which contained some HTML basic commands. Of course, the students needed their tutor's help to design

Table 1. – Tutoring dialogue coding scheme

Mutual understanding		
Role	**Category**	**Examples**
Tutor	To accept student's utterance	T: "*yes*" "*ok, right*"
	To check student's understanding	T: "*is it clear now?*", "*is it ok?*"
Student	To accept tutor's utterance	S: "*yes*", "*ok*"
	To check tutor's utterance	S: "*could you repeat, please?*", "*what?*"
Tutor's and Student's tutoring intrinsic speech acts		
Role	**Category**	**Examples**
Tutor	To find out student's ongoing task	T: "*Did you try moving it on the red icon?*"
	To help student	T: "*Close the window and open the other file*"
	To encourage student	T: "*That's good you've nearly finished*"
Student	To give tutor information about ongoing task	S: "*I still have to finish this part of the exercise*"
	To ask tutor's help	S: "*Is the I tag in the HEAD part of the text?*"

the web page. When time ended, each subject was asked to complete a post-test questionnaire (same questions of the pre-test), then (s)he was debriefed and dismissed.

4.2 Measures

We transcribed verbatim all experimental sessions that we had videotaped. Based on studies concerning the grounding processes in communication [6], affordances of

visual information in mediated communication [1, 9] and tutoring dialogues [19], we created the following coding scheme: *a*) the *tutor's proactive behavior*: we distinguished the tutor's spontaneous interventions towards the student as *proactive interventions* from the tutor's *reactive interventions* when s(he) replied to a student's call; *b*) the *mutual understanding*: we categorized all verbal markers that students and tutors had used during their dialogue to ground their mutual understanding (Table 1); *c*) the *tutoring intrinsic speech acts*: we categorized the tutors' and students' speech acts related to the intrinsic nature of the tutoring dialogue (Table 1). Three trained researchers coded the transcribed dialogues. We checked the reliability of coding by means of the reproducibility test, obtaining an average value for students' acts (K=.58) and a high value for tutors' speech acts (K=.82). d) *Students' task performance*: we scored the web page that students had realized.

5 Experiment 1: The Effects of Kinesic Cues on Tutoring Dialogue

5.1 Hypotheses

H1) Given the great difficulty of the task for a HTML beginner, we made the hypothesis that if the tutor could observe the students' faces, (s)he would infer their difficulties during the practical work and then (s)he would be *more proactive* in helping them without waiting for their call .

H2) We supposed that mutual understanding would be easier if the students and the tutor could see each other: we expected that the number of verbal markers for the common ground process would be fewer when the participants could see each other than when they could not.

H3) Consequently, we expected that their dialogue would be grounded on intrinsic tutoring speech acts and that intrinsic tutoring speech acts would be more numerous when participants could see each other than when they could not.

H4) We expected that students' performance would improve when tutor could see their faces while they were performing the task, because tutor could help them in a suitable manner.

5.2 Participants

We recruited 12 tutors, half men and half women (age M=30.9, S.D.=7.7). All of them were computer scientists owning good skills in HTML programming. We also recruited 48 undergraduate students in psychology (36 women and 12 men; age M=24.1, S.D.=6.2), all of them unskilled in HTML programming.

5.3 Experimental Conditions

We set the following conditions: c1) *audio only*: tutor and students could only talk to each other through the audio channel; c2) *audio & human face*: the tutor and each of the two students could see each other's face and upper torso on a personal monitor, and they could talk to each other by means of the audio channel.

Each tutor performed in both conditions (within-participants experimental design), whereas half students were assigned to one of the two conditions (between- partici-

pants design). We controlled that the age of the students involved in the two conditions were equivalent (t(46)=1.318;n.s.) and so was the HTML knowledge (t(46)=0.09;n.s.). To guard against order effects, we counterbalanced the tutors' performing order setting.

5.4 Main Outcomes

We conducted independent *t-test* and we calculated the value of the effect size *d*.

H1) Tutors' proactive interventions were more in the *audio & human face* than in the *audio only* condition (M = 4.6 (2) vs. 1.8 (0.9), t(22) = 4.27; $p<.001$, $d = 1.8$). H1 was confirmed.

H2) Concerning the mutual understanding, tutor had significantly produced fewer verbal markers to check students' understanding in *audio & human face* than in *audio only* condition (M=3.9 (3.6) *vs*. 10 (6.4), t(22)=2.85; $p<.01$, $d=1.17$). About other students' and tutors' verbal markers, we did not find any statistically significant differences between the two conditions. H2 was partially confirmed.

H3) Concerning the speech acts referred to tutoring dialogue, we present the outcomes about tutors' speech acts and then the outcomes about students' speech acts.

i) Tutors produced significantly more speech acts to encourage students in *audio & human face* than in *audio only* condition (M=3.2(1.3) *vs*. 1.2(1.2), t(22)=2.9; $p<.01$, $d=1.14$). Moreover, the number of tutors' speech acts oriented to know students' ongoing task was significantly higher in *audio & human face* than in *audio only* condition (M=60.7 (33.2) *vs*. 35.3 (8.1), t(22)=4.27; $p<.02$, $d=1.05$).

ii) Consequently, students' speech acts oriented to give tutors any details about the ongoing task was significantly higher in *audio & human face* than in *audio only* condition (M=63.6 (28.4) *vs*. 43.8 (14.4), t(22)=2.15; $p<.05$, $d=0.88$).

No statistically significant differences between the two conditions were found about other speech acts. H3 was partially confirmed.

H4) Concerning the learning score, no significant differences were found between the two conditions (*audio only* M=4.2 (1.4) *vs. audio & video-person* M= 4.6 (1.5), t(22)=0.83; n.s., $d=1.1$). H4 was rejected.

5.5 Discussions

The aim of this first experimentation was to measure the effects of kinesic cues on several measures, such as the tutor's proactive behavior, the mutual understanding between the tutor and the students, the intrinsic tutoring speech acts and the student's task performance. Concerning the tutor's proactive behavior, the tutor assisted more spontaneously the students without waiting for their help request when tutor and students could see each other. This outcome confirms that, as in face-to-face setting [7], tutors take the floor before students explicitly produce an aid request. Moreover, tutors could easily check the students' understanding level by just observing their faces. This outcome corroborates that the addressees' face would help to be aware of their understanding [12]. Even if, however, the students' verbal markers are not significantly different between the two conditions, we must consider that tutors have generally the floor in tutoring dialogue and that students play the role of a reactive addressee rather than the leading role [7]. On the other hand, it could be possible that, given the difficulty of the task, they chose to check tutor's utterances by using verbal

markers in order to be sure of the content of tutor's aid. The outcomes concerning the students' task performance imply that human faces do not improve the performance in procedural tasks [3, 17, 18].

6 Experiment 2: The Effects of Ostensive-Inferential Cues

The experimental method of this experiment was the same than for the first one.

6.1 Hypotheses

H1) We expected the tutor would be more proactive when (s)he could observe the students' actions, because (s)he was always aware of their difficulties and (s)he could take the floor before they asked for help.

H2) We expected that the number of verbal markers for mutual understanding would be fewer if tutors could observe the students' actions.

H3) We expected that the number of intrinsic tutoring speech acts would be higher when the tutors could observe the students' actions rather than when they could not.

H4) We also predicted the students' performance would improve when the tutor could observe their actions during the practical work session, because (s)he could choose the most suitable help to their needs.

6.2 Participants

We recruited the same tutors (N = 12) as for the first study, and 72 undergraduate students in psychology (50 women and 22 men; age M=23.8, S.D.=5.1), all of them unskilled in HTML programming.

6.3 Experimental Conditions

We set the three following conditions: c1) *audio & human face*: the tutor and the students could see each other's face and upper torso on the screen and they talked to

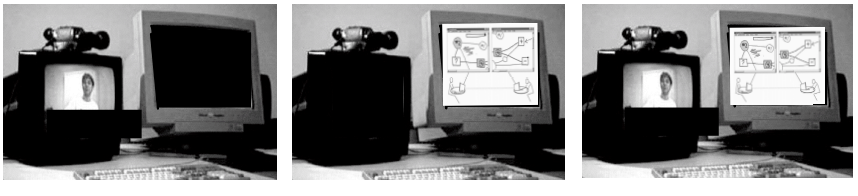


Fig. 1. - Experimental conditions study 2 (from left): audio & human face *vs*. audio & student's action *vs*. audio & human face & student's actions

each other through the audio channel; c2) *audio & student's actions*: the tutor and the students could talk through the audio channel, and the tutor could observe both students' computer screens (by means of VNC™ software). So, the tutor could observe the actions that each student was performing on their own computer desktop and (s)he could make the decision to help them; c3) *audio & human face & student's actions*:

the tutor could observe both students' face and upper torso and their computer screens, while (s)he could talk to each other by means of the audio channel.

As for the first study, each tutor performed all conditions (within-participants design) and we counterbalanced the order of the tutors' performance to guard against order effects. The students were equally assigned to the three experimental conditions. We controlled their age ($F(2, 69) = 1.73$; n.s.) and their HTML knowledge ($F(2, 69) = 0.16$; n.s.) in each of the three groups.

6.4 Main Outcomes

We conducted an analysis of variance (ANOVA) and *a priori* comparisons between conditions.

H1) About the tutors' proactive behaviour, the number of their spontaneous interventions were significantly different between conditions ($F(2, 33) = 4.671$; $p < .02$, $\eta^2 = 0.22$). In fact, the tutors were significantly more proactive in *audio & human face & student's actions* than in *audio & student's actions* condition (M = 6.6 (1.9) *vs*. 4.6 (2), $t(33) = 2.366$; $p < .02$, $d = 1.02$) and than in audio & video-person condition (M = 6.6 (1.9) *vs*. 4.2 (2.2), $t(33) = 2.859$; $p < .007$, $d = 1.16$). H1 was confirmed.

H2) About mutual understanding, the number of tutors' and students' verbal markers used to ground the understanding are not significantly different between conditions. H2 was rejected.

H3) About tutoring speech acts, we will first present outcomes for tutors (i) and then for students (ii).

i) Tutors. One-way ANOVA showed that the number of tutors' speech acts to know the students' ongoing task was significantly different between conditions ($F(2,33) = 4.529$; $p < .02$, $\eta^2 = 0.21$). Tutors produced fewer speech acts to know the students' task in *audio & student's actions* than in *audio & human face* condition (M = 27.4 (7.6) *vs*. 35.3 (8.1), $t(22) = 2.8$; $p < .01$, $d = 1$) and fewer in *audio & human face & student's actions* than in *audio & human face* condition (M = 26.6 (6.8) *vs*. 35.3 (8.1), $t(22) = 2.6$; $p < .005$, $d = 1.16$). Moreover, the tutors' speech acts to help the student were significantly different between conditions ($F(2, 33) = 15.248$; $p < .000$, $\eta^2 = 0.48$). In fact, tutors' speech acts to help students were fewer in *audio & human face* than in *audio & student's actions* (M=14.8 (10.4) *vs*. 40.2 (12.9), $t(33) = 4.99$; $p < .000$, $d = 2.18$) and fewer in *audio & human face* than in *audio & human face & student's actions* (M=14.8 (10.4) *vs*. 37.9 (13.8), $t(33) = 4.55$; $p < .000$, $d = 1.91$). Finally, concerning the tutors' speech acts to encourage students, their number was significantly different between conditions ($F(2, 33) = 4.813$; $p < .01$, $\eta^2 = 0.22$). In fact, tutors more often encouraged the students in *audio & human face* than in *audio & student's actions* condition (M=3.2 (2.3) *vs*. 0.8 (1.1), $t(33) = 2.69$; $p < .01$, $d = 1.33$) and more often in *audio & human face & student's actions* than in *audio & human face* (M=3.2 (2.8) *vs*. 0.8 (1.1), $t(33) = 2.69$; $p < .01$, $d = 1.12$).

ii) Students. About the students' speech acts to give tutors information about the ongoing task: the one-way analysis of variance showed significant differences between conditions ($F(2, 33) = 4.529$; $p < .02$, $\eta^2 = 0.21$). Students less often gave their tutors information about their ongoing task in *audio & human face & student's actions* than in *audio & human face* condition (M=26.5 (14.1) *vs*. 43.8 (14.4), $t(22) = 2.5$; $p < .01$,

d=1.21) and less often in *audio & student's actions* than in *audio & human face* condition (M=27.7 (18.4) *vs*. 43.8 (14.4), t(22)=2.6; *p*<.005, *d*=0.97). All other students' speech acts did not differ between conditions. H3 was partially confirmed.

H4) The students' web page score was significantly different between conditions ($F(2, 69) = 5.776$; $p < .005$, $\eta^2 = 0.14$). The students performed significantly better in *audio & human face & student's actions* than in *audio & human face* condition (M = 5.9 (1.6) *vs*. 4.5 (1.5), t(69)=3.27; *p*<.002, *d*=0.88) and better in *audio & student's actions* than in *audio & human face* condition (M=5.5 (1.2) *vs*. 4.5 (1.5), t(69)=2.4; *p*<.02, *d*=0.73). H4 was confirmed.

6.5 Discussions

The second study focused on the effects of ostensive-inferential cues referring to student's actions on tutoring dialogue. The overall outcomes showed: i) concerning the tutor's proactive behavior, the effect of ostensive-inferential cues was not as important as we expected. In fact, the number of tutor's proactive interventions produced with ostensive-inferential cues was nearly the same as produced with kinesic cues only. It appeared that ostensive-inferential cues only did not lead tutors to increase their proactive contributions. In fact, the tutors' proactive behavior would increase if both the kinesic and ostensive-inferential cues were available to the tutors. Against our expectations, ostensive-inferential cues did not improve the mutual understanding. Moreover, kinesics and ostensive-inferential cues support in an equivalent manner the mutual understanding. This outcome suggests that even if the tutor could observe both students' faces and actions, (s)he checked their understanding through verbal markers, and so corroborating tutors' behavior in a face-to-face setting [20]. However, in other mediated communication activities [1, 15], ostensive-inferential cues decreases verbal markers used for mutual understanding. This would suggest that the properties of ostensive-inferential cues have different effects on mutual understanding between distant partners, depending on the characteristics of the specific joint activity. Besides, when ostensive-inferential cues are available, tutors' questions to know the students' task status decreased and tutors' speech acts to help students increased. We highlight that the number of tutors' encouraging speech acts was higher with kinesic cues than with ostensive-inferential cues. This suggests that through kinesic cues the content of tutoring dialogue is oriented towards socio-relational issues rather than task issues. Concerning the students' task performance, ostensive-inferential cues improved students' task performance better than kinesic cues did.

7 Conclusion

The aim of these two experiments was to understand the effects of nonverbal cues, such as kinesic and ostensive-inferential cues, on mediated tutoring dialogue. Main outcomes of experiment 1 showed that when the tutor and the students could see each other's face, the tutor more often took the floor spontaneously, (s)he produced fewer verbal markers to check students' understanding and (s)he more willingly encouraged them. A possible interpretation is that kinesic cues, increasing the level of social presence, led tutor to monitor more carefully learners' ongoing task and to encourage them, helping both students and tutors in mutual understanding process. Outcomes

from experiment 2 partially corroborated the first ones, and also confirmed some previous researches (e.g., kinesic cues improve proactive behavior) [9]. Moreover, ostensive-inferential cues let the tutor and the student focus on intrinsic tutoring contents, and improved student's performance. Although these outcomes should be considered under some limits (e.g., visual parallaxes sometimes lowered the quality of the interaction between participants), if we are asked to suggest some tips for the design of tools supporting synchronous mediated tutoring dialogue, we argue that: a) if we want the tutor to be proactive, it would be better to let the tutor and the student see each other; b) if we want the student to improve learning, it would be better to let the tutor observe the student's actions. These outcomes show that ostensive-inferential cues improve the learning performance of students involved in procedural task (practical works, training in using software, etc.) It would be necessary to corroborate their effects on declarative content tasks. More researches would be needed to better understand the effects of the interaction between the different nonverbal cues, as well as to allow an efficient tutoring dialogue.

References

1. Kraut, R.E., Fussell, S.R., Siegel, J.: Visual information as a conversational resource in collaborative physical tasks. Human Comp. Inter. 18, 13–49 (2003)
2. Gergle, D., Kraut, R.E., Fussell, S.R.: Language efficiency and visual technology: minimize effort with visual information. J. Lang. Soc. Psyc. 23, 491–517 (2004)
3. Masoodian, M., Apperley, M., Frederickson, L.: Video support for shared work-space interaction: an empirical study. Inter. Comp. 7, 237–253 (1995)
4. Sperber, D., Wilson, D.: Relevance. Blackwell, Oxford (1987)
5. Argyle, M., Graham, J.: The Central Europe Experiment: looking at persons and looking at things. J. Envir. Psyc. Nonv. Behav. 1, 6–16 (1977)
6. Clark, H.H.: Using languages. Cambridge University Press, Cambridge (1996)
7. Fox, B.A.: The Human Tutorial Dialogue Project: Issues in the Design of Instructional Systems. Lawrence Erlbaum, Hillsdale (1993)
8. Manusov, V., Patterson, M.L.: The handbook of nonverbal communication. Sage Publications (2006)
9. Karsenty, L.: Cooperative work and shared visual context: an empirical study of comprehension problems in side-by-side and remote help dialogues. Hum. Comp. Inter. 14, 283–315 (1999)
10. Daft, R.L., Lengel, R.H.: Organisational information requirements, media richness and structural design. Management Science 32, 554–571 (1986)
11. Whittaker, S.: Theories and methods in mediated communication. In: Graesser, A.C., Gernsbacher, M.A., Goldman, S.R. (eds.) Handbook of discourse processes, pp. 123–164. Lawrence Erlbaum, Hillsdale (2002)
12. Doherty-Sneddon, G., Anderson, A., O'Malley, C., Langton, S., Garrod, S., Bruce, V.: Face-to-face interaction and video mediated communication: a comparison of dialogue structure. J. Exp. Psyc.: Applied 3, 105–125 (1997)
13. Isaacs, E., Tang, J.C.: What video can and cannot do for collaboration. Multim. Syst. 2, 63–73 (1994)
14. Whittaker, S., Geelhoed, E., Robinson, E.: Shared workspaces: how they work and when are they useful? Inter. J. Man-Mach. Studies 39, 813–842 (1993)

15. Gergle, D., Kraut, R.E., Fussell, S.R.: Language efficiency and visual technology: minimize effort with visual information. J. Lang. Soc. Psyc. 23, 491–517 (2004)
16. Anderson, A.H., Newlands, A., Mullin, J., Fleming, A.M., Doherty-Sneddon, G., Van Der Velden, J.: Impact of video-mediated communication on simulated service encounters. Inter. Comp. 8, 193–206 (1996)
17. Doherty-Sneddon, G., Anderson, A., O'Malley, C., Langton, S., Garrod, S., Bruce, V.: Face-to-face interaction and video mediated communication: a comparison of dialogue structure. J. Exp. Psyc.: Applied 3, 105–125 (1997)
18. O'Malley, C., Langton, S., Anderson, A., Doherty-Sneddon, G., Bruce, V.: Comparison of face-to-face and video-mediated communication. Inter. Comp. 8, 177–192 (1996)
19. Chi, M.T.H., De Leeuw, N., Chiu, M.H., Lavancher, C.: Eliciting self-explanations improves understanding. Cogn. Science 18, 439–477 (1994)
20. Van Lehn, K., Siler, S., Murray, C., Yamauchi, T., Baggett, W.B.: Why do only some event cause learning during human tutoring? Cogn. Inst. 21, 209–249 (2003)

Affective Transitions in Narrative-Centered Learning Environments

Scott W. McQuiggan, Jennifer L. Robison, and James C. Lester

Department of Computer Science, North Carolina State University, Raleigh NC 27695
{swmcquig,jlrobiso,lester}@ncsu.edu

Abstract. Affect has been the subject of increasing attention in cognitive accounts of learning. Many intelligent tutoring systems now seek to adapt pedagogy to student affective and motivational processes in an effort to increase the effectiveness of tutorial interaction and improve learning outcomes. To this end, recent work has begun to investigate the emotions experienced during learning in a variety of environments. In this paper we extend this line of research by investigating the affective transitions that occur throughout narrative-centered learning experiences. Further analysis differentiates the likelihood of affective transitions stemming from pedagogical agent empathetic responses to student affect.

1 Introduction

Affect has begun to play an increasingly important role in intelligent tutoring systems. The ITS community has seen the emergence of work on affective student modeling [8], detecting frustration and stress [6, 19, 24], modeling agents' emotional states [1, 15], devising affectively informed models of social interaction [16, 21, 23], detecting student motivation [25], and diagnosing and adapting to student self-efficacy [5]. All of this work seeks to increase the fidelity with which affective and motivational processes are understood and utilized in intelligent tutoring systems in an effort to increase the effectiveness of tutorial interactions and, ultimately, learning.

Recent work seeking to characterize the affective experience of learners interacting with intelligent learning environments has considered student affective trajectories occurring during learning. D'Mello *et al.* [11] studied the likelihood of affective transitions among six affective states (boredom, flow, confusion, frustration, delight, and surprise) that were found to be relevant to complex learning [9]. In general, learners are likely to persist in the same affective state (e.g., transitioning from a state of boredom to boredom is likely, and in some cases, significantly more likely than transitioning to another affective state). This analysis was conducted in the AutoTutor learning environment [9, 11]. Baker *et al.* were able to replicate many of D'Mello *et al.*'s [11] findings when they calculated the likelihood of affective transitions in the Incredible Machine: Even More Contraptions, a simulation-based learning environment [3]. Baker *et al.* extend their analyses to investigate how usage choices [2] affect emotion transitions. This work found that confused learners are likely to game the system. Further, it was found that students who game the system are unlikely to transition into a confused state [3].

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 490–499, 2008.

In this paper we investigate the likelihood of affective transitions in a narrative-centered learning environment, CRYSTAL ISLAND. The CRYSTAL ISLAND environment utilizes narrative as a mechanism to contextualize learning, making the experience meaningful. Contextualized learning experiences are known to encourage regulated learning behavior [22] and influence student learning and motivation [18]. Because CRYSTAL ISLAND incorporates an engaging storyline into the learning experience, we supplement the known relevant emotions to learning used by D'Mello *et al.* [11] and Baker *et al.* [3] with affective states that may be relevant to the story (anger, anxiety, boredom, confusion, delight, excitement, fear, flow, frustration, and sadness). We extend our analysis of affective transitions to evaluate the impact of character empathetic responses (parallel vs. reactive empathy) to student affect and the relative impact on transitions.

The paper is organized as follows. Section 2 describes CRYSTAL ISLAND, the narrative-centered learning environment that has been developed in our lab for the domains of microbiology and genetics. Section 3 presents the experimental method utilized for collection of student affective experiences. In Section 4 we report findings on probable transitions in narrative-centered learning and present analyses of the impact of empathy on such transitions. Results are discussed in Section 5. Section 6 notes the limitations of the work, followed by conclusions and future work in Section 7.

2 Crystal Island

The CRYSTAL ISLAND environment is being created for the domains of microbiology and genetics for middle school students. It features a science mystery set on a recently discovered volcanic island where a research station has been established to study the unique flora and fauna. The user plays the protagonist, Alex, attempting to discover the genetic makeup of the chickens whose eggs are carrying an unidentified infectious disease at the research station. The story opens by introducing the student to the island and the members of the research team for which her father serves as the lead scientist. As members of the research team fall ill, it is her task to discover the cause and the specific source of the outbreak. She is free to explore the world and interact with other characters while forming questions, generating hypotheses, collecting data, and testing her hypotheses. Throughout the mystery, she can walk around the island and visit various locations. She can pick up and manipulate objects, and she can talk with characters to gather clues about the source of the disease. In the course of her adventure she must gather enough evidence to correctly choose which breeds of chickens need to be banned from the island.

The virtual world of CRYSTAL ISLAND, the semi-autonomous characters that inhabit it, and the user interface were implemented with Valve Software's Source™ engine, the 3D game platform for Half-Life 2. The Source engine also provides much of the low-level (reactive) character behavior control. The character behaviors and artifacts in the storyworld are the subject of continued work.

The following scenario illustrates a student's interactive narrative experience in CRYSTAL ISLAND. In the course of having members of her research team become ill, she has learned that an infectious disease is an illness that can be transmitted from one

Fig. 1. Overview of CRYSTAL ISLAND

Fig. 2. The user, Alex, with Jin, the camp nurse on CRYSTAL ISLAND

organism to another. As she concludes her introduction to infectious diseases, she learns from the camp nurse that the mystery illness seems to be coming from eggs laid by certain chickens and that the source of the disease must be identified. The student discovers through a series of tests that the bad eggs seem to be coming from chickens with white-feathers. The student then learns that this is a codominant trait and determines that any chicken containing the allele for white-feathers must be banned from the island immediately to halt the spread of the disease. The student reports her findings back to the camp nurse.

3 Methods

3.1 Participants

The subjects of the study consisted of 35 graduate students ranging in age from 21 to 60 (M = 24.4, SD = 6.41) including 9 females and 26 males. Among these students, 60% were Asian (n = 21), approximately 37% were Caucasian (n = 13) and one participant chose not to respond.

3.2 Procedure

Participants entered the experiment room where they completed informed consent documentation. They were randomly assigned to either the control condition or the empathy condition and were seated in front of a laptop computer. They were then given an overview of the experiment agenda, and they completed the pre-experiment questionnaires including the demographics survey, the interpersonal reactivity index survey [12], and the goal orientation survey [14].

Upon completing the pre-experiment questionnaires, participants were instructed to review CRYSTAL ISLAND instruction materials. These materials consisted of the backstory and task description, the character overviews, the map of the island, the control sheet, and definition sheet of the self-report emotions. Participants were then further briefed on the controls via a presentation summarizing the task and explaining each control in detail. Participants maintained access to the materials, including the definition sheet of the self-report emotions, throughout their interaction.

Participants were given 35 minutes to solve the mystery. Solving the mystery consisted of completing 15 goals including learning about various diseases, compiling the symptoms of the sickened researchers, testing a variety of possible sources, and reporting the solution (cause and source) back to the camp nurse.

Six CRYSTAL ISLAND characters (Audrey, Elise, Jin, Quentin, Robert, and Teresa), each play distinct roles in the CRYSTAL ISLAND environment. When subjects decided to interact with these particular characters, they were greeted with empathetic reactions to their expressed affective state, which they communicated through self-reports via an in-game dialog. The self-report dialog asked participants to select the affective state that best described their feelings at that time from a set of 10 affective states (anger, anxiety, boredom, confusion, delight, excitement, fear, flow, frustration, and sadness). This set of emotions was comprised of emotions identified with learning [9, 11, 16] together with basic emotions [13] that may play a role in students' experience of the CRYSTAL ISLAND narrative.

Immediately after solving the science mystery of CRYSTAL ISLAND (or after 35 minutes of elapsed interaction time for subjects who had not solved the mystery), subjects completed a post-experiment questionnaire. This researcher-designed questionnaire assessed perceptions of individual CRYSTAL ISLAND characters. The results of this instrument are outside the scope of this discussion.

4 Results

In this section we first present findings regarding common affective transitions observed in CRYSTAL ISLAND. These findings are followed by an analysis comparing and contrasting likely affective transitions stemming from parallel and reactive empathetic reactions by CRYSTAL ISLAND characters.

To compute transition likelihoods we adopt D'Mello *et al.*'s *L* [11], which is based on Cohen's Kappa [7], and has been used by Baker *et al.* for affective transition analysis in their simulation learning environment [3]. *L* computes the probability that a transition between two affective states (CURRENT → NEXT) will occur, where CURRENT refers to a reported emotion at time *t*, while NEXT refers to the next reported emotion at time *t*+1. D'Mello *et al.*'s *L* accounts for the base frequency of the NEXT affective state in assessing the likelihood of a particular transition. Formally,

$$L(\text{CURRENT} \rightarrow \text{NEXT}) = \frac{P(\text{NEXT} \mid \text{CURRENT}) - P(\text{NEXT})}{1 - P(\text{NEXT})}$$

L's numerator is divided by 1-P(NEXT) to normalize scores between -∞ and 1 [11]. A result of *L* equal to 1 translates to emotion NEXT always following the CURRENT emotion; an *L* value equal to 0 means the likelihood of transitioning to emotion NEXT is equal to chance, i.e., the probability of experiencing NEXT (the base rate) regardless of the CURRENT emotion. An *L* value less than 0 translates to the likelihood of transitioning to emotion NEXT being less than chance (the probability of experiencing NEXT regardless of the CURRENT emotion).

Table 1. Likelihoods for all transitions CURRENT → NEXT for the affective states: **Fr**ustration, **Fl**ow, **Co**nfusion, **De**light, **Bo**redom, **Anx**iety, **Ex**citement, **Ang**er, **Sa**dness, and Fear

	Next									
Current	***Fr***	***Fl***	***Co***	***De***	***Bo***	***Anx***	***Ex***	***Ang***	***Sa***	***Fe***
Fr	0.28	-0.19	0.10	-0.05	-0.07	-0.15	-0.10	-0.02	-0.01	0.09
Fl	-0.04	0.19	0.04	0.02	-0.01	0.03	-0.07	0.01	0.00	0.00
Co	0.04	0.04	0.16	-0.03	0.05	-0.04	0.10	-0.01	-0.01	-0.03
De	0.01	0.10	-0.13	0.21	-0.03	-0.05	-0.33	-0.02	0.00	0.00
Bo	0.13	-0.03	-0.03	-0.08	0.13	-0.04	-0.04	0.00	-0.03	0.04
Anx	-0.08	0.06	0.04	-0.07	-0.01	0.14	-0.19	0.09	0.00	0.00
Ex	-0.05	-0.11	0.06	-0.03	-0.03	0.03	0.24	-0.01	0.01	-0.02
Ang	0.00	-0.07	0.09	-0.39	0.00	0.23	0.01	0.00	0.00	0.00
Sa	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Fe	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

To characterize affective transitions we first compute L for each transition (CURRENT → NEXT), for each student. We then use mean L values across students to determine the likelihood of transitioning from each emotion CURRENT to each emotion NEXT. The results of ANOVAs determine whether the differences in likelihoods of transitioning to each NEXT emotion are significantly different for particular CURRENT emotions.

4.1 Affective Transitions

Aggregating self-reported affective states across the 35 participants we find flow to be the most frequently reported state (42%), followed by excitement (14%), confusion (13%), delight (11%), anxiety (8%), frustration (6%), boredom (3%), sadness (2%), anger (1%), and fear (1%).

ANOVAs indicated that six affective states had statistically significant differences among the likelihoods of transitions. Affective transitions were statistically significantly different transitioning from frustration ($F(9, 340) = 2.06$, $p = 0.03$), flow ($F(9, 340) = 18.3$, $p < 0.0001$), confusion ($F(9, 340) = 1.79$, $p = 0.06$), delight ($F(9, 340) = 5.22$, $p < 0.0001$), anxiety ($F(9, 340) = 2.98$, $p = 0.002$), and excitement ($F(9, 340) = 2.62$, $p = 0.006$).

Frustrated learners are most likely to remain frustrated (Mean $L = .28$) followed by transitions to confusion (.10) and fear (.09). The remaining transitions were below chance levels (i.e., flow (-.19, $t(34) = -4.24$, $p < 0.0001$) and excitement (-.10)).

Learners in the state of flow were most likely to remain in flow (.19) followed by confusion (.04, $t(34) = -3.09$, $p = 0.003$), anxiety (.03), and delight (.02). Both frustration (-.04, $t(34) = -7.91$, $p < 0.0001$) and excitement (-.07) were below chance levels. The remaining transitions did not occur or occurred at chance levels.

Confused students were likely to remain in a confused state (.16) followed by excitement (.10), boredom (.05), frustration (.04), and flow (.04). The likelihood of these and all remaining conditions are summarized in Table 1.

4.2 Affective Transitions by Empathy

Empathy is the expression of emotion based on another's situation and not merely one's own [12]. Its expression can demonstrate that the target's (the recipient of empathetic expression) feelings are understood or shared. In the case of *parallel empathy*, an individual exhibits an emotion similar to that of the target [12]. This is typically based on an understanding of the target's situation and shows the empathizer's ability to identify with the target. *Reactive empathy*, in contrast, focuses on the target's affective state, in addition to her situation [12]. Reactive empathizers will display emotions that are different from the target's, often in order to alter or enhance the target's own affective state. This type of empathy is focused on the target whereas parallel empathy is more self-oriented. As such, reactive empathy can be viewed as a higher level of empathetic behavior.

Recent research with the characters of CRYSTAL ISLAND has investigated the merit of providing characters with empathetic capabilities to effectively respond to unfolding student experiences [20]. In CRYSTAL ISLAND, empathetic responses are short, text-based responses consisting of 1 to 2 sentences. Parallel responses consist of the character expressing the same emotion as the user through text responses. On the other hand, reactive responses demonstrate advanced cognitive processing on the character's part by providing responses designed to be more motivating and thus revealing the character's desire for the user to be in a positive emotional state. The results below investigate the likelihood of affective transitions based on empathetic expressions by CRYSTAL ISLAND characters in response to student CURRENT emotions. The findings suggest that in certain situations, parallel and reactive empathy have significant differences in the affective transitions (NEXT emotion) that are likely to occur.

While the relatively low frequencies of some transitions prevent many of the visible differences from being statistically significant, interesting patterns do emerge. Figures 3 and 4 present the transitions from the state of flow and frustration by empathetic reaction type (parallel or reactive). Analyzing the transitions from the state of flow we find that parallel empathy (.11) is somewhat significantly more likely to support students' remaining in the state of flow than reactive empathy (-.05), $t(12) = -2.08$, $p = 0.06$. Similarly, we find that the likelihood of transitioning to frustration from a frustrated state is significantly more likely when characters empathetic reactions are parallel in nature (.57) than reactive (-.13), $t(12) = -2.09$, $p = 0.059$. Other patterns with visible differences emerging from this analysis of affective transitions are summarized in Table 2. Although the transition frequencies were not sufficiently high for the differences to be statistically significant, they merit discussion.

Table 2. Interesting likelihood for transitions differences by empathetic response type (parallel or reactive)

CURRENT	*Transition State (NEXT)*	*Parallel Empathy Likelihood*	*Reactive Empathy Likelihood*
Boredom	Boredom	.35	-.04
	Confusion	0	-.41
	Flow	-.13	.32
	Frustration	-.08	.26
Anxiety	Anxiety	.33	.05
	Frustration	-.20	.17
Frustration	Frustration	.57	-.13
	Flow	.10	-.25
	Confusion	-.17	.15
Flow	Flow	.11	-.05
	Confusion	.04	.08
Delight	Delight	.21	.21
	Flow	.07	.17

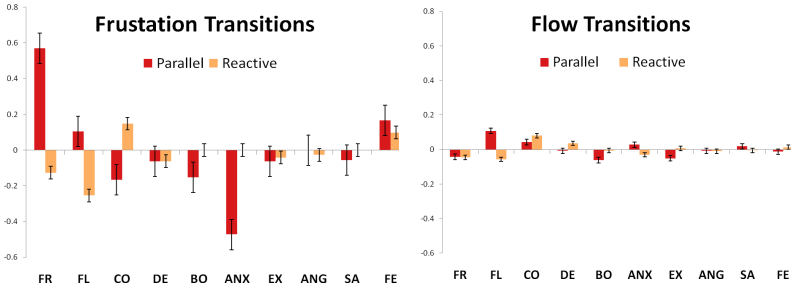


Fig. 3. Transitions from frustration and flow to **FR**ustration, **FL**ow, **CO**nfusion, **DE**light, **BO**redom, **AN**xiety, **EX**citement, **AN**ger, **SA**dness, and **FE**ar

5 Discussion

The analysis of affective state transitions in CRYSTAL ISLAND replicate findings by D'Mello *et al.* [11] and Baker *et al.* [3]. For instance, the state of flow dominated self-reported affect. The dominance of the flow state has been reported in a number of affective studies with intelligent learning environments [3, 9, 11]. Frustration and boredom were reported notably less frequently than in D'Mello *et al.*'s study and was comparably reported to frequencies found in Baker *et al.* Perhaps surprisingly, emotions found to be relevant to learning (boredom, confusion, delight, flow, and frustration) were more prevalent than the narrative affective states (anger, excitement, fear, and sadness) hypothesized to be relevant affective outcomes to experiencing the CRYSTAL ISLAND story.

Among the most likely transitions were transitions where NEXT = CURRENT. This was true for the affective states frustration, flow, confusion, delight, boredom, anxiety, excitement, and anger. This result also replicates the findings of [11] and [3]. D'Mello termed these cycles *vicious cycles* for negative affective states (similar to Burleson's notion of "state of stuck" [6]) and *virtuous cycles* when students are likely to stay in positive states (i.e., flow).

When we consider affective transitions where NEXT occurs at time *t+1* after an empathetic response from a CRYSTAL ISLAND character, we notice differences in the likely affective outcomes. For instance, if a student is in a frustrated state, parallel empathy is likely to elicit a transition in which the student stays frustrated. In contrast, reactive empathy is less likely than chance to prompt the same vicious cycle. Instead reactive empathy tends to promote transitions to a confused state, which is known to have better correlations with learning [9].

When we consider likely transitions from the state of flow, we find that parallel empathy is likely to encourage students to enter a virtuous cycle and remain in the state of flow. Reactive empathy is less likely than chance to produce the flow state and is likely to promote an affective state transition to confusion. Since a flow state is an optimal state of experience [10], we can understand why reactive empathy cannot motivate students to enter a more engaged state.

Analyzing transition patterns from the state of boredom, we find parallel empathy is likely to encourage a vicious cycle while reactive empathy is less likely than chance to produce the same cycle. Instead, reactive empathy is most likely to transition to flow, with frustration slightly less likely than flow. In the future, when we can accurately predict when reactive empathy is likely to encourage flow as opposed to when it is likely to promote frustration, this diagnostic information can inform pedagogical agents' empathetic responses to alleviate student boredom and promote a state of flow.

6 Limitations

The results of this study are affected by the virtual characters that interacted empathetically with participants. It is possible that the gender, narrative role, and pedagogical role of the characters may affect the likelihood of transitions in addition to the type of empathy. Another shortcoming is that affective states were solely collected from student self-reports. In contrast, both D'Mello et al [11] and Baker *et al.* [3] used judged reports of affect in their transition analysis. In the study reported here, video recordings of participants' faces were collected during their interactions with the learning environment to permit future work to consider judged reports of affect with this dataset. Finally, to determine how broadly the results hold, the transitions that were found to be likely with this subject population need to be validated with other populations, such as the intended population of middle school student users.

7 Conclusion

Given the central of role of affect and motivation in cognitive processes, it is becoming increasingly more important for intelligent tutoring systems to consider the affective experiences of students. This study replicates the findings of studies conducted with

AutoTutor [11] and The Incredible Machine simulation-based learning environment [3], including a demonstration of the prominence of the state of flow during learning. By extending our analysis to consider how affective transitions differ given empathetic character responses, the findings can inform the design of heuristics for pedagogical agents to determine when the use of empathy is likely to have desired outcomes and what type of empathy (parallel or reactive) would be best utilized. Such analysis can also inform the utility induced models of empathy [20].

The results suggest two directions for future work. First, they call for investigation of what type of feedback pedagogical agents should consider when empathy does not promote desirable affective states for learning. For instance, reactive empathy was likely to encourage transitions to either flow or frustration. In instances where empathy promoted frustration we should determine why empathy does not work and what type of system response would be more appropriate. Second, analysis of individual differences is necessary to determine the affective transitions common across a variety of demographics such as gender, but also across learning attributes such as efficacy, goal orientation, interest, and abilities to self-regulate both learning and affect.

Acknowledgments

The authors wish to thank the members of the IntelliMedia research lab for their assistance in implementing CRYSTAL ISLAND, Omer Sturlovich and Pavel Turzo for use of their 3D model libraries, and Valve Software for access to the Source™ engine and SDK. This research was supported by the National Science Foundation under Grant REC-0632450 and CNS-0540523. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. André, E., Mueller, M.: Learning affective behavior. In: Proceedings of the 10th Intl. Conf. on Human-Computer Interaction, pp. 512–516. Lawrence Erlbaum, Mahwah (2003)
2. Baker, R., Corbett, A., Koedinger, K., Wagner, A.: Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game the System". In: Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems, pp. 383–390 (2004)
3. Baker, R., Rodrigo, M., Xolocotzin, U.: The dynamics of affective transitions in simulation problem-solving environments. In: Proceedings the 2nd International Conference on Affective Computing and Intelligent Interactions, Lisbon, Portugal, pp. 666–677 (2007)
4. Baylor, A., Kim, Y.: Pedagogical agent design: The impact of agent realism, gender, ethnicity, and instructional role. In: Proceedings of the 7th International Conference on Intelligent Tutoring Systems, pp. 403–592. Springer, New York (2004)
5. Beal, C., Lee, H.: Creating a pedagogical model that uses student self reports of motivation and mood to adapt ITS instruction. In: Workshop on Motivation and Affect in Educational Software, in conjunction with the 12th Intl. Conf. on Artificial Intelligence in Education (2005)
6. Burleson, W.: Affective learning companions: Strategies for empathetic agents with real-time multimodal affective sensing to foster meta-cognitive and meta-affective approaches to learning, motivation, and perseverance. PhD thesis, Massachusetts Institute of Technology (2006)

7. Cohen, J.: A Coefficient of Agreement for Nominal Scales. Educational and Psychological Measurement 20, 37–46 (1960)
8. Conati, C., Mclaren, H.: Data-driven refinement of a probabilistic model of user affect. In: Proceedings of the 10th Intl. Conf. on User Modeling, pp. 40–49. Springer, New York (2005)
9. Craig, S.D., Graesser, A.C., Sullins, J., Gholson, B.: Affect and learning: an exploratory look into the role of affect in learning with AutoTutor. Journal of Educational Media 29(3), 241–250 (2004)
10. Csikszentmihalyi, M.: Flow: The Psychology of Optimal Experience. Harper and Row, New York (1990)
11. D'Mello, S., Taylor, R.S., Graesser, A.: Monitoring Affective Trajectories during Complex Learning. In: Proceedings of the 29th Annual Meeting of the Cognitive Science Society, Austin, TX, pp. 203–208 (2007)
12. Davis, M.: A Social Psychological Approach. Brown and Benchmark Publishers, Madison (1994)
13. Ekman, P., Friesen, W.: The facial action coding system: A technique for the measurement of facial movement. Consulting Psychologists Press, Palo Alto (1978)
14. Elliot, A., McGregor, H.: A 2 x 2 achievement goal framework. Journal of Personality and Social Psychology 80(3), 501–519 (2001)
15. Gratch, J., Marsella, S.: A domain-independent framework for modeling emotion. Journal of Cognitive Systems Research 5(4), 269–306 (2004)
16. Johnson, L., Rizzo, P.: Politeness in tutoring dialogs: run the factory, that's what I'd do. In: Proceedings of the 7th Intl Conf. on Intelligent Tutoring Systems, pp. 67–76. Springer, New York (2004)
17. Kort, B., Reilly, R., Picard, R.: An affective model of interplay between emotions and learning: Reengineering educational pedagogy—building a learning companion. In: Proceedings IEEE Intl. Conf. on Advanced Learning Technology: Issues, Achievements and Challenges, pp. 43–48. IEEE Computer Society, Madison (2001)
18. Linnenbrink, E., Pintrich, P.: Multiple goals, multiple contexts: The dynamic interplay between personal goals and contextual goal stresses. In: Volet, S., Jarvela, S. (eds.) Motivation in learning contexts: Theoretical advances and methodological implications, pp. 251–269. Elsevier, New York (2001)
19. McQuiggan, S., Lee, S., Lester, J.: Early prediction of student frustration. In: Proc. of the 2nd Intl. Conf. on Affective Computing and Intelligent Interaction, Portugal (2007)
20. McQuiggan, S., Robison, J., Phillips, R., Lester, J.: Modeling parallel and reactive empathy in virtual agents: An inductive approach. In: Proc. of the 7th Intl. Joint Conf. on Autonomous Agents and Multi-Agent Systems, Portugal (to appear)
21. Paiva, A., Dias, J., Sobral, D., Aylett, R., Woods, S., Hall, L., Zoll, C.: Learning by feeling: Evoking empathy with synthetic characters. Applied Artificial Intelligence 19, 235–266 (2005)
22. Perry, N.: Young children's self-regulated learning and the contexts that support it. Journal of Educational Psychology 90, 715–729 (1998)
23. Porayska-Pomsta, K., Pain, H.: Providing Cognitive and Affective Scaffolding through Teaching Strategies. In: Proceedings of the 7th International Conference on Intelligent Tutoring Systems, pp. 77–86. Springer, New York (2004)
24. Prendinger, H., Ishizuka, M.: The empathic companion: A character-based interface that addresses users' affective states. Applied Artificial Intelligence 19, 267–285 (2005)
25. de Vicente, A., Pain, H.: Informing the detection of the students' motivational state: an empirical study. In: Proceedings of the 6th International Conference on Intelligent Tutoring Systems, pp. 933–943. Springer, New York (2002)

Word Sense Disambiguation for Vocabulary Learning

Anagha Kulkarni, Michael Heilman, Maxine Eskenazi, and Jamie Callan

Language Technologies Institute
Carnegie Mellon University
5000 Forbes Ave,
Pittsburgh, PA, USA 15213
{anaghak,mheilman,max,callan}@cs.cmu.edu

Abstract. Words with multiple meanings are a phenomenon inherent to any natural language. In this work, we study the effects of such lexical ambiguities on second language vocabulary learning. We demonstrate that machine learning algorithms for word sense disambiguation can induce classifiers that exhibit high accuracy at the task of disambiguating homonyms (words with multiple distinct meanings). Results from a user study that compared two versions of a vocabulary tutoring system, one that applied word sense disambiguation to support learning and another that did not, support rejection of the null hypothesis that learning outcomes with and without word sense disambiguation are equivalent, with a *p*-value of 0.001. To our knowledge this is the first work that investigates the efficacy of word sense disambiguation for facilitating second language vocabulary learning.

Keywords: Vocabulary Learning, Word Sense Disambiguation, English as a Second Language, Computer Assisted Language Learning.

1 Introduction

Learning vocabulary is central to learning any language. Learning a new word implies associating the word with the various meanings it can convey. Consequently it is helpful to think of acquiring vocabulary as a task of learning word-meaning pairs, such as, {(word_1, meaning_1), (word_1, meaning_2), (word_2, meaning_1)} rather than a task of learning words {(word_1), (word_2)}. This approach is, of course, more relevant for some words than others. Many words can convey only a single meaning. However for many other words that is not the case and such words are termed as ambiguous words. It is important for an intelligent tutoring system designed to assist English as a Second Language (ESL) students to improve their English vocabulary, to operate at the level of the word-meaning pairs being learned and not just the words being learned, for several reasons. The most important reason is to be able to assess learning of the particular meanings of a word that the student was exposed to. The second reason is to personalize and adapt the tutoring material in order to expose the student to all or a particular set of meanings of a word. These observations motivate the study of word meaning/sense disambiguation (WSD) for supporting vocabulary learning in a tutoring system.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 500–509, 2008.

2 Background

For this study, we extended an existing tutoring system for ESL vocabulary, which is described below in Section 2.1. Sections 2.2 and 2.3 provide a background about the phenomenon of word sense ambiguity and its aspects relevant to this study.

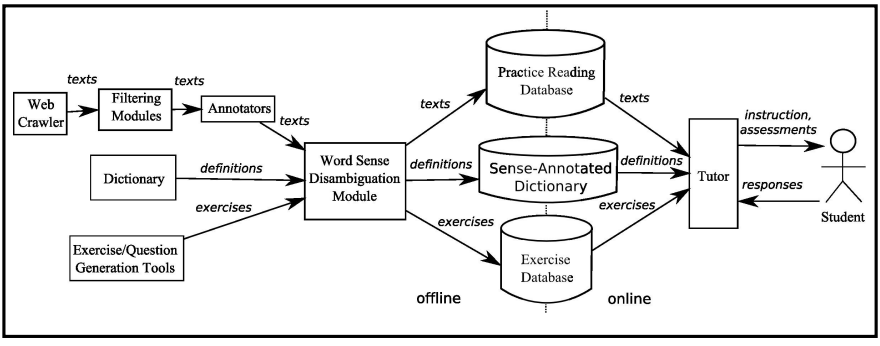


Fig. 1. REAP System Diagram

2.1 REAP Tutoring System

THE REAP tutoring system [1] assists ESL students in improving their vocabulary. Its approach is that of context-based teaching, where the word, its meaning and its usage are demonstrated by exposing the student to the word in a natural text (the practice text) instead of in isolation. For each grade level a human teacher prescribes a set of new words (focus words) that are appropriate for that level. The REAP system personalizes instruction for each student by allowing the student to indicate through self-assessments the words that he or she is familiar with, which modifies word priorities. In each instructional cycle, the student chooses from four different practice texts, each containing two or more words from the student's personalized word list. Once the student has finished reading the text, he or she answers practice questions that are based on the words that he or she was exposed to in the text.

The practice texts containing focus words are selected from the World Wide Web (WWW). As such, REAP brings together the two sub-disciplines of language technologies, Information Retrieval (IR) and Computer Assisted Language Learning (CALL). The quality of the practice text is controlled by selecting only those documents from the WWW that pass through various automatic filters implemented in the REAP system, for instance, text-quality filter, reading-level filter and document length filter. REAP's system diagram is shown in Figure 1. To maximize the time on task, documents containing multiple focus-words are preferred. Students can also indicate preferences for 16 general topics including Arts, Science, and Sports, which are taken into consideration by the system while choosing the practice texts for each student. A machine-readable version of an ESL dictionary, the Cambridge Advanced Learners Dictionary (CALD) [3] is integrated into REAP. Students can use CALD while they read a document, to lookup definitions and examples of a word.

2.2 Lexical Ambiguity in Language

Most languages have words with multiple meanings. Zipf [4] proposed several empirical laws based on the *Principle of Least Effort* to explain some of the phenomena observed in languages. One law proposes that word sense ambiguity arises as a result of two competing forces, both of which try to follow the principle of least effort. On one side, the effort of the speaker of the language will be minimized by using a small vocabulary, that is, by using few words that are capable of conveying many meanings. On the other side, the listener's effort will be minimized by using as many distinct words, in terms of meaning, as possible. Zipf provides empirical evidence for his theory. However, the conclusion most pertinent to this study is that his theory formalizes the common belief that human language vocabularies are a mix of ambiguous and unambiguous words.

2.3 Polysemes and Homonyms

Ambiguous words can be categorized as *polysemes* or *homonyms*. Polysemes are words that can convey multiple related meanings, whereas, homonyms are words that can convey multiple distinct meanings. For example, the word *branch* as a noun has the following definitions listed in CALD that are all related.

1. one of the parts of a tree that grows out from the main trunk and has leaves, flowers or fruit on it
2. a part of something larger
3. one of the offices or groups that form part of a large business organization
4. a part of a river or road that leaves the main part

whereas the following definitions for the word *bark* in the noun form convey clearly distinct meanings.

1. the loud, rough noise that a dog and some other animals make
2. the hard outer covering of a tree

In this study we concentrate on homonyms for two reasons. First, distinguishing between related senses of a word is a highly subjective task. Several studies [5, 6, 7] have shown that the agreement between human annotators is very low on this task. Second, we believe that ESL students can transfer their knowledge about one sense of a word to another related sense of the word without much difficulty, especially in a context-based learning setup. However, we hypothesize that learners are not able to do so for homonyms, and thus assistance should improve learning.

Given this background the two objectives of this work can be stated as:

1. to demonstrate that automatic disambiguation of homonyms that occur in web-pages is possible and,
2. to show that such methods can be applied in a tutoring environment to produce a positive effect on ESL vocabulary learning.

3 Word Sense Disambiguation Methodology

WSD is a well-established research area in the field of natural language processing [8]. Here we concentrate on word sense disambiguation of homonyms that occur in web-pages. For the purposes of this paper we will refer to the task of homonym disambiguation as WSD. This task can be structured as a supervised learning problem where the availability of an annotated training-set is assumed or as an unsupervised learning problem where a training-set is not available. In the supervised learning setting WSD becomes a text classification problem and in the unsupervised setting it is often referred as the word sense discrimination task [15, 16]. The unsupervised framework has the advantage of not requiring a training-set. We experimented with both approaches. However, models learned using supervised methods were consistently more accurate than models learned using unsupervised techniques, thus we focus on the supervised methods in this paper. The decision to use supervised learning techniques was motivated by the need to minimize the potential effects of classification errors on student learning.

The supervised WSD task can be stated formally as follows. The following are given: 1) a homonym h, 2) the set M of distinct meanings that h can convey, and 3) a training-set T that consists of (i, k) pairs where i is a portion of text containing h, and k is the meaning that i conveys. The goal is to learn the best classification model $\hat{C}$ using T for h. A best classification model $\hat{C}$ would be one that generalizes well, that is, it not only performs classification with high accuracy on T but also on the test-set S, which consists of instances of h that were not used for inducing the model. Note that the phrase 'the portion of text' used above is a generic phrase that can refer to a phrase, a sentence, a paragraph, or an entire document containing h. The task of learning a text classification model is typically divided into two broad phases – feature extraction and classification algorithm. Section 3.1 describes the features that were used in this study. We used the Weka [10] implementation of two standard machine learning algorithms, namely, Multinomial Naïve Bayes (MNB) [11] and Support Vector Machines (SVM) [12]. Section 3.1 briefly describes these algorithms.

3.1 Features Types and Classification Algorithms

Loosely speaking, classification models generalize by learning patterns of co-occurrences from the training data. For instance, a classification model for the homonym *bark* might learn from the training examples that if the word *dog* occurs in the vicinity of the word *bark* then it is most likely that the first meaning of *bark* related to dogs, is at work in this instance. In general, identifying multiple such indicators or features from the training data can lead to learning a robust classification model, assuming that the task under consideration lends itself to such abstractions.

We use two types of features, namely, unigrams (UNI) which are lexical features, and part-of-speech-bigrams (POS-BI) which are lexico-syntactic features. [13] shows that using such a combination of features is effective for supervised WSD. Unigrams are the unique words that occur in the training-set instances. However, only those unigrams that occurred within some window $(-w, +w)$ of the homonym are considered. The intuition behind this approach is that a word's local context is more likely to encode information about the word and its sense than distant context. The window size was one of the

parameters that we varied in our experiments (Section 4). Closed-class words such as articles, prepositions and numerals were not included because they rarely help distinguish between the different meanings of a homonym Unigrams that occur only once were also discarded. Generating the part-of-speech-bigrams was a two step process. In the first step, each word which was within a window of five words on the left and right of the homonym in each training-set instance was annotated with its part-of-speech tag, such as, noun, verb, adjective, using the Stanford Part-Of-Speech tagger[1]. Given this sequence of part-of-speech tags, unique pairs of consecutive part-of-speech tags are extracted in the second step. POS-BIs capture the syntactic information about the neighborhood of the homonym, and provide another level of abstraction. Generating other features such as lexical bigrams, trigrams or pos-trigrams, is possible. However, the available evidence for these features becomes very sparse. Intuitively, the number of occurrences in the training-set of the trigram "loud dog bark" would be much less than that of "loud", "dog", and "bark" individually.

The Multinomial Naïve Bayes algorithm [11, 14] is a variation of the classification algorithm based on the Bayes' Rule. MNB makes two assumptions: i) the features are independent of each other given the class (the sense of the word in our case), and ii) the class conditional distribution of the features is a multinomial distribution. Support Vector Machines [12] identify a classification boundary that is maximally separated from all the classes (again, the senses of the homonym, in our case). MNB and SVM are well-known, frequently used, and frequently effective; however, they make very different assumptions about the data. As we will see, the assumptions made by MNB are empirically found to be more appropriate for our data.

4 Experiments with WSD Approaches

We focus on 30 homonyms in this study. The different morphological forms of these homonyms were treated as the same word type. For instance, along with the root form of the homonym *issue*, the derived form *issues,* was also analyzed. The list of 30 words in their root form is given in Table 1. Following is the description of the training-set generation process.

The definitions of a word provided by an ESL dictionary, the Cambridge Advanced Learners Dictionary (CALD), were manually grouped based on the relatedness of the meaning that they convey. An example, for the homonym *issue,* is shown below:

Group 1

1. a subject or problem which people are thinking and talking about

Group 2

2. a set of newspapers or magazines published at the same time or a single copy of a newspaper or magazine

Group 3

3. An issue of shares is when a company gives people the chance to buy part of it or gives extra shares to people who already own some.
4. to produce or provide something official

[1] http://nlp.stanford.edu/software/tagger.shtml

The third column in Table 1 specifies the number of definition groups (senses) for each of the 30 words that were used in this study. These definition groups were used as the sense-inventory for the coding task, which is described next. The training-set was created for each word by annotating occurrences of the word in web-pages, using the word's sense-inventory. This annotation task was performed by an independent group of coders from the Qualitative Data Analysis Program[2] at the University of Pittsburgh. In the initial round, four coders annotated 100 documents. (We use the words 'document' and 'web-page' interchangeably.) The pair of coders with best inter-coder agreement (Cohen's *kappa* = 0.88) was chosen to annotate the remaining documents. To expedite the annotation task, both coders annotated different sets of documents. A "spot-check" process was implemented by periodically providing a subset of the documents to both the coders for annotation. The inter-coder agreement for the entire training-set, in terms of Cohen's *kappa*, based on spot-checks, was 0.92. These high kappa values provide empirical evidence that the task of distinguishing between distinct senses is much less subjective than the task of distinguishing between fine-grained senses, and thus can be automated much more effectively. The annotated dataset thus generated was used in the experiments described below.

For each word, classification models were learned using the two machine learning algorithms described in Section 3.2. Further more, 46 different window sizes (10 through 100, in steps of 5), for the unigram feature extraction task were experimented with. 10-fold cross-validation [9] was used to compare the different classification models learned. The best learning algorithm for each word and the best window size is specified in the second last and the last columns of the Table 1.[3] Multinomial Naïve Bayes algorithm was the best classification algorithm for 22 of the 30 homonyms. The average best window size was 25 (-25, +25). The best accuracy values for each word are compared with the baseline accuracy given in the column 4. The baseline accuracy is computed by assigning labels to any given instance of the word with the *most frequent sense* of the word in the training-set. As the table shows, for some words the baseline accuracy is quite high (e.g., factor, function) indicating that one sense is extremely dominant. This can happen when all or a majority of the instances of the word in the training-set belong to the same topic, such as, science, or arts. Cohen's *kappa*, reported in column 6, indicates the agreement between the gold standard and the predicted senses. Table 1 is sorted on the *kappa* values.

5 User Study and Discussion

A user study was conducted at the English Language Institute (ELI)[4], at the University of Pittsburgh, to test the effects of WSD on ESL vocabulary learning. A total of 56 students from the ELI Reading 4 and Reading 5 classes (respectively upper intermediate and advanced ESL students) participated in the study. The Reading 4 group consisted of 39 students, and the Reading 5 group consisted of 18 students. The pre-test consisted of 30 self-assessment questions similar to the Yes/No test [17], one for

[2] http://www.qdap.pitt.edu/

[3] Note that in this setting it is practical to use different learning algorithms and window sizes for each word, if that yields the best accuracy.

[4] http://www.eli.pitt.edu/

each homonym, where each student was asked to indicate if he or she knew the word. The study was conducted for duration of eight consecutive weeks; one session per reading level per week. The pre-test was conducted, during the first session, and lasted approximately 10 minutes. The practice reading activity started during the same session and continued for the following seven consecutive weeks, one session per week. The post-test was conducted during the eighth and final session. It consisted of cloze questions for each of the <word, sense> pairs that the student was exposed to during the practice reading sessions. These cloze questions were manually created by the teachers at the ELI. Out of the 56 students 47 students attended the final session, the post-test. The following analysis is based only on these 47 students. The experimental group that used the WSD-equipped REAP consisted of 24 students and the control group consisted of 23 students. General ESL proficiency was measured by Michigan Test of English Language Proficiency (MTELP) scores.

Table 1. Summary of Supervised classification models for 30 homonyms

	Homonym	Number of senses	Baseline Accuracy (%)	Best Accuracy (%)	Cohen's Kappa	Classification Algorithm	Window Size (*w*)
1	panel	2	84.92	99.82	0.993	MNB	25
2	transmission	2	51.91	99.15	0.983	MNB	70
3	procedure	2	82.04	99.40	0.980	MNB	85
4	foundation	3	79.17	98.81	0.965	MNB	85
5	principal	2	64.39	98.05	0.957	SVM	40
6	bond	2	81.78	98.67	0.956	SVM	70
7	aid	2	59.76	97.71	0.952	SVM	85
8	tape	2	75.16	98.14	0.951	MNB	40
9	monitor	2	84.36	98.58	0.947	MNB	85
10	code	2	66.18	97.10	0.936	MNB	85
11	volume	3	51.00	96.00	0.934	MNB	85
12	suspend	2	81.48	97.53	0.919	MNB	40
13	contract	3	83.67	97.73	0.919	MNB	40
14	qualify	3	79.81	97.12	0.909	MNB	70
15	major	3	90.24	98.32	0.904	MNB	40
16	conceive	2	80.92	96.95	0.898	SVM	70
17	pose	3	58.26	94.78	0.893	MNB	25
18	trigger	2	59.40	94.33	0.883	SVM	25
19	brief	3	75.81	95.70	0.883	SVM	10
20	parallel	2	53.70	94.14	0.882	MNB	85
21	supplement	2	73.18	95.45	0.882	MNB	70
22	channel	2	53.25	93.49	0.869	MNB	10
23	depress	2	60.66	93.44	0.862	MNB	40
24	manual	2	68.80	93.59	0.850	SVM	10
25	factor	2	91.24	97.72	0.848	MNB	85
26	shift	3	70.71	92.55	0.837	MNB	70
27	function	2	90.84	97.01	0.830	MNB	55
28	issue	3	80.90	92.96	0.767	MNB	85
29	complex	3	58.51	86.86	0.735	MNB	70
30	appreciate	2	68.63	86.27	0.690	SVM	40

The word sense disambiguation models learned for the 30 homonyms, that are described in Section 4, were integrated into the REAP system to support vocabulary learning. This modified the system in two main ways. First, during the reading session whenever a student looked up one of the 30 homonyms in the integrated dictionary, the order of the definitions was adjusted to show the most appropriate definition for that particular document at the top of the list. Prior research such as [4] motivates this by observing that dictionary users often stop reading the entry for a word after reading only the top few definitions, irrespective of whether those provide them the correct definition for the given usage of the word.

The second change improves the quality of the multiple-choice practice questions that follow each reading. Each question requires the student to select one among five alternative definitions of a word that he or she was exposed to in the reading. Generating these five alternative definitions (four distractors and one correct definition) is straightforward for words with single sense. However, for homonyms, without WSD it is not possible to ascertain that the definition of the homonym that conveys the meaning used in a particular document just read is included in the set of five alternatives. For example, a student might read a document that contained '…the weekly issue of Time magazine…' and thus the definition #2 for the *issue,* given in section 4, should be included as one of the five alternatives for the practice question. We refer to such correctly matched definitions as the 'true' definition for that (document, word) pair, in the following discussion. The version of REAP without WSD orders a word's dictionary definitions by their frequency of usage in CALD, and generates the five alternatives for a given word by choosing four distractor definitions for words of the same part of speech and by choosing the definition with highest usage frequency that has not yet been used in any of the previous questions for that student. This methodology has a better chance of including the 'true' definition in the five alternatives than a methodology based on random selection of definition. Nevertheless, it does not always guarantee inclusion of 'true' definition. These post-reading definition questions provide additional practice for the <word, sense> pair that the student was exposed to during the practice reading and thus reinforce the instruction. We claim that providing this additional exposure, where the student is actively involved in the process, promotes robust learning. Mismatched practice questions can potentially confuse the student and thus adversely affect student learning. Thus, studying the effects of this WSD-induced matching as measured by post-test performance is the most revealing comparative analysis.

Table 2 shows the data for this analysis. To make a fair comparison, we analyze only those <word, sense> pairs from the experimental group that would have been mismatched in the practice questions, even with the usage frequency methodology, had it not been for WSD. Thus, 45 <word, sense> pairs were found to have been well-matched in practice questions and texts because of WSD. The performance of the experimental group on these 45 <word, sense> pairs on the post-test is given in the Table 2. The columns split the data according to the information provided by the student during self-assessment. The rows group the data based on the post-test results. For the control group, only those <word, sense> pairs that did not get matched by chance for the practice questions are analyzed.

We use Generalized Estimating Equations (GEE) [18] to test our hypotheses. GEE takes into account the correlations between data-points that are not independently and

identically distributed. The use of GEE was warranted in this study because different data-points--corresponding to post-test results for particular words--were in some cases generated by the same student. Based on the data in Table 2 we perform the following hypothesis test using GEE.

H0: The true distribution of the post-test performance of the experimental group for the chosen words and the true distribution of the post-test performance of the control group is the same.

H1: The true distributions of post-test performance of the experimental group and the control group are not the same.

Analysis with GEE produces a *p*-value 0.001, which strongly supports rejecting the null hypothesis. The mean and the standard deviation for the experimental and control groups are ($M = 0.8$, $SD = 0.404$) and ($M = 0.5$, $SD = 0.505$), respectively. We also performed another analysis for testing the above hypotheses, however, this time two additional explanatory (independent) variables were also included, pre-test information and the MTELP score of the student. This analysis produced a *p*-value of 0.003 for the coefficient learned for the WSD status, which was the only significant coefficient in the model that was fit. Thus we can conclude that the treatment given to the experimental group had a statistically reliable and positive effect on their performance in the task of vocabulary learning.

Table 2. Data from the experimental and control groups of the user study

	Experimental Group				**Control Group**		
	Known	**Unknown**			**Known**	**Unknown**	
Correct	28	8	36	**Correct**	16	6	22
Incorrect	7	2	9	**Incorrect**	16	6	22
	35	10	45		32	12	44

It is important to note that the pre-test required the students to indicate the words that they were familiar with, but did not ask about their familiarity with the <word, sense> pairs. As a result, although it appears from the data in Table 2 that most of the students were familiar with majority of the words included in this study, it is quite likely that a student who indicated being familiar with a word could only be familiar with only one of the meanings of the word. In fact, the second analysis above showed that the pre-test information could not explain students' performance on the post-test.

6 Conclusion

This work establishes that performing sense disambiguation for homonyms helps vocabulary learning in ESL students. It is demonstrated that the task of disambiguating homonyms can be automated by learning classifiers that can assign the appropriate sense to a homonym in a given context with high accuracy. A user study reveals that students equipped with WSD-enabled vocabulary tutor perform significantly better than students using vocabulary tutor without the WSD capabilities.

Acknowledgments

We thank Alan Juffs, Lois Wilson, Greg Mizera, Stacy Ranson and Chris Ortiz at the English Language Institute at the University of Pittsburgh for using REAP in the classroom. We also thank Dr. Howard Seltman and Dr. Ted Pedersen for their guidance and help. This work has been supported by the Barbara Lazarus Women@IT Fellowship, (in part) by the Pittsburgh Science of Learning Center which is funded by the National Science Foundation, award number SBE-0354420 and Dept. of Education grant R305G03123. Any opinions, findings, conclusions or recommendations expressed in this material are the authors', and do not necessarily reflect those of the sponsors.

References

1. Heilman, M., Collins-Thompson, K., Callan, J., Eskenazi, M.: Classroom success of an Intelligent Tutoring System for lexical practice and reading comprehension. In: Proceedings of the Ninth International Conference on Spoken Language Processing (2006)
2. Coxhead, A.: A New Academic Word List. TESOL. Quarterly 34(2), 213–238 (2000)
3. Walter, E. (ed.): Cambridge Advanced Learner's Dictionary, 2nd edn. Cambridge University Press (2005)
4. Manning, C., Schütze, H.: Foundations of Statistical Natural Language Processing, pp. 27–28. The MIT Press (2003)
5. Jorgenson, J.: The psychological reality of word senses. Journal of Psycholinguistic Research 19, 167–190 (1990)
6. Dolan, W.B.: Word Sense Ambiguation: Clustering Related Senses. In: Proc. 15th Int'l. Conf. Computational Linguistics, ACL, Morristown, pp. 712–716 (1994)
7. Palmer, M., Dang, H.T., Fellbaum, C.: Making fine-grained and coarse-grained sense distinctions. Journal of Natural Language Engineering (2005)
8. Ide, N., Jean, V.: Introduction to the special issue on word sense disambiguation: the state of the art. Computational Linguistics 24(1), 1–40 (1998)
9. Mitchell, T.: Machine Learning, pp. 111–112. The McGraw-Hill Companies, Inc. (1997)
10. Witten, I.H., Eibe, F.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
11. McCallum, A., Nigam, K.: A Comparison of Event Models for Naive Bayes Text Classification. In: AAAI 1998 Workshop on Learning for Text Categorization (1998)
12. Vapnik, V.N.: The nature of statistical learning theory. Springer, New York (1995)
13. Mohammad, S., Pedersen, T.: Combining Lexical and Syntactic Features for Supervised Word Sense Disambiguation. In: Proceedings of the Conference on Computational Natural Language Learning, Boston (2004)
14. Larkey, L.S., Croft, W.B.: Combining classifiers in text categorization. In: Proceedings of the Nineteenth Annual International ACM SIGIR Conference, pp. 289–297 (1996)
15. Schütze, H.: Automatic word sense discrimination. Computational Linguistics 24(1), 97–123 (1998)
16. Pedersen, T., Bruce, R.: Distinguishing word senses in untagged text. In: Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (1997)
17. Meara, P., Buxton, B.: An alternative to multiple choice vocabulary tests. Language Testing 4, 142–145 (1987)
18. Hardin, J., Hilbe, J.: Generalized Estimating Equations. Chapman and Hall/CRC (2003)

Student Note-Taking in Narrative-Centered Learning Environments: Individual Differences and Learning Effects

Scott W. McQuiggan, Julius Goth, Eunyoung Ha, Jonathan P. Rowe, and James C. Lester

Department of Computer Science, North Carolina State University, Raleigh NC 27695
{swmcquig,jgoth,eha,jprowe,lester}@ncsu.edu

Abstract. Note-taking has a long history in educational settings. Previous research has shown that note-taking leads to improved learning and performance on assessment. It was therefore hypothesized that note-taking could play an important role in narrative-centered learning. To investigate this question, a note-taking facility was introduced into a narrative-centered learning environment. Students were able to use the facility to take and review notes while solving a science mystery. In this paper we explore the individual differences of note-takers and the notes they take. Finally, we use machine learning techniques to model the content of student notes to support future pedagogical adaptation in narrative-centered learning environments.

1 Introduction

Narrative is central to human cognition. Because of the motivational and contextual properties of narrative, it has long been believed that story-based learning can be both engaging and effective. Research has begun to develop narrative-centered learning environments that combine story contexts and pedagogical support strategies to deliver compelling learning experiences. Contextualizing learning within narrative affords the use of artificial intelligence techniques that tailor narrative and educational content to students' actions, affective states, and abilities. Drawing on an interdisciplinary body of work including intelligent tutoring systems, embodied conversational agents, and serious games, these environments offer the promise of adaptive, motivating learning experiences. Narrative-centered learning environments are currently under investigation in a range of domains, including military soft-skills training [9, 20], anti-bullying education [2], health intervention education [11], and science learning in microbiology and genetics [13].

Note-taking has a long history in educational settings. It has repeatedly been found that student note-taking leads to 1) improved learning, regardless of whether the students had the opportunity to review their notes prior to evaluation [19], and 2) increases in test performance [4,10]. Note-taking is believed to simulate the generative process in which students encode connections between prior knowledge and learning content [15]. Students' must self-regulate note-taking during learning episodes to manage strategy use, prior knowledge, and attentional capacity [19, 22]. Thus, self-regulatory processes can be observed via student note-taking.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 510–519, 2008.

In this paper we investigate the merit of modeling student note-taking behavior. The approach adopted here draws on recent work in which linguistic features extracted from student writing have played important roles in modeling student learning in an intelligent tutoring system [8] and in analyzing student interactions in on-line discussions [12, 16]. ITSPOKE [8] predicts student learning using five sets of linguistic features automatically extracted from the essays written by students. These features include surface, semantic, pragmatic, discourse structure, and local dialogue context features, with the semantic features serving as the strongest predictor. ARGUNAUT [12] assists human tutors mediating student on-line discussions by analyzing student contributions in a discussion and recognizing important student actions. Student action classifiers are trained from features including manual analysis of individual and connected contributions of students, where preliminary results suggest the importance of the *critical-reasoning* feature. Another approach employs *speech acts* to investigate student interactions in on-line discussions [16]. Two speech act classifiers, a *question* classifier and an *answer* classifier, were constructed from *n*-gram features automatically computed from student postings.

Motivated by note-taking findings in the learning sciences literature and research in human language technologies, we analyze notes taken by middle school students in an experiment with the CRYSTAL ISLAND learning environment. The remainder of the paper is organized as follows: Section 2 describes CRYSTAL ISLAND, the narrative-centered learning environment that has been developed in our lab and its note-taking functionalities. Sections 3, 4, and 5 report on an empirical study using CRYSTAL ISLAND that examines individual differences in note-taking and preliminary models of note taking. Design implications and limitations are discussed in Sections 6 and 7, respectively. Conclusions and directions for future work follow in Section 8.

2 Crystal Island and Note-Taking

CRYSTAL ISLAND is a narrative-centered learning environment that features a science mystery set on a recently discovered volcanic island. The curriculum underlying CRYSTAL ISLAND's science mystery is derived from the state standard course of study for eighth-grade microbiology. Students play the role of the protagonist, Alyx, who is attempting to discover the identity and source of an unidentified infectious disease plaguing a newly established research station. The story opens by introducing the student to the island and members of the research team for which the protagonist's father serves as the lead scientist. Several of the team's members have fallen gravely ill, including Alyx's father. Tensions have run high on the island, and one of the team members suddenly accuses another of having poisoned the other researchers. It is the student's task to discover the outbreak's cause and source, and either acquit or incriminate the accused team member.

CRYSTAL ISLAND's expansive setting includes a beach area with docks, a large outdoor field laboratory, underground caves, and a research camp with an infirmary, lab, dining hall, and living quarters for each member of the team. Throughout the mystery, the student is free to explore the world and interact with other characters while forming questions, generating hypotheses, collecting data, and testing hypotheses. Students can pick up and manipulate objects, take notes, view posters, operate

lab equipment, and talk with non-player characters to gather clues about the source of the disease. During the course of solving the mystery, students are minimally guided through a five-problem curriculum. The first two problems deal pathogens, including viruses, bacteria, fungi, and parasites. Students gather information by interacting with in-game pathogen "experts" and viewing books and posters in the environment. In the third problem, students are asked to compare and contrast their knowledge of four types of pathogens. The fourth problem guides the student through an inquiry-based hypothesis-test-and-retest problem. In this problem students must complete a "fact sheet" with information pertaining to the disease inflicting members of the CRYSTAL ISLAND research team. Once the "fact sheet" is completed and verified by the camp nurse, the student completes the final problem regarding the treatment of viruses, bacteria, fungi, and parasites, and selects the appropriate treatment plan for sickened CRYSTAL ISLAND researchers. The story and curriculum are interwoven throughout the student experience.

The virtual world of CRYSTAL ISLAND, the semi-autonomous characters inhabiting it, and the user interface were implemented with Valve Software's Source™ engine, the 3D game platform for Half-Life 2. The Source engine also provides much of the low-level (reactive) character behavior control. The character behaviors and artifacts in the storyworld are the subject of continued work. Note-taking functionalities were recently added to the CRYSTAL ISLAND environment. Students access their notes using the 'N' key, which launches an in-game dialog where students can review their notes and supply additional comments if they so choose. Below we report findings on the use of note-taking in a CRYSTAL ISLAND study and investigate the potential benefits of modeling student notes.

3 Experiment Method

3.1 Participants

There were 54 female and 62 male participants varying in age and race. Approximately 2% of the participants were American Indian or Alaska Native, 5% were Asian, 29% were Black or African American, 58% were Caucasian, 6% were Hispanic or Latino, and 6% were of other races. Participants were all eighth grade students ranging in age from 12 to 15 (M = 13.27, SD = 0.51). The students had recently completed the microbiology curriculum mandated by the North Carolina state standard course of study before receiving the instruments, tests, and interventions of this experiment.

3.2 Materials and Apparatus

The pre-experiment paper-and-pencil materials for each participant were completed one week prior to intervention. These materials consisted of a researcher-generated CRYSTAL ISLAND curriculum test, demographic survey, achievement goals questionnaire [5], Self-Efficacy for Self-Regulated Learning (SESRL) [3], Science Self-Efficacy, modified from [14], and immersion tendencies questionnaire [20]. The CRYSTAL ISLAND curriculum test consists of 23 questions created by an interdisciplinary team of researchers and was approved for language and content by the students' eighth-grade science teachers. Elliot

and McGregor's achievement goals questionnaire is a validated instrument that measures four achievement goal constructs (mastery-approach, performance-approach, mastery-avoidance, and performance-avoidance goals) [6]. Bandura's Self-Efficacy for Self-Regulated Learning scale [3] consists of 11 items rated by participants on a 7-point Likert scale. Witmer and Singer developed and validated the Immersive Tendencies Questionnaire (ITQ) to measure individual predispositions towards presence experiences [20]. The ITQ consists of three subscales: activity involvement tendency, activity focus tendency, and video game playing tendency. Participants indicate their level of tendency for each item on a 7-point Likert scale. Witmer and Singer found individual tendencies, as recorded by the ITQ, to be predictive of presence [20].

Post-experiment materials were completed immediately following intervention. These materials consisted of the same CRYSTAL ISLAND curriculum test, achievement goals questionnaire [6], interest [18], science self-efficacy, and the presence questionnaire [20]. The interest scale was adapted from those used by Schraw to capture differences across groups and to examine within-subject relationships with learning outcomes [18]. Participants' presence experience was captured by the Presence Questionnaire (PQ) developed and validated by Witmer and Singer [20]. The PQ contains several subscales including involvement/control, naturalism of experience and quality of the interface scales. The PQ accounts for four categories of contributing factors of presence: control, sensory, distraction, and realism.

4 Design and Procedure

4.1 Design

The experiment randomly assigned students to a CRYSTAL ISLAND narrative condition or a minimal-narrative condition. The focus of the study was to investigate the role of note-taking in narrative-centered learning. Students received an intervention consisting of the CRYSTAL ISLAND microbiology curriculum through one of two deliveries. The CRYSTAL ISLAND narrative condition supplemented the curriculum with the full CRYSTAL ISLAND narrative, including the poisoning scenario, character backstories, and character personality. The CRYSTAL ISLAND minimal-narrative condition supplemented the curriculum with the minimal story required to support the curriculum. In this condition, the story consisted of research members falling ill and the request for the student to uncover the mysterious illness. The minimal-narrative condition did not include the poisoning storyline, character back stories or explicit character personality. Students were able to take notes in both conditions.

4.2 Participant Procedure

Participants entered the experiment room having completed the pre-test and instrumentation one prior to the intervention. Participants were first instructed to review CRYSTAL ISLAND instruction materials. These materials consisted of the CRYSTAL ISLAND backstory and task description, a character handout, a map of the island, and a control sheet. Participants were then further directed on the controls via a presentation explaining each control in detail. This included a brief remark that the 'N' key could be used to take notes. Five minutes were allotted for this instruction.

Participants in the CRYSTAL ISLAND conditions (narrative and minimal-narrative) were given 50 minutes to work on solving the mystery. Solving the mystery consisted of completing a number of goals including learning about pathogens, viruses, bacteria, fungi, and parasites, compiling the symptoms of the sickened researchers, recording features of hypothesized diseases causing the CRYSTAL ISLAND illness, testing a variety of possible sources, and reporting the solution (cause and source) back to the camp nurse to design a treatment plan.

Immediately after solving the science mystery of CRYSTAL ISLAND, or 50 minutes of interaction, participants completed the post-experiment questionnaires. First to be completed was the CRYSTAL ISLAND curriculum test, followed by the remaining post-experiment questionnaires. Completion of post-experiment materials took no longer than 35 minutes for participants. In total, experiment sessions lasted 90 minutes.

5 Results

5.1 Annotating the Notes Corpus

To analyze the type of notes taken by students, the corpus was annotated using a coding scheme consisting of six categories of tags that characterize the contents of student notes (Table 1). The four main categories were *Narrative (N), Curricular (C), Hypothesis (H),* and *Procedural (P)*. Notes containing facts from the narrative storyline, such as summaries of the unfolding plot, observations of particular objects located in specific area of the environment, and symptoms of ill-stricken characters were tagged with *N*. Similarly, notes pertaining to facts from the curriculum, such as definitions or characteristics of viruses and bacteria, were tagged with *C*. Student notes that explicitly expressed possible solutions regarding the source or cause of the outbreak or solution to the scientific mystery were tagged as *H*. A hypothesis could be either narrative (e.g., suspecting a character of poisoning others) or curricular (e.g., guessing the cause of the disease wreaking havoc on the island). Notes that were directed at maintaining a set of tasks to be completed were tagged as *P*. Two additional categories include Garbage (*G*) and Other (*O*). *G* is used to mark notes that do not contain any meaningful information while *O* covers the remaining notes that contain meaningful information but do not belong to any one of the categories in the current coding scheme.

Using the annotation scheme described above, four judges each tagged the corpus. Inter-rater reliability was determined using both Fleiss' kappa [7], which allows for more than two raters, and Cohen's kappa [5], for each pair of raters. There was full agreement between all four judges according to Fleiss's kappa (K = .70). There were six possible pairings of judges. The paired-judge kappas ranged from moderate agreement to full agreement: judge2-judge4 (.59), judge1-judge4 (.62), judge3-judge4 (.66), judge2-judge3 (.73), judge1-judge3 (.77), and judge1-judge2 (.82). From the tagged corpus, a voting scheme was used to select the tag with the highest frequency for each note. In instances of ties (n = 17) the scheme defaulted to Judge 1.

For purposes of analysis, we calculate the frequency of notes taken in each category for each student. These frequencies are the basis for the results reported below.

Table 1. Tagging scheme for note-taking content

Category(tag)	Description	Student Example	Freq.
Narrative (*N*)	Notes contain facts from the unfolding storyline	*Teresa - fever, pain, vomiting*	79
Curricular (*C*)	Notes contain facts from the learning content	*fungi is easily spread on its own from person to person*	156
Hypothesis (*H*)	Notes contain a possible solution to the source and / or cause of the mysterious illness	*I think that she might have something to do with Adola.*	16
Procedural (*P*)	Notes pertain to tracking tasks/goals to be completed	*Something is wrong with dad and i have to find what*	22
Garbage (*G*)	Notes do not contain any meaningful information	*BLAH BLAH BLAH*	4
Other (*O*)	Notes contain meaningful information, but do not belong to any other category	*Scientific method: 1. find method question/problem*	17

5.2 The Effects and Individual Differences of Note-Taking

Fifth-three percent of the students (n = 62, Total n = 116) took notes in CRYSTAL ISLAND. In general, post-test performance and learning gains were unaffected by student note-taking (i.e., no statistical significance found). However, students who took notes on hypotheses (n = 10) surrounding the solution to the mystery did perform significantly better on the curriculum post-test, $t(114) = 2.18$, $p = 0.03$. Hypothesis note-takers also performed significantly better on the curriculum pre-test, $t(114) = 2.16$, $p = 0.03$. This suggests that high achieving students are more likely to take hypothesis notes.

In reviewing the differences between the narrative and minimal-narrative conditions we find that students in the minimal-narrative condition took significantly more curriculum notes, $t(114) = 2.59$, $p = 0.01$. Meanwhile students in the narrative condition took significantly more procedural notes, $t(114) = -2.40$, $p = 0.01$. Perhaps surprisingly, there was no statistically significant difference the notes taken regarding the unfolding narrative between conditions.

Gender played a significant role in note-taking. Overall, females took significantly more notes than males, $t(114) = 2.19$, $p = 0.03$. Females also took significantly more curriculum notes ($t(114) = 2.08$, $p = 0.04$) and narrative notes ($t(114) = 1.83$, $p = 0.06$, marginal significance). While there were few garbage notes (n = 4, all composed by male students), a two-tailed t-test reveals marginal significance, $t(114) = -1.64$, $p = 0.10$. There were no significant differences between gender for hypothesis and procedural notes.

High mastery-approach students (students with goals of understanding content for the sake of its own value), as measured by Elliot and McGregor's Achievement Goals Questionnaire [6], took significantly more notes than low mastery-approach students, $t(114) = 2.06$, $p = 0.04$. This includes high mastery-approach students taking significantly more narrative notes ($t(114) = 2.44$, $p = 0.01$) and procedural notes ($t(114) = 2.01$, $p = 0.05$)

than low mastery approach students. There were no significant differences for curriculum notes or hypothesis notes between low and high mastery-approach students. There were also no significant differences when considering other goal orientations (performance-approach, performance-avoidance, and mastery-avoidance).

Finally, there were significant correlations between note-taking and student efficacy for self-regulated learning (SRL) [3]. A positive correlation was found between hypothesis note-taking and self-efficacy for SRL, $r(114) = 0.185$, $p = 0.04$. There was also a significant positive correlation between narrative note-taking and self-efficacy for SRL, $r(114) = 0.29$, $p = 0.002$.

5.3 Modeling the Content of Student Note-Taking

We consider several machine learning techniques, namely, support vector machines (SVMs), naïve Bayes, nearest neighbor, and decision trees to induce models that predict note-taking categories characterizing the content of the notes. All models are induced using the Weka machine learning toolkit [21] using a tenfold cross validation scheme to produce the training and testing datasets. Tenfold cross-validation is widely used for obtaining the acceptable estimate of error [21].

We utilize Weka's StringToWordVector filter to transpose a string value to a feature vector consisting of unigram tokens. Each token represents an individual feature within the feature vector. The filter supports several additional parameters, such as binary versus word count, TF and IDF weighting, and several optional stemming algorithms. For our analysis, the unigram tokenizer was chosen over the *n*-gram tokenizer, in part because of the filter's inability to eliminate stopwords prior to tokenization. For instance, phrases such as *towards the door* would not eliminate the stopword *the* prior to tokenization. Instead of creating a single bigram *towards door*, the filter would create two bigrams *towards the* and *the door*. The default stoplist, as well as a stemming algorithm, were chosen to reduce the dimensions of the feature space and improve classifier performance [1].

For modeling purposes, notes were stripped of punctuation, contractions were tokenized (i.e., *hasn't* → *has not*), and typos and misspellings were corrected. Additionally, numbers and measurements, such as *15* and *nm/nanometer* were each aggregated into a special token [16].

The best performing induced model (SVM) correctly classified 86.4% of instances. The SVM model was followed by naïve Bayes (83.2%), nearest neighbor (81.0%), and decision tree (80.6%). The highest true positive rates (SVM model) were achieved with the *curriculum* and *narrative* class. These two classes also comprised 86% of all instances. While hypothesis and procedural classes performed worse on recall (37.5% and 50% respectively), it is worth noting that precision values were reasonable for both classes (54.5% and 73.3%). The kappa between the judge-annotated corpus and the SVM classification was .77. Using the tag occurring most frequently, *curriculum*, as a baseline measure (57.1%) we find the frequency with which the SVM model correctly classified instances significantly outperformed this baseline model, $\chi^2(5, N = 273) = 49.84$, $p < 0.0001$.

6 Discussion

Note-taking offers a view into the problem-solving processes undertaken by students. The study reveals that students who took hypothesis notes performed better on post-tests, confirming inquiry-based learning findings that it is critical that learning environments scaffold students' hypothesis generation activities. The individual differences suggest which students are likely to take notes, a finding that can inform the design of tutorial interventions that encourage note-taking for students who otherwise would be unlikely to take notes. The results identify several correlations between note-taking and self-efficacy for self-regulated learning (SRL), which can provide insight into student strategy use.

The results also suggest that we can accurately model the content of student note-taking. Because the note-taking classifiers can operate with a high level of accuracy, we may be able to incorporate them into learning environments to monitor strategy use. The key diagnostic information they can provide offers the opportunity for learning environments to scaffold note-taking strategies that are likely to lead to desirable learning outcomes.

7 Limitations

The experiment was designed to control for time on task, allowing 50 minutes for the intervention. As a result of this constraint, only 49 students of the 116 CRYSTAL ISLAND participants finished or were working on the final problem at the end of the 50 minute session. An alternative design might consider controlling for task completion. The time constraint may have had an adverse effect on students' strategies to make use of note-taking, resulting in fewer students' taking notes or a diminished quantity of their notes.

8 Conclusion

Given the role of note-taking in self-regulatory learning processes, the study has design implications for intelligent tutoring systems in general and narrative-centered learning environments in particular. The results indicate that students taking hypothesis notes regarding likely solutions to a narrative-centered science problem show better performance on post-test measures. Furthermore, the ability to induce models that accurately predict the content of student notes provides a view into student self-regulatory processes, which can be used by tutoring systems to monitor some forms of student strategy use.

The results suggest several directions for future work. First, it will be interesting to extend this analysis across a larger corpus of notes. The study reported here highlights the merits of note-taking within narrative-centered learning environments; future studies should consider conditions designed to motivate note-taking and specific note-taking formats (e.g., [10]). Another interesting direction for future work is considering alternative approaches to motivating students to take notes given individual differences. In particular, it is important to explore motivational techniques in the context of note-taking that target students with approach goals in contrast to those with avoidance goals. It is important to identify which motivational techniques for

note-taking are most effective for students with a mastery orientation in contrast to those with a performance orientation.

Acknowledgments. The authors thank members of the IntelliMedia lab for their assistance, Omer Sturlovich and Pavel Turzo for use of their 3D model libraries, and Valve Software for access to the Source™ engine and SDK. This research was supported by the National Science Foundation under Grant REC-0632450. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Androutsopoulos, I., Koutsias, J., Chandrinos, K.V., Spyropoulos, C.D.: An experimental comparison of naïve Bayesian and keyword-based anti-spam filtering with personal e-mail messages. In: Proceedings of SIGIR 2000, 23rd ACM International Conference on Research and Development in Information Retrieval, pp. 160–167. ACM Press, Athens (2000)
2. Aylett, R., Louchart, S., Dias, J., Paiva, A., Vala, M.: FearNot! - An Experiment in Emergent Narrative. In: Panayiotopoulos, T., Gratch, J., Aylett, R.S., Ballin, D., Olivier, P., Rist, T. (eds.) IVA 2005. LNCS (LNAI), vol. 3661, pp. 305–316. Springer, Heidelberg (2005)
3. Bandura, A.: Guide for constructing self-efficacy scales. In: Pajares, F., Urdan, T. (eds.) Self-Efficacy Beliefs of Adolescents, pp. 307–337. Information Age Publishing, Greenwich (2006)
4. Bauer, A., Koedinger, K.: Evaluating the effect of technology on note-taking and learning. In: CHI Extended Abstracts on Human Factors in Computing Systems, Montreal, Canada, pp. 520–525 (2006)
5. Cohen, J.: A coefficient of agreement for nominal scales. Educational and Psychological Measurement 20, 37–46 (1960)
6. Elliot, A., McGregor, H.: A 2 x 2 achievement goal framework. Journal of Personality and Social Psychology 80(3), 501–519 (2001)
7. Fleiss, J.L.: Measuring nominal scale agreement among many raters. Psychological Bulletin 76(5), 378–382 (1971)
8. Forbes-Riley, K., Litman, D., Purandare, A., Rotaru, M., Tetreault, J.: Comparing Linguistic Features for Modeling Learning in Computer Tutoring. In: Proceedings of 13th International Conference on Artificial Intelligence in Education (AIED), Los Angeles (2007)
9. Johnson, W.L.: Serious Use of a Serious Game for Language Learning. In: Proceedings of AIED 2007, IOS Press, Marina del Rey (2007)
10. Kiewra, K., Benton, S.L., Kim, S., Risch, N.: Effects of note-taking format and study technique on recall and relational performance. Contemporary Educational Psychology 20, 172–187 (1995)
11. Marsella, S., Johnson, W.L., LaBore, C.: Interactive Pedagogical Drama for Health Interventions. In: Proceedings of the 11th International Conference on Artificial Intelligence in Education, Australia (2003)
12. McLaren, B.M., Scheuer, O., De Laat, M., Hever, R., De Groot, R., Rose, C.P.: Using Machine Learning Techniques to Analyze and Support Mediation of Student E-Discussions. In: Proceedings of 13th International Conference on Artificial Intelligence in Education (AIED), Los Angeles (2007)

13. Mott, B., Lester, J.: Narrative-centered tutorial planning for inquiry-based learning environments. In: Proceedings of the 8th International Conference on Intelligent Tutoring Systems, Jhongli, Taiwan, pp. 675–684 (2006)
14. Nietfeld, J.L., Cao, L., Osborne, J.W.: The effect of distributed monitoring exercises and feedback on performance and monitoring accuracy. Metacognition and Learning 1(2), 159–179 (2006)
15. Peper, R.J., Mayer, R.E.: Generative effects of note-taking during science lectures. Journal of Educational Psychology 78, 34–38 (1986)
16. Ravi, S., Kim, J.: Profiling Student Interactions in Threaded Discussions with Speech Act Classifiers. In: Proceedings of 13th International Conference on Artificial Intelligence in Education (AIED), Los Angeles (2007)
17. Riedl, M., Stern, A.: Believable Agents and Intelligent Story Adaptation for Interactive Storytelling. In: Proceedings of the 3rd International Conference on Technologies for Interactive Digital Storytelling and Entertainment, Darmstadt, Germany (2006)
18. Schraw, G.: Situational interest in literary text. Contemporary Educational Psychology 22, 436–456 (1997)
19. Van Meter, P., Yokoi, L., Pressley, M.: College students' theory of note-taking derived from their perceptions of note-taking. Journal of Educational Psychology 86(3), 323–338 (1994)
20. Witmer, B., Singer, M.: Measuring presence in virtual environments: A presence questionnaire. Presence: Teleoperators and Virtual Environments 7(3), 225–240 (1998)
21. Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufman, San Francisco (2005)
22. Zimmerman, B.J., Schunk, D.H.: Self-regulated Learning and Academic Achievement. Springer, New York (1989)

Assessing Aptitude for Learning with a Serious Game for Foreign Language and Culture

W. Lewis Johnson and Shumin Wu

Alelo Inc., 11965 Venice Blvd., Suite 402, Los Angeles, CA 90066 USA
ljohnson@alelo.com, shuminwu@usc.edu

Abstract. The Tactical Language and Culture Training System is interactive environment for learning foreign language and culture, designed to help people quickly acquire spoken communication skills. It is a serious game, combining interactive game experiences as well as interactive lessons. As part of our research, we wish to understand what individual learner characteristics predict successful learning with this approach, and investigate whether the approach can be improved so that a wider range of learners can learn effectively with it. This paper reports on an experiment, to assess which learners learn most effectively with TLCTS, and attempt to identify the individual factors that predict successful training with TLCTS. A group of US Marines participated in a session of focused training with Tactical Iraqi™, an Iraqi Arabic course designed for military use. Performance scores and interaction logs were analyzed to determine which learners were most successful, and why.

Keywords: Student Assessment, Serious Game, Language Learning.

1 Introduction

The Tactical Language and Culture Training System is a serious game platform that helps learners quickly acquire knowledge of foreign language and culture through a combination of interactive lessons that focus on particular skills, and interactive games to practice and apply these skills. The system makes extensive use of intelligent tutoring and other artificial intelligence technologies, including automated speech recognition, spoken dialog and animated agents, natural language process and learner modeling. TLCTS is very widely used. At least twenty thousand copies of TLCTS courses have been distributed, and tens of thousands of learners have used them to date. The most widely used course is the Tactical Iraqi™, which teaches colloquial Iraqi Arabic.

A study by the Marine Corps Center for Lessons Learned (MCCLL) currently is documenting strong evidence of Tactical Iraqi™'s effectiveness. It examines the experience of the 2nd Battalion and 3rd Battalion, 7th US Marine Regiment (2/7 and 3/7 Marines), who trained with Tactical Iraqi™ prior to their most recent tour of duty in Iraq. The 3/7 attracted the attention of MCCLL because it did not suffer a single combat casualty during its most recent tour of duty. In the opinion of the 3/7 officers, the training greatly increased the battalion's operational capability as it enabled it to

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 520–529, 2008.

operate more efficiently, with an increased understanding of the situation and better relationships with the local people. They felt that the Marines who trained with Tactical Iraqi™ achieved a substantial level of language proficiency, so much so that they deserved to receive college credit for the language proficiency they gained. These results, while preliminary, suggest that Tactical Iraqi™ training led to improved on-the-job performance (a Kirkpatrick level 3 result) [12] and this in turn contributed to improved organizational outcomes (a Kirkpatrick level 4 result). These results follow earlier experimental studies that provide scientific evidence that Tactical Iraqi™ produces learning gains [14].

However, there remain a number of questions that previous research does not answer. Can one predict what types of learners will benefit the most from training with TLCTS? What patterns of learner behavior and performance are predictive of success with TLCTS?

This paper presents preliminary results from a field study attempting to address these questions. A group of Marines took part in a focused training session with Tactical Iraqi™, attempting to identify which individuals show the most promise of learning effectively with the software. This work is an example of what Chan has referred to as adoption-based research [2]: research that contributes to, and is predicated upon, the successful adoption of effective learning systems. Studies such as these, conducted with learners in authentic educational settings, are necessary to understand how educational software performs in practice, and is a necessary step toward transition of learning technology into regular field use.

2 System Overview

The following is brief overview of some of the main capabilities that TLCTS training systems provide. More detail may be found elsewhere [9, 10, 11].

Figure 1 shows images of TLCTS trainers in use. Current systems run on Windows PCs equipped with a headset microphone. Each course includes a Skill Builder, consisting of a set of interactive lessons, each of which focuses on communicative tasks. The top left of Figure 1 shows a Tactical Language learning lab installed for the U.S. Army 3rd Infantry Division at Ft. Stewart, GA. The top right of Figure 1 shows a typical Skill Builder lesson page. The learner can hear recordings of example phrases, and practice saying those phrases. The integrated speech recognizer, trained on language learner speech, gives the learner feedback as to whether or not their speech was intelligible and matched the target phrase. Learners practice in a series of exercises that progressively prepare learners for employing their language and cultural knowledge in conversational settings.

Two kinds of interactive games are included in TLCTS training systems. The bottom right of Figure 1 shows the Arcade Game in Tactical Pashto™, in which learners navigate their characters through a town by giving spoken commands in Pashto. The bottom left shows the Mission Game in which learners communicate with non-player characters using speech and gesture in order to carry out a mission. In this scenario, from Tactical Iraqi™, the player is instructing Iraqi non-player characters in the proper procedure for manning a security checkpoint.

Fig. 1. Images from the Tactical Language and Culture Training System (TLCTS)

TLCTS users receive tutorial feedback in the Skill Builder on their use of language. Depending on the type of exercise, the system can give feedback on pronunciation, morphological and grammatical forms, word choice, or cultural pragmatics, as in this example. In the games, on the other hand, the use of tutorial feedback is limited, as it was found to interfere- with game play. Instead, in the Mission Game feedback is integrated into the responses of the non-player characters in the game. The game display signals whenever the character's attitude changes, and the changes in attitude can influence the way the character responds to the learner.

3 Research Opportunity and Research Questions

The US Marine Corps Training and Education Command (TECOM) is currently conducting a multi-year study, called SEPTR (Simulation Enhanced Pre-deployment Training and Rehearsal) to evaluate the use of simulation-based training in preparing units for deployment overseas. The goals of the study are to test and validate existing training simulations, identify opportunities for improvement of those systems, and identify requirements for future training systems. The study is also expected to develop model methods for integrating simulation-based training effectively into training programs. These methods may then be disseminated across the Marine Corps, leading to the adoption of computer simulations as a standard method for training.

Because of TLCTS early success, TECOM selected it for inclusion in the SEPTR evaluations. The 2/7 Marines agreed to participate in SEPTR, and agreed to include

TLCTS into its current predeployment training program. However like any unit preparing for deployment overseas, the 2/7 has many skills to train, and very little time to complete the training. The battalion therefore decided to organize a "masters program", in which two Marines per squad would receive intensive training in language and culture. The challenge then was to quickly identify the Marines that were likely to benefit the most from TLCTS training, and enroll them in an intensive program of 40 or more hours of training with Tactical Iraqi™.

The standard method for assessing language aptitude is to employ a language aptitude test, e.g., the Defense Language Aptitude Battery (DLAB) used to determine who may pursue training as a military linguist. Such language aptitude tests are only moderate predictors of learning outcomes, typically yielding correlations of between 0.4 and 0.6 with learning outcomes as determined by a variety of outcome measures [4]. This is partly because other factors such as motivation influence language learning outcomes [5], but it also may be because current aptitude batteries may not test the full range of abilities relevant to language learning. In fact, the DLAB does not engage subjects in speaking the language, or using it for face-to-face communication in culturally appropriate ways. In contrast, TLCTS places particular emphasis on face-to-face communication in simulated social encounters. It is therefore not clear how strong a predictor the DLAB would be for the skills that TLCTS trains.

Therefore, instead of DLAB we decided to use a sample of material from Tactical Iraqi™ itself to assess likelihood for success in the masters training program. All candidate Marines would complete several hours of beginner-level training with TLCTS. The curriculum selected for this assessment would introduce the candidates to aspects of the phonology, morphology, syntax, and pragmatics of Iraqi Arabic, as well as non-verbal gestures and other cultural information relevant to face-to-face encounters with people in Iraq. We would then collect and analyze the data from the training sessions, including quiz scores, estimates of learner skill mastery, interaction logs, and speech recordings. We would also collect background information on each candidate, as well as self-assessments of their interest and motivation to learn Arabic. These data would allow us to answer the following questions:

1. Which candidates were most successful in their training?
2. Which characteristics of individual learners were conducive to success?
3. What patterns of training behavior led to success?

The choice of a version of the training system itself an assessment tool is unusual, but affords a number of advantages. It tests a wide range of cognitive abilities relevant learning language, wider than what is typical of language aptitude tests. It gives us an opportunity to determine whether trainees are able to assess their own language performance, plan their learning activities to achieve mastery, and recognize when they have successfully mastered the target language skills. Meanwhile, by taking part in the assessment the candidates are learning language and cultural skills that are potentially valuable to the trainees. This enhances motivation, both at the individual level (individual candidates are more likely to have intrinsic motivation to learn the language skills and do well) and at the organizational level (the officers in the battalion are more willing to set aside training time for the candidates to participate, and are more likely to have interest in successful training outcomes).

The data collected from this assessment, as well as from the training sessions of the candidates who are ultimately selected to complete the masters training program, gave us the opportunity to further investigate some research questions that are of concern to our research. One is the following:

4. Are game-based learning techniques useful in promoting learning?

Although games have attracted significant interest in educational circles, evidence of their effectiveness is mixed. This has led some educational researchers to question their value. Distractive elements [3] and learning-unrelated reward systems [13] are blamed for lowering productivity of learning activities. Undesired behaviors were reported where learners tried to use "shortcuts" to succeed in games by exploring the system properties instead of the learning materials [1]. Other researchers are optimistic about learning by playing games, but suggest games should be paired with traditional classroom curriculums and practices [6]. Previous studies and experience with TLCTS courses has also produced mixed results. Reports from TLCTS users indicate that they consider the game and simulation elements of TLCTS courses to be important and without them TLCTS would not have been chosen to be part of the SEPTR study. However an evaluation of an earlier version of Tactical Iraqi™ indicated that trainees actually rated the Skill Builder more highly than the game components [14]. We hypothesized that the subjects in the earlier study did not receive a proper orientation briefing regarding proper use of the learning software, and that the content focus of the game experiences needed to be adapted to make it more relevant to their jobs and missions. We wished to see whether better initial orientation, and recent improvements to the Mission Game, would result in improved attitudes toward the game experiences. We also wished to collect data on learner interaction with the games, to see whether there are opportunities to further improve the structure of the game experiences, and/or incorporate automated guidance and feedback, to help learners make most productive use of the games.

4 Study Procedure

The 2/7 officers selected 49 Marines to take part in the initial assessment, and organized them into two groups of approximately 25. The session proctor gave an initial twenty-minute orientation, demonstrated the software, and explained how to use it for training. The proctor told the candidates to strive to master the material, reviewing and repeating the learning materials, exercises, and quizzes as necessary.

Candidates then spent ten minutes completing a short questionnaire. The questionnaire asked whether the candidates had been deployed to Iraq before, if so how many times, and how motivated they were to learn Arabic. These questions were asked because motivation has previously been found to affect language learning outcomes [2], and in the case of Tactical Iraqi™ previous evaluations showed that trainees who had previously been deployed to Iraq had higher motivation to learn Arabic [8]. Candidates were asked to report their background information that reveal their maturity, experience, and/or job responsibilities, and training experience, which we hypothesized might influence how the candidates learn. The candidates then trained for approximately 45 minutes in the Tactical Iraqi™ Skill Builder. They were directed to

focus on four lessons: *Getting Started* (a tutorial), *Meeting Strangers* (vocabulary, phrases and etiquette relating to meeting strangers, possessive morphological endings), *Introducing Your Team* (Arabic terms for military ranks, phrases and etiquette relating to making introductions, definite articles, demonstratives, grammatical gender and agreement), and *Pronunciation Lesson 1* (easy Arabic consonants, long vs. short vowels, single vs. double consonants). The proctor provided the candidates with occasional technical assistance, but otherwise left them to train on their own. After a 10 minute break, the candidates were then directed to resume training in the Skill Builder for another 45 minutes. They were then directed to spend twenty minutes in the Mission Game, and then take another ten-minute break. Finally, the candidates completed another 30 minutes of Skill Builder training.

5 Study Results

Of the 49 participating Marines, one was excluded from the analysis presented here because he did not complete the survey questionnaire. Each subject was assigned a score between 1 (low) and 5 (high) for his performance in each of the three learning environments: Skill Builder, Arcade Game, and Mission Game. The Skill Builder scores were assigned according to the number lessons attempted, the number of lessons completed with a high quiz score (80% or better), and the number of individual language and cultural skills that the learner model indicated were fully mastered. The Arcade Game scores were assigned according to the number of levels played, completed, and the number of hints requested by the learner to complete the level. Similarly, the Mission Game scores were assigned according to the number of scenes played, the number of scenes completed, and the number of hints the learner used to complete the scene. Overall performance scores were computed based on the environment performance scores and time spent within each learning environment, using the following formula:

$$OverallPerformanceScore = \sum_{env} (T_{env} \times Score_{env}) / \sum_{env} T_{env} \ , \qquad (1)$$

where *env* represents the three learning environments, T_{env} is the time spent in a particular environment, and $Score_{env}$ is the assigned score for this environment. Note that the overall performance scores are continuous values computed out of ordinal environment performance scores.

5.1 General Results: Which Candidates Were Most Successful?

Certain observations were recorded during the proctoring sessions. First, although the candidates were instructed to focus on the Skill Builder lessons, some trainees still remained in the two game environments that interested them until the proctor specifically directed them back to the lessons. Secondly, some trainees left early for various reasons. Thirdly, some trainees who had used TLCTS before tended to skip Skill Builder lessons and devoted more of their training time to the game environments.

Therefore, actual training time (as determined from the log data) had a relatively high variance (time in Skill Builder: M = 1.08 hrs, SD = 0.72 hrs; time in Mission Game: M = 0.92 hrs, SD = 0.55 hrs; time in Arcade Game: M = 0.36 hrs, SD = 0.36 hrs). And this in turn resulted in high variance in performance scores in each environment (Skill Builder score: M = 2.92, SD = 1.46; Mission Game score: M = 2.92, SD = 1.49; Arcade Game score: M = 2.48, SD = 1.38).

Thus we needed to compute a summary score that can fairly reflect the trainees' overall performance. We hypothesized that the result the trainee could achieve in a particular learning environment was proportional to the time he has spent in this environment. Therefore, the aforementioned method (1) was introduced to compute the overall performance score and is expected to counteract the noise that perturbs the accuracy of otherwise simply computed average score that would be used as the overall performance score. We argue this method is valid because language and culture skills taught/practiced in these environments are closely related. For example, we regard those who invested most of their training time in one environment and accomplished great results as good learners even though they might have scored low in other environments due to time constraints. On the other hand, if a trainee evenly distributed his time but only does averagely in each environment, we view this trainee as a mediocre performer.

As a result, the average overall performance score for this population (N=48) is close to the medium category (M=2.91, SD=1.13, %95CI = [2.585, 3.241]). We found 10 most successful candidates who achieved high performance scores (>4.0). 1 out of 10 scored 5 in all the three environments; 3 out of 10 scored 5 in two environments, and the rest 6 scored 5 in one environment. The best candidates spent on average 2.5 hours pure training time with the system (SD = 0.43 hrs).

5.2 Which Individual Characteristics Were Conducive to Success?

The 11 characteristics we examined are categorized into 4 groups. The personal trait category includes *age*, *education*, self-reported *motivation to learn Arabic language and culture,* and *experience of training with TLCTS before*; the military experience category includes *rank*, *time in service*, *experience of deployment to Iraq*; the linguistic category includes *language spoken other than English* and *language formally studied*; the music ability category includes self-rated *musical talent*, *ability to sing or play instrument*, and *experience of formal music training*.

T-tests show that 32 trainees who identified their motivation greater or equal to 4 outperformed the 14 trainees having motivation below 4 ($t(44) = 2.012$, $p = 0.50$). Older trainees (>=20 year old) scored lower than younger ones (<20), but the difference is not statistically significant ($t(46) = -1.491$, $p = 0.14$). No significant difference was found for education, either. The 21 trainees who received some college education had performance close to the 27 trainees who only received high school degrees ($t(45.75) = -0.383$, $p = 0.715$). Interestingly, former TLCTS trainers did not have superior performance than fresher users do. Rather, they scored a little lower than those who have never trained with TLCTS before ($t(46) = -0.123$, $p = 0.902$) as they would be expected to. The proctor observed that some former trainees devoted little effort to the Skill Builder lessons and played a lot in the game environment, but they were not able to complete the entire game, probably because their language skills had

decayed. Additionally, it also could be that some of the former trainees did not learn much in the previous experience, or only spent a little time on the system. Finally, among the former trainees there was a cluster of trainees who had both very low motivation and performance.

In the military experience category, rank did not effect the training results, as the average scores for three groups of different ranks are approximately the same (Rank > E-3 Score: M = 2.88, SD=1.46; Rank = E-3 Score: M = 2.91, SD = 1.17; Rank = E-2 Score: M = 2.95, SD = 1.04). However, the group with less than one year of *time in service* and the group with more then one year had statistically different performance ($t(45) = 1.961$, $p = 0.056$). As for experience of deployment to Iraq, there is no significant finding between the group with the experience and the group without ($t(44) = -.822$, $p = 0.416$).

Those who had studied another foreign language performed at a level that was close to those who did not ($t(46) = 0.115$, $p = 0.909$).In the language experience category, only 4 trainees speak a language other than English, so it is impossible to draw conclusions about the role of foreign language fluency.

In the music ability category, no significant effect is found. Trainees who rated their music talents higher seemed to score slightly lower than those who identified themselves as "I have no talent in music" ($t(46) = -0.551$, $p = 0.584$). Similarly, trainees who reported practicing singing or playing instrument were outperformed by their non-practicing counterparts ($t(45) = -1.091$, $p = 0.281$). However, those having taken formal music training scored a little higher ($t(45) = 0.430$, $p = 0.669$). But those results are not statistically significant to verify hypotheses.

In summary, characteristics such as *motivation* and *time in service* seem promising to be conductive to success. We do not find significant effect with other characteristics. The findings are reinforced when we take a look at the group of those successful candidates. We found out among the 10 best trainees, 90% reported high motivation, and 70% served in military more than 1 year. T-tests on the best candidate group and the other trainee group also show that motivation has significant effects on the overall performance ($t(44) = 2.381$, $p = 0.021$), while the effect of time in service seems not statistically significant ($t(9.07) = 1.036$, $p = 0.372$).

5.3 What Patterns of Training Behavior Led to Success?

We examined the activity patterns of the successful candidates against the rest of participants. It was found that successful learners did particularly well in Skill Builder lessons, compared with the rest of the trainees (quizzes completed: $t(9.65) = 2.654$, $p = 0.025$; skill mastered: $t(46) = 2.691$, $p = 0.100$). We believe that this provided them with good foundations to be able to apply the language and culture skills they learned from the lessons to the other game environments. In the Arcade Game, 60% of them never requested a single hint to complete a level, and therefore were never penalized by minus points because of hint requests.

Log files show that they also performed in-game learning. For instance, 60% of them used this strategy: when playing the mission scenes, they first heavily used the hint facility to go through them, and then replayed the scenes and finally completed them The best performer group requested 59.10 mission game hints on average, compared with the other performer group which used only 20.97 hints on average

($t(9.87) = 2.382$, $p = 0.039$). As we can see the successful learners used different strategies in the Mission Game and Arcade Game. The difference between these two games explains the distinction of their behaviors. In the Mission Game even though the aide agent can offer hints on the expected speech in English and Arabic, the learner would not be able to memorize it if he/she did not build up enough skill level from the Skill Builder lessons due to the complexity of the speech. Therefore, they need to request hints often. In the Arcade Game, especially the beginner levels, expected utterances are relatively short and simple, and therefore medium-leveled skills can be directly applied.

6 Study Changes and Future Work

After the assessment data described in this article were collected, the 2/7 Marines received word that they might have to deploy to Afghanistan instead of Iraq. The 2/7 therefore called a halt to the Iraqi assessment, and made plans to initiate it again with the Dari language spoken in Afghanistan. This is an example of the challenges inherent in conducting *in vivo* evaluations of learning software in the context of training practice. Such evaluations have greater external validity than studies in controlled laboratory settings, but they must adapt to the constraints of the organization's training activities.

Our future work includes the plan to collect more data from other Marines units to find out whether they were successful in their training. We also plan to observe their final live training exercise, in which they must interact with Iraqi role players. This will help to determine how effective their training really was.

7 Conclusions

A critical lesson we learnt from design of game-base training is how to design learning environments to optimize pacing. ITS research doesn't often consider the question how to keep learners engaged for extended periods. This of course is a key issue for computer games, which are typically designed specifically to promote extended play. The experience with Tactical Iraqi shows that this is a critical issue, and the game elements help to maintain a sustainable learning pace.

One of the attractions of game-based learning is that games promote motivation. Our results indicate that motivation is overall a key predictor of learning success. However the experience shows that games also motivate learners to engage in learning of their choice, rather than follow a designated program of instruction. We conclude from this that we need to provide learners with that freedom of choice, yet we should also provide learners advice of what to work on next, to make sure that they are being productive at all times. And that in turn requires instructional planning capability that adapts to the learner's choices, and a learner modeling capability that is works robustly regardless of the learner's choices.

Acknowledgments. This work was sponsored in part by the US Marine Corps, Program Manager for Training Systems (PM TRASYS).

References

1. Baker, R.S., Corbett, A.T., Koedinger, K.R.: Detecting Learner Misuse of Intelligent Tutoring Systems. In: Proceedings of the 7th International Conference on Intelligent Tutoring System, pp. 531–540 (2004)
2. Chan, T.W.: The four problems of technology-enhanced learning. In: Plenary address to AIED 2007 (2007)
3. Conati, C., Klawe, M.: Socially Intelligent Agents to Improve the Effectiveness of Educational Games. In: Proceedings of AAAI Fall Symposium on Socially Intelligent Agents - The human in the loop (2000)
4. Ellis, R.: The study of second language acquisition. Oxford University Press, Oxford (1994)
5. Gardner, R.: Social psychology and second language learning: The role of attitudes and motivation. Edward Arnold, London (1985)
6. Henderson, L., Klemes, J., Eshet, Y.: Just Playing a Game? Educational Simulation Software and Cognitive Outcomes. Journal of Educational Computing Research 22(1), 105–129 (2000)
7. Johnson, W.L.: Serious use of a serious game for language learning. In: Luckin, R., et al. (eds.) Artificial Intelligence in Education, pp. 67–74. IOS Press, Amsterdam (2007)
8. Johnson, W.L., Beal, C.: Iterative evaluation of a large-scale, intelligent game for language learning. In: Artificial Intelligence in Education, IOS Press, Amsterdam (2005)
9. Johnson, W.L., Beal, C., Fowles-Winkler, A., Lauper, U., Marsella, S., Narayanan, S., Papachristou, D., Vilhjálmsson, H.: Tactical Language Training System: An Interim Report. In: Lester, J.C., et al. (eds.) Intelligent Tutoring Systems: 7th International Conference, ITS 2004, pp. 336–345. Springer, Berlin (2004)
10. Johnson, W.L., Marsella, S., Vilhjálmsson, H.: The Tactical Language Training System. In: Proceedings of I/ITSEC 2004 (2004)
11. Johnson, W.L., Vilhjálmsson, H., Marsella, S.: Serious Games for Language Learning: How Much Game, How Much AI? In: Artificial Intelligence in Education. IOS Press, Amsterdam (2005)
12. Kirkpatrick, D.F.: Evaluating Training Programs: The Four Levels, Berrett-Koehler, San Francisco (1994)
13. Prensky, M.: Digital Game-Based Learning. McGraw-Hill (2001)
14. Surface, E.A., Dierdorff, E.C., Watson, A.: Special Operations Language Training Software Measurement of Effectiveness Study: Tactical Iraqi Study Final Report. Special Operations Forces Language Office, Tampa, FL, USA (2007)

Story-Based Learning: The Impact of Narrative on Learning Experiences and Outcomes

Scott W. McQuiggan, Jonathan P. Rowe, Sunyoung Lee, and James C. Lester

Department of Computer Science, North Carolina State University, Raleigh NC 27695
{swmcquig,jprowe,slee7,lester}@ncsu.edu

Abstract. Within the intelligent tutoring systems community, narrative is emerging as an effective medium for contextualizing learning. To date, relatively few empirical studies have been conducted to assess learning in narrative-centered learning environments. In this paper, we investigate the effect of narrative on learning experiences and outcomes. We present results from an experiment conducted with eighth-grade middle school students interacting with a narrative-centered learning environment in the domain of microbiology. The study found that students do exhibit learning gains, that those gains are less than those produced by traditional instructional approaches, but that the motivational benefits of narrative-centered learning with regard to self-efficacy, presence, interest, and perception of control are substantial.

1 Introduction

Narrative is the subject of increasing interest within the intelligent tutoring systems community. Researchers have begun to develop narrative-centered learning environments (NLEs) that combine story contexts and pedagogical support strategies to deliver effective, engaging educational experiences. Contextualizing learning within narrative affords the use of artificial intelligence techniques that tailor narrative and educational content to students' actions, affective states, and abilities. Drawing on an interdisciplinary body of work, including intelligent tutoring systems, embodied conversational agents, and serious games, these environments offer the promise of adaptive, motivating learning experiences to students. NLEs are currently under investigation in a range of domains, including military soft-skills training [7,15], anti-bullying education [1], health intervention education [11], and science learning in microbiology and genetics [12].

By incorporating learning into narrative-based, virtual environments, researchers hope to tap into students' innate facilities for crafting and understanding stories. Contextualized learning experiences are known to encourage regulated learning behavior [14]. Narrative is also well suited to alternative learning paradigms such as guided discovery and inquiry-based learning. Leveraging stories' ability to draw audiences into plots and settings, NLEs can introduce novel perceptual, emotional, and motivational experiences, as well as establish connections between narrative content and pedagogical subject matter in young learners [19]. Further, NLEs can effectively support the factors shown to contribute to student levels of motivation. Such contextual experiences influence student learning and motivation [8].

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 530–539, 2008.

There is a strong theoretical foundation and several active projects that support this line of work [1], but there has been limited empirical investigation of learning outcomes within narrative-centered learning environments. Because of the challenges inherent in developing and deploying these types of systems, learning outcome evaluations in previous work has largely been subjective or preliminary in scope. This papers seeks to provide an empirical basis for the evaluation and investigation of NLEs. It presents results from an empirical study conducted with eighth-grade middle school students interacting with an "early generation" NLE, CRYSTAL ISLAND.

2 Related Work

Much of the work on NLEs has focused on developing AI-based approaches that provide rich, adaptive narrative-based learning experiences and respond appropriately to student actions in the environment. FearNot! is a character-driven learning environment for the domain of anti-bullying social education [1]. The environment emphasizes autonomous, highly affective characters that foster empathetic relationships with students, who in turn offer coping suggestions to the victimized virtual character. FearNot! has been the subject of several small- and large-scale studies, although the subjective nature of the domain renders objective, learning-gain results impractical. Carmen's Bright IDEAS seeks to teach health intervention skills to mothers of pediatric cancer patients [11]. The environment combines autonomous characters with director and cinematographic agents in order to provide a dramatic story that satisfies pedagogical goals. Students control the main character's (Carmen's) decisions as she copes with the stresses and problems inherent in caring for an ill child. Carmen's Bright IDEAS has been the subject of clinical trials, but reported results have also been limited.

Intelligent NLEs have recently been developed for military soft-skills training, particularly in leadership and language learning scenarios. IN-TALE is an interactive narrative system that integrates autonomous character behaviors and an Automated Story Director to provide dramatic simulation experiences for social and cultural leadership training [15]. The system draws upon previous work in narrative planning and believable agent behavior to balance narrative coherence and user-agency in the simulation environment. The Tactical Language and Culture Training System (TLCTS) is a story-centric, serious game designed for language learning [7]. TLCTS use a combination of interactive lessons and games to train students in spoken and non-verbal communication, as well relevant cultural knowledge. Over the course of the last several years, TLCTS has transitioned into widespread use by the US military and other groups. However, large-scale, summative empirical results for learning outcomes have not yet been presented for either IN-TALE or TLCTS [6].

Despite the presence of several promising and successful examples of NLEs, empirical evaluation remains limited. We seek to extend preliminary results in narrative-centered learning by reporting on a controlled experiment assessing learning outcomes between several versions of a NLE and a more traditional, didactic format.

3 Crystal Island

CRYSTAL ISLAND is a narrative-centered learning environment built on Valve Software's Source™ engine, the 3D game platform for Half-Life 2. CRYSTAL ISLAND features a science mystery set on a recently discovered volcanic island. The curriculum underlying CRYSTAL ISLAND's science mystery is derived from the North Carolina state standard course of study for eighth-grade microbiology. Students play the role of the protagonist, Alyx, who is attempting to discover the identity and source of an unidentified infectious disease plaguing a newly established research station. The story opens by introducing the student to the island and members of the research team for which the protagonist's father serves as the lead scientist. Several of the team's members have fallen gravely ill, including Alyx's father. Tensions have run high on the island, and one of the team members suddenly accuses another of having poisoned the other researchers. It is the student's task to discover the outbreak's cause and source, and either acquit or incriminate the accused team member.

CRYSTAL ISLAND's setting includes a beach area with docks, a large outdoor field laboratory, underground caves, and a research camp. Throughout the mystery, the student is free to explore the world and interact with other characters while forming questions, generating hypotheses, collecting data, and testing hypotheses. The student can pick up and manipulate objects, take notes, view posters, operate lab equipment, and talk with non-player characters to gather clues about the source of the disease. During the course of solving the mystery, the student is minimally guided through a five problem curriculum. The first two problems focus on pathogens, including viruses, bacteria, fungi, and parasites. The student gathers information by interacting with in-game pathogen "experts" and viewing books and posters in the environment. In the third problem, the student is asked to compare and contrast her knowledge of four types of pathogens. In the fourth problem, the student is guided through an inquiry-based hypothesis-test-and-retest problem. In this problem she must complete a "fact sheet" with information pertaining to the disease inflicting members of the CRYSTAL ISLAND research team. Once the "fact sheet" is completed and verified by the camp nurse, the student completes the final problem concerning the treatment of viruses, bacteria, fungi, and parasites, and selects the appropriate treatment plan for sickened CRYSTAL ISLAND researchers. The story and curriculum are interwoven throughout the interactive experience.

4 Method

4.1 Participants

There were 88 female and 91 male participants varying in age and race. Approximately 2% of the participants were American Indian or Alaska Native, 5% were Asian, 29% were Black or African American, 58% were Caucasian, 6% were Hispanic or Latino, and 6% were of other races. Participants were all eighth-grade students ranging in age from 12 to 15 ($M = 13.27$, $SD = 0.51$). The students had recently completed the microbiology curriculum mandated by the North Carolina state standard course of study before receiving the instruments, tests, and interventions of this experiment.

4.2 Materials and Apparatus

The pre-experiment paper-and-pencil materials for each participant were completed one week prior to intervention. These materials consisted of a researcher generated CRYSTAL ISLAND curriculum test, demographic survey, Achievement Goals Questionnaire [4], Self-Efficacy for Self-Regulated Learning scale (SESRL) [3], Science Self-Efficacy scale, modified from [13], and Immersion Tendencies Questionnaire [21]. The CRYSTAL ISLAND curriculum test consists of 23 questions created by an interdisciplinary team of researchers and was approved for language and content by the students' eighth-grade science teachers. Elliot and McGregor's Achievement Goals Questionnaire is a validated instrument which measures four achievement goal constructs (mastery-approach, performance-approach, mastery-avoidance, and performance-avoidance goals) [4]. Bandura's Self-Efficacy for Self-Regulated Learning scale [3] consists of 11 items rated by participants on a 7-point Likert scale. Witmer and Singer developed and validated the Immersive Tendencies Questionnaire (ITQ) to measure individual predispositions towards presence experiences [21]. The ITQ consists of three subscales: activity involvement tendency, activity focus tendency, and video game playing tendency. Participants indicate their level of tendency for each item on a 7-point Likert scale. Witmer and Singer found individual tendencies, as recorded by the ITQ, to be predictive of presence (discussed in Section 6.2) [21].

Post-experiment materials were completed immediately following intervention. These materials consisted of the same CRYSTAL ISLAND curriculum test, Achievement Goals Questionnaire [4], Science Self-Efficacy scale, an interest scale [19], and the Presence Questionnaire [21]. The interest scale was adapted from those used by Schraw to capture differences across groups and to examine within-subject relationships with learning outcomes [19]. Participants' presence experience was captured with the Presence Questionnaire (PQ) developed and validated by Witmer and Singer [21]. The PQ contains several subscales including involvement/control, naturalism of experience and quality of the interface scales.

5 Design and Procedure

5.1 Design

The experiment randomly assigned the entire eighth grade population of Centennial Campus Middle School in Raleigh, North Carolina to four groups: holdout, CRYSTAL ISLAND narrative condition, CRYSTAL ISLAND minimal-narrative condition, or PowerPoint condition (see Table 1 for condition breakdown). Participants in the holdout condition did not receive an intervention and served as the control group for this experiment and planned longitudinal studies. In the remaining three conditions, students received an intervention consisting of the CRYSTAL ISLAND microbiology curriculum delivered in one of three formats. The CRYSTAL ISLAND narrative condition supplemented the curriculum with the full CRYSTAL ISLAND narrative, including a poisoning scenario, character back-stories, and rich character personalities. The CRYSTAL ISLAND minimal-narrative condition supplemented the curriculum with the minimal story required to support the curriculum. In this condition, the story strictly consisted of research members

falling ill and the request for the student to identify the mysterious illness. The minimal-narrative condition did not include the poisoning storyline, character back-stories, or explicit character personality. The PowerPoint condition consisted of a narrated PowerPoint presentation of the same curriculum that was used in CRYSTAL ISLAND. Much of the text and images of the slides actually appear in CRYSTAL ISLAND in the form of books, posters, and character dialogue. The PowerPoint condition did not contain a corresponding storyline.

Table 1. Subject population by condition

Condition n = 179							
Holdout n = 29		Narrative n = 60		Min-narrative n = 56		PowerPoint n = 33	
Female n = 18	Male n = 11	Female n = 30	Male n = 30	Female n = 24	Male n = 32	Female n = 16	Male n = 17

5.2 Participant Procedure

Participants entered the experiment room having completed the pre-test and instrumentation one week prior to the intervention. Participants were first instructed to review CRYSTAL ISLAND instruction materials. These materials consisted of the CRYSTAL ISLAND back-story and task description, a character handout, a map of the island, and a control sheet. Participants were then further directed on the controls via a presentation explaining each control in detail.

Participants in the three intervention conditions (narrative, minimal-narrative, and PowerPoint) were given 50 minutes to work on solving the mystery. Solving the mystery consisted of completing a number of goals including learning about pathogens, viruses, bacteria, fungi, and parasites, compiling the symptoms of the researchers who had falled ill, recording features of hypothesized diseases causing the CRYSTAL ISLAND illness, testing a variety of possible sources, and reporting the solution (cause and source) to the camp nurse to design a treatment plan.

Immediately after solving the science mystery of CRYSTAL ISLAND, or 50 minutes of interaction, participants completed the post-experiment questionnaires. First to be completed was the CRYSTAL ISLAND curriculum test, followed by the remaining post-experiment questionnaires described above. Completion of post-experiment materials took no longer than 35 minutes for participants. In total, sessions lasted 90 minutes.

6 Results

6.1 Learning Outcomes

Investigating learning in CRYSTAL ISLAND as measured by the difference of post-test and pre-test scores, we find that, overall, students exhibited learning gains (learning gain, M = 0.07, SD = 0.14). On average, students answered 1.6 (SD = 3.3) more questions correctly on the post-test than on the pre-test. Matched pairs t tests (comparing post-test to

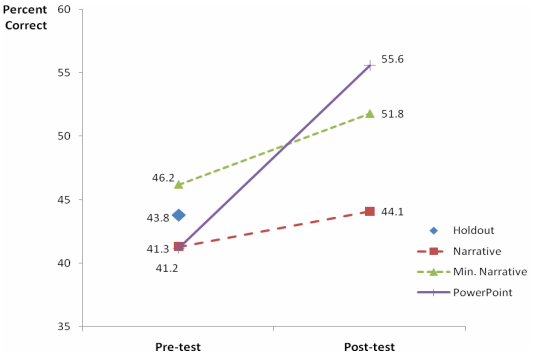


Fig. 1. Learning gains (pre- to post-test) by condition

pre-test scores) indicate that these learning gains are significant overall, $t(149) = 5.51$, $p < 0.0001$, and significant (although weakly significant in the narrative condition) within each condition (narrative condition: $t(58) = 1.43$, $p = 0.07$, minimal-narrative condition: $t(55) = 2.97$, $p < 0.005$, and the PowerPoint condition: $t(34) = 5.74$, $p < 0.0001$). Further, the learning gains in each condition were significantly different, $F(2, 149) = 10.38$, $p < 0.0001$. There was no significant difference among pre-test scores between conditions, $F(4, 179) = 0.94$, $p = 0.44$. The largest learning gains occurred in the PowerPoint condition ($M = 0.15$, $SD = 0.15$), followed by learning gains in the minimal-narrative condition ($M = 0.06$, $SD = 0.14$), and the lowest learning gains in the narrative condition ($M = 0.02$, $SD = 0.11$). Students in the hold out condition did not take a post-test, and therefore no learning gain results are available for that condition. The CRYSTAL ISLAND curriculum test consisted of 23 items leading to a learning gain of 0.043, which equates to getting one additional question correct in the post-test compared to the pre-test. Thus, on average, students in the PowerPoint condition answered 3.5 more questions correctly ($SD = 3.6$) on the post-test, with participants in the minimal-narrative and narrative conditions answering 1.3 ($SD = 3.2$) and 0.5 ($SD = 2.7$) more questions correctly, respectively. Learning gains are depicted in Figure 1. If we consider only the students who completed the CRYSTAL ISLAND mystery in the narrative condition, we find no significant difference between post-test scores with the PowerPoint condition, $F(1, 48) = 0.32$, $p = 0.58$. However, the learning gains in the PowerPoint condition were somewhat significantly better than the students who finished the CRYSTAL ISLAND narrative experience, $F(1, 48) = 4.09$, $p = 0.05$.

Interestingly, there was an effect of gender on learning in CRYSTAL ISLAND. When we consider only the problems on the CRYSTAL ISLAND curriculum test for which students were exposed to (not all students solved the CRYSTAL ISLAND mystery and completed all problem-solving activities), we find gender played a significant role, $F(1, 114) = 4.44$, $p = 0.037$. In CRYSTAL ISLAND, on average, male students got an additional 1.3 problems correct ($SD = 2.4$) on post-tests compared to pre-tests, while female students got an additional 0.4 problems correct ($SD = 1.7$).

6.2 Presence Outcomes

Presence contributes to the goal of transparency in technology-mediated interactions. Although there has been substantial debate on formal definitions, there is a general consensus that presence describes a user's sense of "being there" when interacting with a mediated environment [5, 17]. Presence has been alternatively defined as "the perceptual illusion of nonmediation" [9], as well as "the subjective experience of being in one place

or environment, even when one is physically situated in another" [21]. Witmer and Singer further distinguish presence from involvement. Involvement refers to the degree of attention and meaning devoted to some set of stimuli [21]. Here we report on students' reported sense of presence while interacting in the CRYSTAL ISLAND storyworld (narrative and minimal-narrative conditions only).

Narrative had a significant effect on student presence, $F(1, 115) = 4.23$, $p = 0.04$. Higher presence was reported in the narrative condition ($M = 147.35$, $SD = 30.6$) compared to the minimal-narrative condition ($M = 136.5$, $SD = 25.8$). Gender was also found to have a weakly significant effect on presence, $F(1, 115) = 2.87$, $p = 0.09$, with females reporting higher levels of presence ($M = 146.9$, $SD = 26.1$) than males ($M = 137.9$, $SD = 30.5$). Students reporting high-levels of interest (as gauged by the interest scale modified from [19]) reported higher levels of presence than students with low-levels of interest. There was a significant correlation of interest with student presence, $r(114) = 0.36$, $p = 0.0001$, and several of the PQ's subscales, including: involvement/control ($r(114) = 0.42$, $p < 0.0001$), naturalism of experience ($r(114) = 0.27$, $p = 0.003$), and resolution ($r(114) = 0.29$, $p = 0.002$). Self-efficacy and presence also had a significant interaction. Students with high science efficacy reported higher levels of presence than less efficacious students, $r(114) = 0.35$, $p = 0.0001$. Likewise, students reporting greater levels of involvement and control (a PQ subscale) also reported higher science efficacy, $r(114) = .28$, $p = 0.002$.

Student goal orientation was found to affect presence as well. In particular, there was a significant effect of mastery approach on presence in both CRYSTAL ISLAND conditions, $F(1, 114) = 8.65$, $p = 0.004$, and performance avoidance on presence, $F(1, 114) = 4.59$, $p = 0.034$. Mastery oriented students reported greater levels of presence than performance-oriented students. Students who sought to avoid negative performance outcomes also reported higher levels of presence than students who did not seek to avoid negative performance outcomes.

7 Discussion

The experiment found that students who interacted with the CRYSTAL ISLAND environment achieved significant learning gains. While pre- to post-test performance differences were greatest in the PowerPoint condition, the findings support the hypothesis that students received clear motivational benefits from interacting with CRYSTAL ISLAND. Further, student levels of presence had significant relationships with factors relevant to learning and motivation, including self-efficacy, interest, involvement/control, and goal orientation. While learning gains were higher in the minimal-narrative condition, students reported higher levels of presence in the narrative condition, carrying promising implications for motivation.

Drawing upon the experiment's learning gain results, it is possible that the narrative condition's additional story content overloaded cognition, enabling students to learn more without the supplemental storyline on proximal assessment. An important direction for future work is conducting longitudinal studies to determine the the long-term effects of narrative on learning and inform scaffolding strategies for reducing cognitive load.

The study found significant effects of presence, with higher reported presence in the narrative condition. With the benefits of efficacious learners having been widely demonstrated [2,22], it is important to note that higher presence levels also lead to higher levels of reported self-efficacy. If further study can identify the narrative factors that contribute to motivation and efficacy, we can enhance the ability of NLEs to support student problem solving, increase student effort, persistence, and resilience when confronted with failure, and raise the levels of success students are likely to achieve [2, 18, 22].

When considering the Involvement/Control subscale of the Presence Questionnaire [21], the findings indicated that high levels of Involvement/Control are correlated with higher reports of self-efficacy. Perception of control is known to have motivational benefits [10]. As a factor contributing to presence, involvement/control suggests probable relationships between presence and motivation. The findings of this study highlight the potential for positive connections between narrative and motivation that deserve further investigation. Further exploration of these relationships will contribute to a deepened understanding of the narrative factors that relate story content, presence, learning, motivation, and self-efficacy, as well as our ability to regulate these factors in an effort to support pedagogical objectives.

The study also found an effect of student goal orientation on perceptions of presence among the middle school participants. The gaming environment, on which CRYSTAL ISLAND is built, may have had an effect on performance-oriented students, encouraging them to attempt to solve the mystery quickly. Meanwhile, it seems that mastery oriented students, who tend to measure accomplishments by learning successes, reported a greater perception of presence. It is probable that mastery oriented students were more likely to take their time throughout their interactions, focusing their attention on the content of learning environment so that their presence experience was heightened.

8 Limitations

The experiment was designed to control for time on task, allowing 50 minutes for the intervention. As a result of this constraint and the amount of content in CRYSTAL ISLAND, only 49 of the 116 CRYSTAL ISLAND participants finished or were working on the final problem at the end of the 50 minute session. An alternative design, which will be adopted in future work, would consider controlling for task completion. Another limitation is that this study, at the time of writing, does not include a longitudal test to assess the hypothesized benefits of narrative.

9 Conclusion

Narrative is receiving increasing attention in the ITS community as a medium for contextualizing learning in meaningful ways while creating rich, engaging experiences for learners. To date, there has been little empirical work supporting the use of narrative in interactive learning environment. In a controlled experiment with an "early-generation" NLE, it was found that students do in fact exhibit learning gains, that those gains are less than those in produced by traditional instructional approaches, but that the motivational

benefits of narrative-centered learning, particularly with regard to self-efficacy, presence, interest, and perception of control, are substantial.

The results highlight two important directions for future work. First, the contextual benefits of narrative may be more pronounced in a longitudinal evaluation of learning rather than in the assessment administrated immediately following intervention as in the study reported here. Second, it is important to begin exploring the educational role of the myriad components of narrative in learning episodes, such as plot coherence, drama, and character identification, and their impact on problem solving, learning outcomes, and engagement [16].

Acknowledgments. The authors wish to thank members of the IntelliMedia lab for their assistance, Omer Sturlovich and Pavel Turzo for use of their 3D model libraries, and Valve Software for access to the Source™ engine and SDK. This research was supported by the National Science Foundation under Grant REC-0632450. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Aylett, R., Louchart, S., Dias, J., Paiva, A., Vala, M.: FearNot! - An Experiment in Emergent Narrative. In: Panayiotopoulos, T., Gratch, J., Aylett, R.S., Ballin, D., Olivier, P., Rist, T. (eds.) IVA 2005. LNCS (LNAI), vol. 3661, pp. 305–316. Springer, Heidelberg (2005)
2. Bandura, A.: Self-efficacy: The exercise of control. Freeman, New York (1997)
3. Bandura, A.: Guide for constructing self-efficacy scales. In: Pajares, F., Urdan, T. (eds.) Self-Efficacy Beliefs of Adolescents, pp. 307–337. Information Age Publishing, Greenwich (2006)
4. Elliot, A., McGregor, H.: A 2 x 2 achievement goal framework. Journal of Personality and Social Psychology 80(3), 501–519 (2001)
5. Insko, B.: Measuring presence: Subjective, behavioral and physiological methods. In: Riva, G., Davide, F., IJsselsteijn, W. (eds.) Being There: Concepts, Effects and Measurements of User Presence in Synthetic Environments, pp. 109–119. Ios Press, Amsterdam (2003)
6. Johnson, W.L., Beal, C.: Iterative Evaluation of an Intelligent Game for Language Learning. In: Proceedings of AIED 2005. IOS Press, Amsterdam (2005)
7. Johnson, W.L.: Serious Use of a Serious Game for Language Learning. In: Proceedings of AIED 2007. IOS Press, Marina del Rey (2007)
8. Linnenbrink, E.A., Pintrich, P.R.: Multiple goals, multiple contexts: The dynamic interplay between personal goals and contextual goal stresses. In: Volet, S., Jarvela, S. (eds.) Motivation in Learning Contexts: Theoretical Advances and Methodological Implications, pp. 251–269. Elsevier, New York (2001)
9. Lombard, M., Ditton, T.: At the heart of it all: The concept of presence. Journal of Computer Mediated Communication 3(2) (1997)
10. Malone, T., Lepper, M.: Making learning fun: A taxonomy of intrinsic motivations for learning. In: Snow, R., Farr, M. (eds.) Aptitude, Learning, and Instruction: III. Conative and Affective Process Analyses, pp. 223–253. Erlbaum, Hillsdale (1987)

11. Marsella, S., Johnson, W.L., LaBore, C.: Interactive pedagogical drama for health interventions. In: Proceedings of the 11th International Conference on Artificial Intelligence in Education, Australia (2003)
12. Mott, B., Lester, J.: Narrative-centered tutorial planning for inquiry-based learning environments. In: Proceedings of the 8th International Conference on Intelligent Tutoring Systems, Jhongli, Taiwan, pp. 675–684 (2006)
13. Nietfeld, J.L., Cao, L., Osborne, J.W.: The effect of distributed monitoring exercises and feedback on performance and monitoring accuracy. Metacognition and Learning 1(2), 159–179 (2006)
14. Perry, N.E.: Young children's self-regulated learning and the contexts that support it. Journal of Educational Psychology 90, 715–729 (1998)
15. Riedl, M., Stern, A.: Believable agents and intelligent story adaptation for interactive storytelling. In: Proceedings of the 3rd International Conference on Technologies for Interactive Digital Storytelling and Entertainment, Darmstadt, Germany (2006)
16. Rowe, J., McQuiggan, S., Lester, J.: Narrative presence in intelligent learning environments. In: Working Notes of the 2007 AAAI Fall Symposium on Intelligent Narrative Technologies, Washington (2007)
17. Schubert, T., Friedmann, F., Regenbrecht, H.: Embodied presence in virtual environments. In: Paton, R., Neilson, I. (eds.) Visual Representations and Interpretations, pp. 269–278. Springer, London (1999)
18. Schunk, D., Pajares, F.: The development of academic self-efficacy. In: Wigfield, A., Eccles, J. (eds.) Development of Achievement Motivation, pp. 15–31. Academic Press, San Diego (2002)
19. Schraw, G.: Situational interest in literary text. Contemporary Educational Psychology 22, 436–456 (1997)
20. Wells, C.: The Meaning Makers: Children Learning Language and Using Language to Learn. Heinnemann, Portsmouth (1986)
21. Witmer, B., Singer, M.: Measuring presence in virtual environments: A presence questionnaire. Presence: Teleoperators and Virtual Environments 7(3), 225–240 (1998)
22. Zimmerman, B.: Self-efficacy: An essential motive to learn. Contemporary Educational Psychology 25, 82–91 (2000)

An Architecture for Combining Semantic Web Techniques with Intelligent Tutoring Systems

Pedro J. Muñoz Merino and Carlos Delgado Kloos

Carlos III University of Madrid, Department of Telematics Engineering,
Avda de la Universidad, 30, E-28911 Leganés (Madrid), Spain
{pedmume,cdk}@it.uc3m.es

Abstract. There are several possible applications of Semantic Web techniques in Intelligent Tutoring Systems (ITSs) and important advantages can be obtained from their application. This paper presents an architecture to combine Semantic Web techniques with ITSs, explaining its elements, relationships, challenges, and the different design criterions, offering some guidelines to make decisions when different implementation solutions are possible. We have implemented an instance of this architecture using the XTutor ITS and the CWM (Closed World Machine) Semantic Web reasoner. The implemented framework permits personalization of problems with hints for students. The paper also describes this specific implementation of the general architecture as well as some user adaptation examples with the implemented framework.

1 Introduction

There are many different ITSs; a categorization of them can be seen in [1]. The functions that such systems can perform vary; some examples are student assessment [2], or adaptive hypermedia [3], [4]. Some ITSs use educational information in a proprietary way, while others follow educational standards to enable interoperability.

There are several advantages in using Semantic Web techniques with ITSs, which justifies its combination. At present, few works have discussed architectures for enabling this combination. Reference [6] presents a Web service-based architecture to enable Semantic Web methods in adaptive hypermedia. Our architecture approach presents a different point of view, because it focuses on the relationship between Semantic Web reasoners (from now on reasoners) and existing ITSs, explaining its architectural elements, design criterions, implementation problems, etc. Moreover, a few works exist which explain specific system solutions but they do not cover the different architectural design criterions and implementation problems. In this work, we contribute to these issues, explaining an architecture for combining Semantic Web with ITSs that is general enough. Furthermore, we explain some challenges that Semantic Web techniques present when applied with ITSs; those challenges require architecture implementation decisions, and we give some recommendations.

We illustrate a specific case of the general architecture with the implementation of a framework to provide adaptive hints in problem-based learning based on this architecture and using the CWM reasoner [7], a new specification of hints we defined [8], and a hint player [8] that we implemented into the XTutor ITS [9].

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 540–550, 2008.

The remainder of this paper is organized as follows. In Section 2, there is an outline of the related work about hint tutors and Semantic Web applications in education. Section 3 explains the general proposed architecture for combining Semantic Web with ITSs. Section 4 presents a specific implementation of the architecture for adaptive hints. In Section 5, some user adaptation examples with the implemented framework for adaptive hints are shown. Finally, Section 6 is devoted to the conclusions.

2 Related Work

2.1 Hint Tutors

It is clear that the provision of hints as a teaching methodology during problem solving has a positive impact on student learning (e.g. [10]). Therefore, several hint ITSs exist, as well as related works about this topic (e.g. [11], [12], [13]). Several of these systems allow the personalization of hints but using techniques different from the Semantic Web, such as Bayesian Networks (e.g. [11]).

2.2 Semantic Web Applications in Education

Ontology engineering is a key aspect for the success of Semantic Web. In [14] there is a clear vision of the use of ontologies for Artificial Intelligence in education. There are educational ontologies for different purposes such as competences [15], or domain ontologies that can be derived from the text [16]. A repository of ontologies for education was built [17]. Some ontologies have also been created to enable the inclusion of e-learning standards into the Semantic Web, such as for SCORM (Sharable Content Object Reference Model) [18], or IMS-LD (Learning Design) [19], [20].

There are different possible applications of Semantic Web in education. We focus on adaptive applications. The different types of adaptive hypermedia are explained in [3]. Adaptive applications using the Semantic Web have been proposed for contents [21], assessments [22] or educational feedback [23]. In the implemented framework described in this paper, we focus on a different domain, which is the provision of adaptive hints, taking into account the defined elements of our specification of hints.

3 General Architecture

Fig. 1 shows a graphical representation of the proposed architecture. Section 3.1 explains its elements and relationships. Section 3.2 describes the design criterions, analyzing the advantages and disadvantages of the different possible decisions.

3.1 Description of the Elements and Their Relationships

The *Data Storage* contains the main information that the system can use. The data is divided into two groups: *static* and *dynamic*. Information is considered static when it does not change (e.g. specific course objectives), while information is considered dynamic when it can change (e.g. the number of times a student has visited some content). In any case, teachers or system designers must provide all the static information at the beginning, and the initial state of dynamic information when applicable.

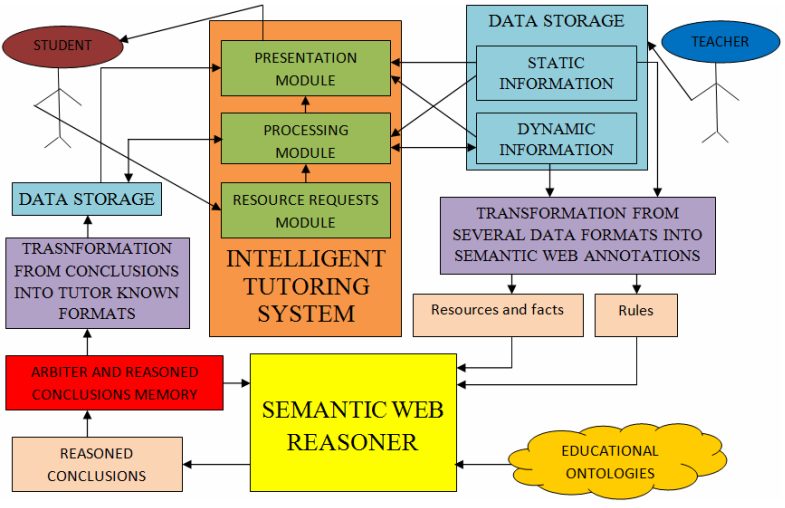


Fig. 1. General Architecture for Combining Semantic Web techniques with ITSs

The *Semantic Web reasoner* is a tool that permits logic reasoning based on a set of *rules*, *resources* and *facts* according to defined *ontologies*, reaching *conclusions* as a result. Rules, resources and facts must be in specific formats (e.g. *RDF/XML* [24] or *N3* [25]). Therefore, a *transformation module* is required to convert the different information formats into Semantic Web annotations (unless some information resources are directly annotated in Semantic Web languages, for which the transformation module does nothing).

The *Arbiter and reasoned conclusions memory* selects the conclusions to store in each moment in order not to repeat already made inferences. The *reasoned conclusions* of the reasoner are in a format that is not usually understandable by ITSs. Therefore, another *transformation module* is required to convert those conclusions into ITS known formats. Here again, some conclusions may not need transformation if the ITS does not use this information (but the reasoner may use it again, if it represents feedback conclusions to the reasoner as shown in Fig. 1).

Lastly, the ITS receives *Resources Requests* from students that are then transmitted to the ITS *Processing module* which takes the proper actions, having data storage information as input. Dynamic data can be modified as a result of the ITS processing module. Finally, some resources (responses to requests) must be presented to the students, using the ITS *Presentation module* that can obtain any information from the Data Storage. Defined educational specifications may bring together (e.g. in XML files) processing and presentation information.

3.2 Design Criterions and Decisions

Several design criterions and decisions must be taken when implementing the architecture. Fig. 2 shows an overview of the different issues to decide.

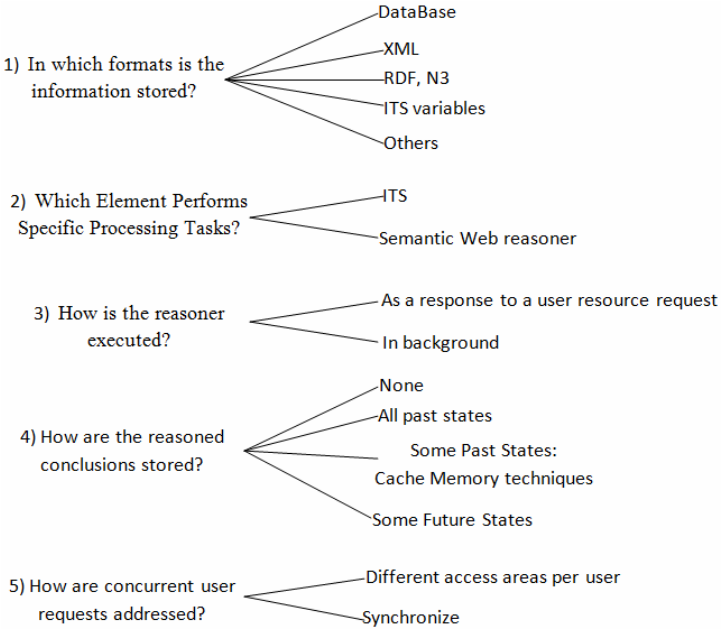


Fig. 2. List of decisions to be taken in the defined architecture

In Which Formats Is the Information Stored? The same information can be stored in different formats (e.g. a data base format, XML, N3 or ITS variables). We must make a decision about which format to use, based on the following points:

1) *Desired level of interoperability*. The desired level of interoperability should be decided (e.g. with educational systems that interpret XML specifications, or with Semantic Web tools that interpret Semantic Web formats such as RDF or N3). It may be the case that interoperability is not important, so any formats are possible.
2) *Existing implemented tools:* For example, if we had an ITS player already implemented which interprets an XML format for assessments, then information about assessment description might be put into this format to take advantage of the tool.

Sometimes the information must be replicated in different formats. The designer can write the information directly or a transformation module can be used.

Which Element Performs Specific Processing Tasks? To decide which element will perform each specific task, we should achieve a balance between these issues:

1) *The formats of the existing information*.
2) *The desired rules for reuse*. If we want to reuse some Semantic rules between different systems (e.g. ITSs), then the reasoner should do the processing.
3) *Execution Time*. In general, an ITS performs a processing task more quickly than the reasoner. For tasks where execution time is critical, this is important.

How is the Reasoner Executed? The reasoner can be executed as a response to a student resource request or in background. The advantages of the former are:

1) R*esponses to the students are up to date*, because the reasoner is executed for each specific request.
2) *It is not necessary to store past reasoned conclusions*. Since each request is transmitted to the reasoner, then the conclusion can be reached at that moment.
But the main advantage of executing the reasoner in background is the *reduction of the system response time to the user*. When a user request arrives, the system may respond to the user with some data that had been inferred previously, so there is no extra time for reasoning on that specific request.

Finally, it is important to note that both methods can be combined.

How Are the Reasoned Conclusions Stored? For each application, there is a set of different input combinations that need reasoning processing. For example, it could be the combinations between the different user states and educational resources. The different possibilities regarding which reasoned conclusions should be stored are:

1) *All past states*: All conclusions from any past states are stored. Indeed, all the states can be reasoned before the system starts working. With this solution, we save reasoning time, not having to repeat the same thing several times. This is recommended if there are few states, no storage limitations, and the response time is critical.
2) *Some past sates*. Techniques similar to cache memory can be used to select the past stored states. This is recommended when there are a lot of different possible states, some memory limitation or the response time is not so critical.
3) *None*. This is recommended when the response time for a request is not critical.
4) *Some Future States*. It consists of doing the reasoning for selected future states considering the greatest likelihood of appearing based on the present ones. This is recommended in the same cases as *Some past states*, and they can be applied together.

How Are Concurrent User Requests Addressed? Concurrent user request problems occur whenever the reasoner or/and the ITS try to write simultaneously to data that is common to several users. Solutions can include a specific storage area for each user or synchronize the part of code that accesses shared resources. Another effect of the concurrency problem is that the response time can increase because of the techniques to avoid it. This is particularly important for the reasoner. For each user request, at least one process related to the reasoning is created. This is more time-consuming than a thread for each request. In addition, if there are synchronized parts, then only one request is at the CPU at each time which increments the response time.

4 Framework Implementation for Personalized Hints

This section presents a specific implementation of the general architecture for achieving personalized hints. Section 4.1 introduces XTutor and the extension module of hints we developed. Section 4.2 explains the specific elements of the architecture. Finally, Section 4.3 explains the different design criterions and implementation decisions.

4.1 Overview of XTutor and Its Hint Player

XTutor [9] is an ITS developed at the Massachusetts Institute of Technology. It includes a library of standard tags to provide some types of questions for students. We complemented it with a new library (*hints.tags.py*) to allow XTutor to provide hints. Full details of our specification of hints are in [8]. The questions with hints are created as XML files (called XDOCs) and XTutor is able to interpret and run them [8].

4.2 Elements of the Implemented Architecture

Fig. 3 shows the architecture that we implemented to obtain adaptive hints. This architecture is a particular case of the architecture presented in Fig. 1, and the same colour denotes the same function. In this implementation, the reasoner used is CWM [7], the ITS used is XTutor, and the Semantic Web language is *N3* [25].

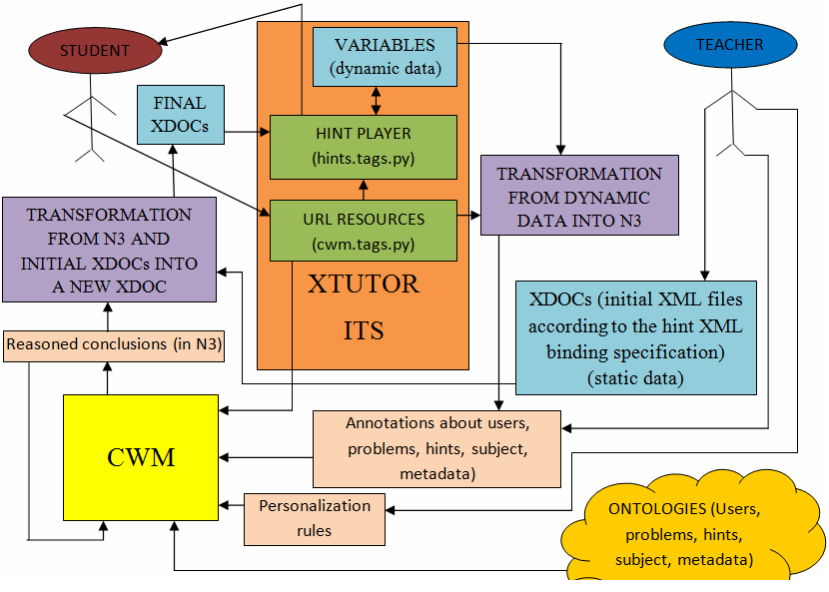


Fig. 3. Implemented Architecture for achieving personalized hints

The student requests an URL resource. This request is received by the *cwm.tags.py*. That is a program file we have implemented which executes some processing tasks. It calls CWM several times. Reasoning is performed by CWM based on the *personalization rules*, obtaining *conclusions*. The data that CWM receives as input are previous CWM conclusions related to the same user request; and *Semantic annotations* about users (preferences, knowledge, etc.), problems and hints (such as the number of students that solved the problem correctly without hints), and the subject domain. For each of these aspects, there is an *ontology*. Some static information (XML, rules and Semantic annotations) is written at the beginning (e.g. the *XDOC* initial files that teachers create that are XML files representing problems and hints), while other

information changes dynamically and is stored as XTutor *variables* (such as the number of students that answered a problem without hints correctly) and needs a conversion from XTutor to N3, so a *transformation module* is required.

Once the whole sequence of specific rules is performed, a final conclusion is obtained by CWM. This conclusion is related to the types, number of hints, etc. that will be applied to the requested resource. Next, as part of the *cwm.tags.py* file there is a *transformation module* from the N3 final conclusions and the initial XDOCs to a final XDOC compliant with the defined specification of hints. The initial XDOC is transformed to a different XDOC based on the reasoned conclusions. At present, the initial XDOC is not used as an input for reasoning by CWM so it is only generated in XML format by the teacher; however we will introduce it too in the future. To do so, a transformation module from XDOC formats into N3 annotations will be required.

Finally, the *cwm.tags.py* calls the hint player we implemented. This module receives the final XDOC as input and it runs it. The hint player performs processing and presentation tasks. The hint player processes information using state variables (e.g. present scoring, hints viewed or hints answered correctly), modifies variables according to the interaction, and presents an HTML response page to the student with the problem with hints. The problem with hints is personalized according to the conclusions inferred by CWM. Some of the variables modified in XTutor as a result of the interaction have an effect on the next CWM reasonings. For example, if a student responds incorrectly to a problem, then the student's knowledge level will decrease in the concepts covered by such problem. These necessary dynamic data variables are transmitted through the transformation module to obtain N3 annotations to be used in reasonings.

4.3 Decisions in the Specific Implemented Architecture

Now, we tackle the decisions made. Firstly, we have the following information:

1) *XDOCs describing problems with hints.* This is to allow interoperability at the XML level, to have data in a format understandable by the XTutor hint player (as it can perform quick processing and presentation of XML files), and because at this moment we do not need to reason about this aspect in CWM.
2) *Some information about users, problems, hints, subject concepts and rules in N3*. This is because we want to perform all the processing related to personalization with CWM (this is for code reuse, high level abstraction language, etc.), so it is necessary to write it in a CWM understandable format. In addition, we want to have Semantic Web interoperability of such aspects.
3) *Dynamic information that changes between interactions.* The database and the XTutor variables are the dynamic information. Since we do not need interoperability for this information, the quickest way for processing is to store such information in XTutor proprietary formats. But the information that is needed for reasoning is transformed, so this information will be replicated in two different formats.

CWM performs some processing tasks (to determine the personalized hints) because of rule reuse, while XTutor performs the other processing tasks.

At present, CWM is executed for each user request, but not in background. In addition, there are no conclusions stored (neither past nor future conclusions). This is

because we have not had time response problems with the present implemented rules and number of users accessing to the system. But in case rules implied a higher processing time, it would then be worthwhile to consider other techniques.

Finally, we are using different access areas per user to avoid concurrent user requests problems. There is no concurrency problem for static or dynamic data that is controlled by XTutor because XTutor controls concurrency by itself. The only data that can bring concurrency problems are the final XDOC generated. Note that if some conclusions were stored (past or future conclusions), then a possible solution would be to store them as a part of the final generated XDOC (this is permitted by the specification of hints defined). In this case the initial XDOC for each request would be the latest XDOC generated and it would be necessary to synchronize access to the final XDOCs, since several users may want to write concurrently in the same XDOC. Furthermore, CWM would be executed as a response to a request only in case the data related to the incoming request would not have been reasoned previously.

5 User Adaptation Examples

Figures. 4, 5, 6 and 7 show four different user adaptation examples in our framework. All the users request the same URL resource, so they obtain the same root problem (a multiple choice about servlets). But each student receives different personalized hints as a result of the reasoning performed by CWM. The first student (Fig. 4) does not receive a hint because he/she has a strong knowledge level in all the concepts.

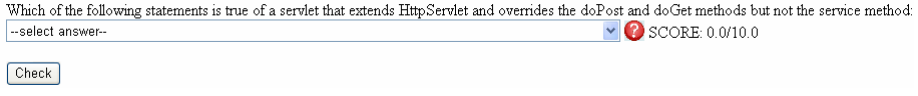


Fig. 4. User 1 adaptation example

The second student (Fig. 5) receives one problem as a hint about the only concept included in the root problem that the student does not master. Among the candidate hints, the one whose level of difficulty is appropriate to the student level is selected.

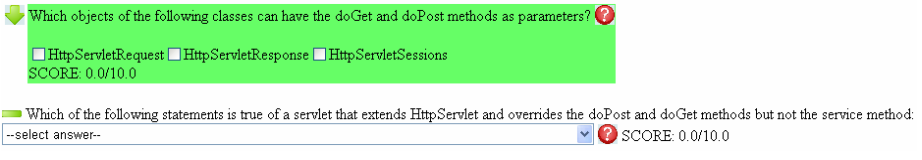


Fig. 5. User 2 adaptation example

Fig. 6 shows a student that has a low knowledge level for all the concepts included in the root problem, so he/she is presented with a sequence hint composed by four problems, one for each concept.

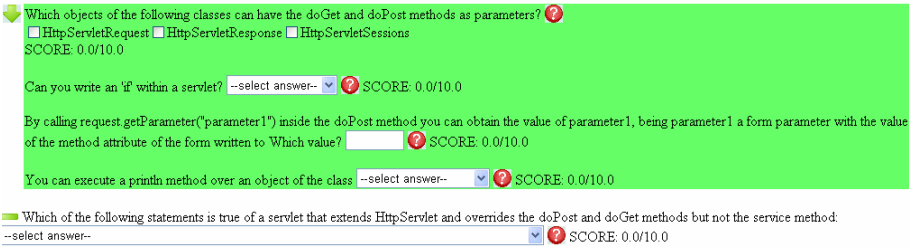


Fig. 6. User 3 adaptation example

User 4 does not receive the hints immediately (Fig. 7). Instead, he/she receives a list with meta-information about the concepts covered for each hint he/she can see (which it is in our hint terminology a hint group). This is because User 4 has a combination of features in his/her personality model, so CWM always provides a hint group for this user, and it will always provide as many hint possibilities as concepts included in the initial root problem, but he/she will be able to select only a maximum of *n* hints, being *n* the number of concepts the student does not know well. In this case User 4 had a low level only on one concept, so he/she could only select a maximum of one hint out of the four available.

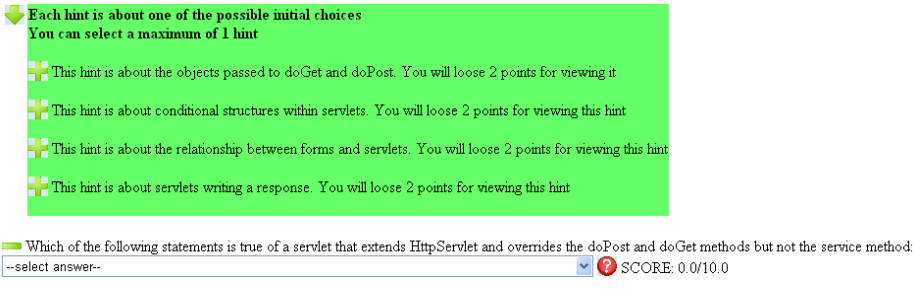


Fig. 7. User 4 adaptation example

In these user adaptation examples, the number and types of hints provided, the concepts chosen, etc. for each specific student are a result of the reasoning by CWM based on the different data. Finally, the conclusion results are transformed into XML files understandable by the XTutor hint module, which is the one that performs the final presentation to the users.

6 Conclusions and Future Work

In this paper we have presented an architecture for combining Semantic Web techniques with ITSs. This architecture is feasible to develop as we have implemented a specific case of it, for the provision of personalizing hints using the CWM Semantic Web reasoner, and the XTutor ITS.

During the implementation we encountered different problems and specific design criterions were required. We analyzed these different problems and made decisions for our particular case, but then we generalized the different design criterions which we explain in this paper together with some guidelines for taking decisions.

The implemented framework introduces Semantic Web techniques in the provision of adaptive hints that are personalized for students. In addition, it combines the use of a new defined XML specification of hints, the XTutor hint player that we implemented, the CWM reasoning capabilities, and other features of the XTutor.

At present, we are working in introducing more personalized rules in the system, extending the existing ones. We are planning to introduce this framework in classroom during this year.

Acknowledgments. Work partially funded by Programa Nacional de Tecnologías de la Sociedad de la Información, MEC-CICYT project MOSAIC-LEARNING TSI2005-08225-C07-01 and 02.

References

1. Murray, T.: Authoring intelligent tutoring systems: An analysis of the state of the art. International Journal of Artificial Intelligence in Education 10, 98–129 (1999)
2. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L.: Andes physics tutoring system: Five years of evaluations. In: 12th International Conference on Artificial Intelligence in Education, pp. 678–685. IOS Press, Amsterdam (2005)
3. Brusilovsky, P.: Adaptive Hypermedia. User Modeling and User-Adapted Interaction 11(1/2), 87–110 (2001)
4. Brusilovsky, P., Peylo, C.: Adaptive and intelligent Web-based educational systems. International Journal of Artificial Intelligence in Education 13(2-4), 159–172 (2003)
5. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American (May 17, 2001)
6. Henze, N., Krause, D.: Personalized access to web services in the semantic web. In: The 3rd Int. Semantic Web User Interaction Workshop, SWUI (2006)
7. Berners-Lee, T.: CWM, `http://www.w3.org/2000/10/swap/doc/cwm`
8. Muñoz Merino, P., Delgado Kloos, C.: A software player for providing hints in problem-based learning according to a new specification. Computer Applications in Engineering Education (Accepted January 12, 2008) (in Press, 2008)
9. XTutor Intelligent Tutoring System, `http://xtutor.org/`
10. Harskamp, E., Ding, E.: Structured collaboration versus individual learning in solving physics problems. International Journal of Science Education 28(14), 1669–1688 (2006)
11. Gertner, A., Conati, C., VanLehn, K.: Procedural Help in Andes: Generating Hints using a Bayesian Network Student Model. In: 15th National Conference on Artificial Intelligence, Madison, pp. 106–111 (1998)
12. Guzman, E., Conejo, R.: Self-assessment in a feasible, adaptive web-based testing system. IEEE Transactions on Education 48(4), 688–695 (2005)
13. Razzaq, L., Heffernan, N.: Scaffolding vs. hints in the Assistment System. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 635–644. Springer, Heidelberg (2006)

14. Riichiro, M., Bourdeau, J.: Using Ontological Engineering to Overcome Common AI-ED Problems. International Journal of Artificial Intelligence in Education 11, 107–121 (2000)
15. Lefebvre, B., Gauthier, G., Tadié, S., Duc, T., Achaba, H.: Competence ontology for domain knowledge dissemination and retrieval. Applied Artificial Intelligence 19(9), 845–859 (2005)
16. Zouaq, A., Nkambou, R., Frasson, C.: Building Domain Ontologies from Text for Educational Purposes. In: Duval, E., Klamma, R., Wolpers, M. (eds.) EC-TEL 2007. LNCS, vol. 4753, pp. 393–407. Springer, Heidelberg (2007)
17. Dicheva, D., Sosnovsky, S., Gavrilova, T., Brusilovsky, P.: Ontological Web Portal for Educational Ontologies. In: Workshop on Applications of Semantic Web in E-Learning at 12th International Conference on Artificial Intelligence in Education, Amsterdam (2005)
18. Aroyo, L., Pokraev, S., Brussee, R.: Preparing SCORM for the Semantic Web. In: Meersman, R., Tari, Z., Schmidt, D.C. (eds.) OTM 2003. LNCS, vol. 2888, pp. 621–628. Springer, Heidelberg (2003)
19. Psyche, V., Bourdeau, J., Nkambou, R., Mizoguchi, R.: Making Learning Design Standards Work with an Ontology of Educational Theories. In: 12th Artificial Intelligence in Education, Amsterdam, pp. 539–546 (2005)
20. Lama, M., Sánchez, E., Amorim, R.R., Vila, X.A.: Semantic Description of the IMS Learning Design Specification. In: Workshop on Applications of Semantic Web in E-Learning, Amsterdam, pp. 37–46 (2005)
21. Henze, N., Dolog, P., Nejdl, W.: Reasoning and Ontologies for Personalized E-Learning in the Semantic Web. Journal of Educational Technology and Society 7(4), 82–97 (2004)
22. Cheniti-Belcadhi, L., Henze, N., Braham, R.: An Assessment Framework for eLearning in the Semantic Web. In: 12th GI- Workshop on Adaptation and User Modeling in interactive Systems, Berlin, pp. 11–16 (2004)
23. Jovanović, J., Gašević, D., Brooks, C., Eap, T., Devedzic, V., Hatala, M., Richards, G.: Leveraging the Semantic Web for Providing Educational Feedback. In: 7th IEEE ICALT Conf., Niigata, pp. 551–555 (2007)
24. RDF/XML Syntax Specification, `http://www.w3.org/TR/rdf-syntax-grammar/`
25. Berners-Lee, T. (ed.): Notation3, `http://www.w3.org/DesignIssues/Notation3.html`

The Use of Ontologies to Structure and Support Interactions in LOR

Aude Dufresne[1], Mohamed Rouatbi[2], and Fethi Guerdelli[3]

[1] Department of Communication, University of Montreal,
C.P 6125 Succ Centre-Ville, Montréal, H3C 3J7
[2] Département d'Informatique et de Recherche Opérationnelle, University of Montreal
[3] Département d'Informatique, UQAM
Aude.Dufresne@umontreal.ca

Abstract. Actors using Learning Object Repositories are faced to a diversity of components created for specific situations. This research focused on creating a generic solution to integrate various applications and scenarios representation in the context of help to actors using LOR. The ODIS framework was developed and tested to interlink applications written in different programming languages, using shared OWL models: Explor@Graph, a Computer Supported Experimentation System, a scenario Editor and a Learning Object Repository. Models and instances are stored in a Sesame RDF database, offering many services such as import, export and query. Ontologies are used to standardize representations and models, so information can be exchanged and also to organize feedback and help in the environment. We will present how ODIS uses ontologies to transpose learning scenarios representations, to enrich them with user models and to define help.

Keywords: Ontologies, Help, Learning Object Repositories, Learning Environment, Authoring tools, eLearning, Collaborative system.

1 Supporting Actors in Learning Object Repositories

The present research was done in the context of the LORNET effort to integrate different learning object repositories and to develop an integration framework and a set of applications to exploit them in the context of learning. More specifically this research is concerned with defining help to actor using the LOR. Learning Objects can vary in granularity or types, ranging from a full course, to a definition, an image or a software program. Many repositories are being developed (ARIADNE, MERLOT, NIME, LORNET) and they pose interesting integration and compatibility problems, which can be solved using ontological representations. A simple integration solution was to develop standards for the metadata descriptions of resources (LOM) and for the definition of learning structures of activities (IMS-LD). If the level of difficulty and the structure of the domain are well defined, adaptive mechanisms can be defined to scaffold the access to resources as in adaptive hypermedia [1, 2, 3].

Our goal was to support not only access to resources, but also the unfolding of more specific descriptions of activities, not only for learners, but also for other actors

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 551–562, 2008.

using the LOR. The architecture to support activities in such a generic and distributed environment was not trivial, it was to offer advice and some control on the activities, in a generally flexible environment to suit the typically open tasks like exploring the content or building a course. It was important for the system to offer an open and light integration of components, in order to face the diversity and rapid evolution of eLearning resources. We also wanted to organize it around formal representations, which could be easily changed and made to follow standards being developed in the field for the exchange of resources or the description of learning scenarios (IMS-LD).

We will present here two problems which are addressed by this research: first on how the help can be given inside different objects and applications, running simultaneously with different implementations; second, we needed to find a way to maintain the models which are at the root of help, models of the domain, of the users, of the tasks, of the context and applications. The solutions to both problems are linked since they rely on formal specifications of communication processes among applications as web services or otherwise, and on the maintenance of the models and instances described using those models.

A framework was developed to sepecify how the different contexts and application sshould be alligned, using OWL representations and a Sesame database to exchange structures and instances of specific ontologies. Using these generic representations, help can be defined to link user overlay models, with contextual navigation activities. We will present how the solution was developed using different generic tools, through the Ontology Data Integration System (ODIS).

2 Support in Learning Environments

Learning environments have explored different level of technological support to activities, ranging from the generic but passive Learning Management System like WebCT or Sakay, to more specialized and interactive environments like simulations, knowledge based system and Computer Supported laboratory, to less technical but more conceptual models or methodologies to structure interactions. Other researches try to unleash the full computational power into supporting further interactivity for learning, adding intelligent mechanisms to search for resources, to follow learner progression and to adapt his environment, like in adaptive hypermedia [2, 4].

This research is at the frontier of these different approaches, and rely on a collaborative system, where both structured interaction controlled by the system is possible, as well as flexible navigation and open task from the user. The Explor@Graph system [5, 6] was designed to let a professor easily structure learning environments inspired by the MOT representation [17], gathering resources from LOR and organizing them inside conceptual graphs linking activities, resources and concepts. The teacher can use the environment to organize the exploration of content and learning evaluations. The system leaves flexibility for students to be active, using intelligent mechanisms to support their interaction providing retroaction on progression and linking discussion with content [7].

We saw the opportunity to use ontologies to define support at a higher level using more generic knowledge structures underlying the learning activity as suggested by [8], for example to find unfinished prerequisites to a task. Further more, the help given to the activity was limited to the Explor@Graph environment and had to be extended in the context of the integration of different learning resources and learning management systems.

3 The Use of Ontologies to Improve Learning Environments

As an AI technique, ontologies[1] have raised great interest as a technology to support learning, first because they appeared promising to improve searching in resources, by adding an intelligent layer based on semantic representations of the domain, the activities, or the pedagogical models [9, 10].

The first interest for ontologies was for their potential to improve searching [11] in what would be called the semantic web. The integration of semantic representations makes it possible to search using classes, properties, relations and synonymous.

Aside from their potential for searching, ontologies have been used to support visualization and access, for example Topic Maps is a standard (ISO/IEC 13250 [12]) which was developed to describe the organization of content and that can be used to exchange and organize the visualization of resources. In specific domains, topic maps can be constructed from expert knowledge of the domain, or they may be directly extracted from a corpus of textual resources and even from multimedia resources using speech to text. The semantically organized structure of contents can be used to display and browse information for learning , in an organized and flexible way [13].

Once informations are organized in a semantic structure, adaptive hypermedia techniques can be used to define intelligent access [14], choosing elements to display depending on the context and the progression or preferences of learner. In the same direction, ontologies can be used for the personalization of learning environments and their adaptation to cultural or accessibility differences.

Another use of ontologies is to help define standards and strategies in the domain of Learning. Standards like the LOM, SCORM, but also of models of pedagogical activities IMS-LD and strategies are being developed to ensure that learning resources can be interconnected and shared in the community and across learning applications.

Finally ontologies can be used as annotation tools. Tools like MagPie [15] give user powerful tool to organize resources as they read them and to exchange them with peers as in the FOAF system or in general in community of practices. The power of ontologies can be used to structure annotations and visualize content to which they are attached. Furthermore, the modeling of content and user's navigation in parallel can support peer helping.

4 Integration of Applications and Help Inside Explor@Graph

The Explor@Graph interface was designed to be a unifying interface to represent conceptual structures of activities, concepts and resources, but also to facilitate the definition of support on those elements [16]. Each graph or node may have static and dynamic metadata associated, which is used to highlight visually the structure of information - completion, ordering. Following the MISA[2] methodology and Explor@ models [17], the

[1] The term ontology has been used as a generic term to cover different abstract representation of semantic structures, which sometimes are limited taxonomies of the domain concepts, sometimes called topic maps. Sometimes they include specifications of objects, classes and their properties (RDF), sometimes they include restrictions on properties (OWL) or logic rules to extract new knowledge (OWL-DL). In the context of this paper, we use ontology as a generic term, though we will specify its level of description.

[2] MISA is a design tool for learning systems, the web tool supporting this methodology is named ADISA, developed by LICEF.

Explora@Graph eLearning environment organizes navigation around the conceptual structures of concepts (rational models), tasks (functional models) and documents (media model) using typed relations like composition, precedence, resources and production). The conceptual structures are annotated by the user himself who may change his user's model. These annotations serve as feedback and can be used as overlay models by the help system to adapt advices and adaptive help in the interface - open or highlight elements, propagation of user models using the structures of relations. Using these information, rules can easily be defined using a WISIWIG interface inside the Explor@Graph Editor, by selecting components, to verify conditions or events in the environment and to trigger actions on components (highlight or change a node or user model property) or to present help messages.

During interaction with the Explor@Graph Navigator, helping rules are stored with the structure of the course and the user models. These rules will be sent to the Generic Advisor that will follow the user during different learning activities giving him advices and updating the user and the group models. This follow up is done using spying modules attached to each application that sends every action done by the user and notifys the expert system. After receiving these events depending on the set of rules and other facts collected about the user, the expert system can trigger two different types of actions: internal actions, which modify users and group models won't have noticeable effect; the second type, external actions, could be a dialog box, an animated Microsoft Agent or a change in the environment as highlighting a text, opening a node or opening a menu, selecting a button, etc. For that Control Units are used. generated by parsing the user interface. Most of the control units use type discovery and the reflection APIs to manipulate the user interface.

Though, the environment can be very powerful, in the context of the LOR, we needed to explore integration among other applications or resources and display help in relation to them. Thus we decided to develop a data integration framework for communication between the Explor@Graph activity description environment and other applications to describe and save the structures, their instances and also dynamic information as they are updated during the activity.

5 Integration of Applications and Help Using XSD Specifications

In order to define the support to activities inside the LOR, we have used different applications, in which integration and help can be defined. Generally, part of the structure is defined through the various applications of the LOR, but to define support we present a framework where other help structures can be defined [18].

Figure 1 presents the general framework:

- The LOR where resources are saved, indexed and grouped.
- The User and Context Manager where are defined users, groups, roles in relation to tasks, and where resources and privileges are linked to a context.
- The Explor@Graph system (Editor and Navigator) where are defined graphical navigation structures, relating elements of tasks, concepts or documents, and where support rules can be defined using the properties of the represented structures as conditions and offering actions like MsAgent messages, or control functions on elements (show properties, change them, execute methods of objects, etc.).

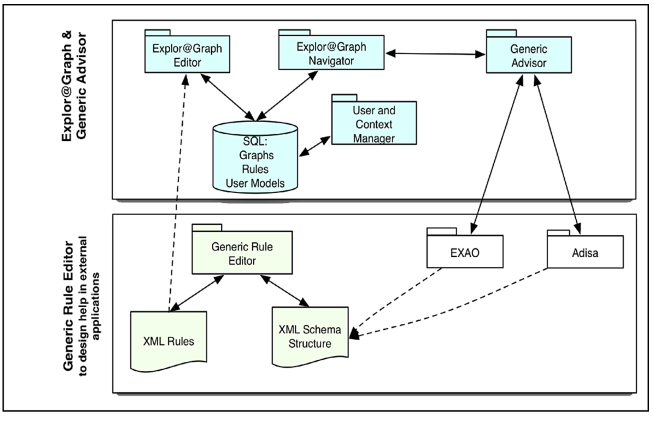


Fig. 1. Integration and support across applications using XML communication with the Generic Advisor, XSD models of external applications and a Generic Rule Editor

- In the first version [18], to generalize support, other structures were defined in XML Schema to represent other applications and support components.

- The generic Rule editor is used to create conditions and actions using those XML elements to be used in support rules .
- The Generic Advisor is an Expert System with a communication interface that receives events from applications, search for adequate rules and send selected actions back to applications, to be executed or to update internal models.

The system was used to define contextual help inside a computer supported laboratory [18-20]. In this first implementation of generic support, the Explor@Graph system is a fully integrated and usable system to define structures and support [21] but, the integration with other applications is minimal. It proceeds through manual description of XSD structures and XML components of external applications interfaces, which are read in the Generic Rule Editor, and where conditions and actions can be defined. Elements of rules may be transferred from the Generic Rule Editor to Explor@Graph, as external actions or conditions, using copy/paste. Thus elements of representations of concepts, scenarios and external environments can be mixed in rules:

For example a rule may be defined stating that,

```
if (EG openedGraph = ExperienceOxygen)
   and (Plugin sensor = temperature)
then MsAgent Prof speaks "Good you are using the tem-
perature sensor. In fact, colder water contains more
oxygen, let's see if you can measure that.."
Action : Open EXAO form, highlight fields or tools to
be used with values for the specific task.
```

But in this implementation [22], the instances of the structure of external applications were manually defined. Also the XML representations were limited to represent object relations among classes and did not use ontological restrictions or deduction. We wanted to be able to persist contextual and user models associated with those external applications. Finally, we wanted to generalize the navigation structures, to make them compatible with other LMS and especially compatible to description standards like IMS-LD, in the perspective of the integration of LORs.

6 ODIS - Ontological Data Integration System

In the new ODIS framework [23] (Figure 2) different OWL ontologies are developed using PROTEGE to describe applications and conceptual models. A composite ontology is developed that describes the integration, alignment and augmentation.

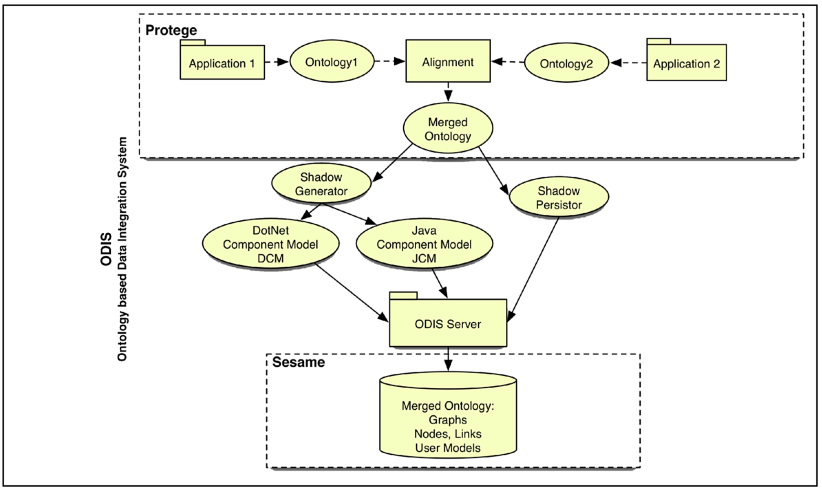


Fig. 2. ODIS framework- **Integration** of OWL ontologies using Protege and the generation of components

This merged ontology serves as an input to the ShadowGenerator which will generate the different software components DCMs (DotNet Component Model) and JCMs (Java Component Model).

The main functionality of these software components is to allow any application involved in this integration to persist, update and delete data using its own format-ontology- without worrying about the mapping, the communication layers and the data storage repositories (Figure 3). In fact, the DCMs and/or the JCMs use the services provided by ODIS server which relays on the ShadowPersistor software component and Sesame APIs to query, update and save these models. This automatic code generation accelerates the development time related to data integration considerably allowing the developers and system designers to focus on what counts the most. However, this method involves defining manually how the ontologies are to be merged, which is then done automatically by the applications.

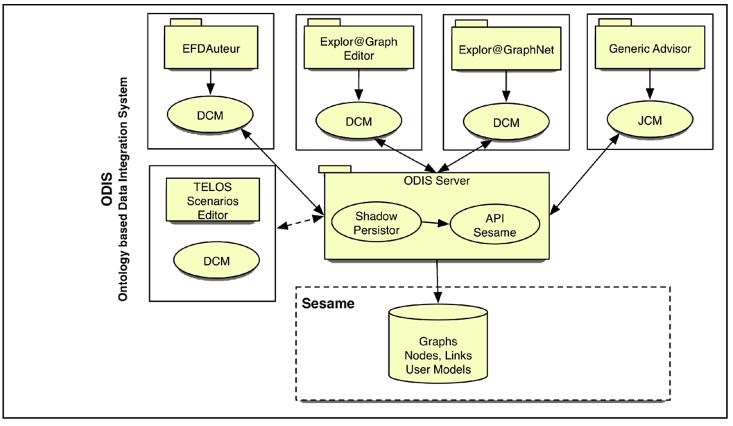


Fig. 3. ODIS framework : **Interaction** across different applications using the generated components communicating with a Sesame RDF database

Explor@GraphNet[3] is used to display structures which are imported from the Sesame database. Thus the functions to help define help inside Explor@Graph will be used with the structures coming from other LMS. In interaction, different applications use the ODIS server to access and communicate structures. Generic structures have been defined to query the ontology according to its graph structure. For example:

```
find all the "activity" nodes, which are target of a
"prerequisite" link to an "activity"
find one node "resource"  which "is not completed" and
linked as "resource" to a "concept" node.
```

Elements between quotation marks can be any predicate associated to a node, a link or a graph, and the ontology is used to infer relations and properties of objects and their class. For example:

```
If A prerequisite to B and B prerequisite to C than A
is prerequisite to C.
```

Queries use the classes and predicates of the ontology in the RDF database, for example *to select all nodes with a prerequisite relation to a node:*

Select DISTINCT X from
{Edge} ns10:Target_node_uid_eg {"2068"^^xsd:long},
{Edge} ns10:Src_node_uid_eg {X},
{Edge} ns10:EdgeType {ns10:Prerequis},
{Node} ns10:NodeID {X}
using namespace
owl = <http://www.w3.org/2002/07/owl#>,
ns10= <http://www.owl-ontologies.com/unnamed.owl#>,
xsd = <http://www.w3.org/2001/XMLSchema#>

[3] Is a more generic version of Explor@Graph navigator, developed in VB.Net.

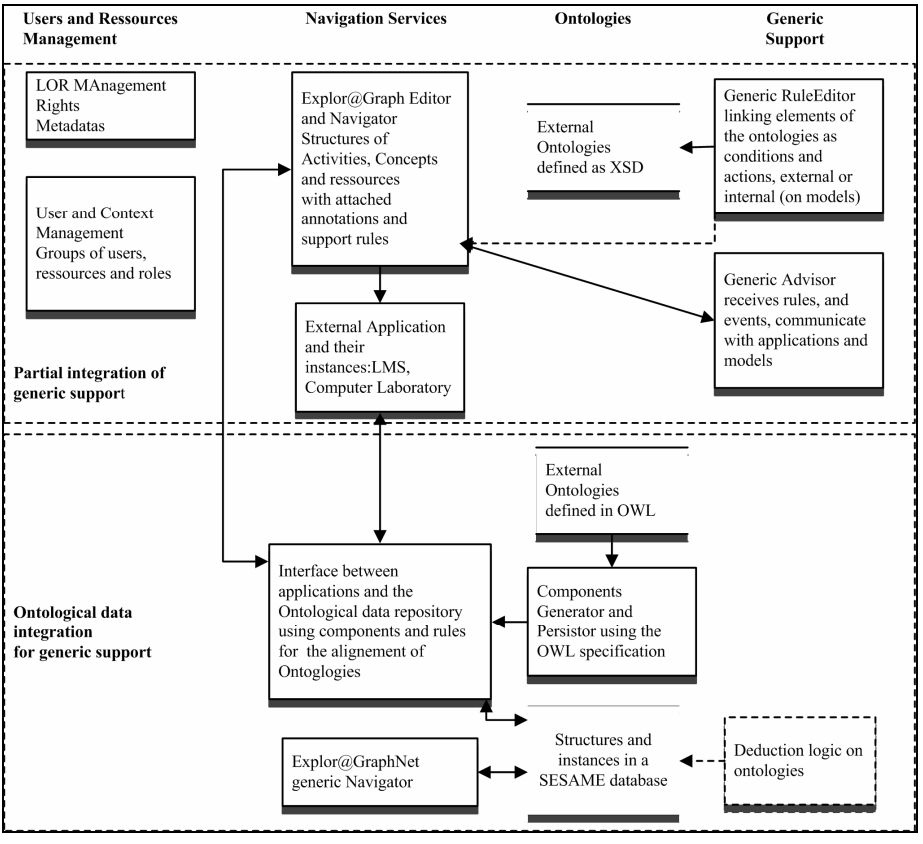


Fig. 4. Generic Framework of the support to functions, with the different levels of structures

Figure 4 presents a summary of the research we presented and explain how the new ODIS framework can be used in combination with the previous Explora@Graph and Generic Rule Editor framework to support the definition of generic help across applications. Using generic navigation services, ontologies and generic applications to define and display help.

7 Application of ODIS with the Doctoral Training Environment

The new framework was used to connect the Doctoral Training Environment [24] with the Explor@GraphNet graphic interface. The DTE Authoring Tool allows users to design and generate Doctoral Training Environments (DTE), hence it is named DTE-Author. It was built upon CONCEPT@, Télé-université's authoring environment, in connection with PALOMA, its Learning Objects Repository. Structures of activities and actions are described inside the DTE and are then exported in the Sesame database.

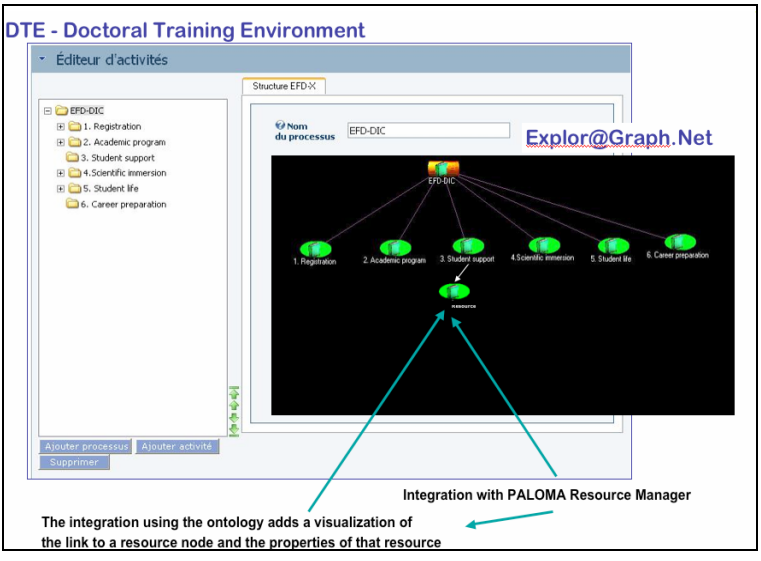


Fig. 5. Interconnection of the Explor@GraphNet graphical interface with the Doctoral Learning Environment using ontologies

Using the ODIS framework, the ontology of the DTE was aligned to the Explor@GraphNet ontology, for example, the alignment adds the visual properties to the elements; resources becomes independent nodes with a specific icon, a 'resource' link is added between resource node and the activity node, overlay user models are added to task elements imported from the DTE, etc. Since the user models are now properties associated with nodes, they can be persisted using existing functions of Explor@Graph. The structure can then be viewed in the EGN Graphical Interface.

The same generic framework is used to connect the scenario editor developed in LORNET and the Explor@Graph system to the Sesame database, so structures can be imported and exported from it. The Explor@Graph infrastructure to develop and display help functions can then be used on those structures developed in other LMS to integrate support in reaction to navigation inside different environments. Queries on ontologies can be used as conditions in the help system or to highlight elements of the structures, resource or task nodes using ontological inferences.

8 Related Work

As we saw, ontologies have been used to links structure of the domain and resources as in adaptive hypermedia. This has been enriched by the integration of models of the user knowledge and models of peer knowledge and preferences for peer helping. In that direction Brooks & al. MUMS system developed a user model ontology to organize and store user models across applications and to support peer helping in forums, linking content of discussion and preferences of users for a helpee or helper.

Others [25] have tried to develop systems to link learning design and learning object content. They adapted the IMS-LD ontology to link it to resources. Learning context is represented by specific competencies, objectives and evaluations and is linked with the ontology of Learning resources Alo-COM (structure ad content type) in the LOCO-Cite ontology. The purpose of this ontology is to encourage repurposing of resources. But the system does not offer support to interactions in complex resources software applications like the EXAO system.

Jovanović [26] presented the MICE framework that uses an ontology to define context in the use of Learning Objects and to give feedback to users on their interaction with the system. Different applications are connected using this framework - BlueJ a Java Learning environment, G-Study and environment to track the use of resources and I-Help a discussion forum. As in our system, they propose a generic system, that track events in the environments and use the ontology to give feedback on the match between interaction and theoretically defined learner models of the domain, using a pedagogical model the SRL (Self Regulated Learning) model. The ontology they propose, links different ontologies on interaction, domain knowledge associated with user models and programming style models). It uses a Jess inference engine to trigger feedback as styles are recognized. In practice, if the framework is general, the ontology is directly linked to the programming and chatting environment for which it was designed. The rules are defined to offer feedback in relation to activity models not to support adaptation of the environment as in our case.

9 Discussion and Conclusions

The framework that we presented opens many interesting possibilities. For one, Explor@Graph offers a generic interface for designing support system, can be used with its user modeling and support's rules possibilities. It is now open for importing external structures, and can be run simultaneously to follow activity in external applications, using the Sesame RDF database for exchanging information. On the other end, the OWL structures which are developed in PROTEGE are easy to visualize and modify. Components (DCM and JCM) which are exported directly from that structure facilitate the development of communication interfaces between applications and the RDF Sesame repository. The RDF structure can be used for searching or for ontological logic reasoning on the information stored and thus augment power of the Generic Support system. We will use the Sesame database to maintain and share information on models of users and dynamic contexts of execution between the applications (more than just overlay models of nodes properties).

The framework we have presented is a good step in providing support to activities inside a modular, evolving and opened context such as a network of LOR. In fact, the set of applications we have developed have been used to define support in various context: support in applying the MISA methodology inside the Adisa system [27]; support to professors designing courses based on pedagogical strategies inside the ExploraGraph Editor [28]; pedagogical support to students using a Computer Assisted Lab [20] and support to the Doctoral Training Environment [24].

Future research will investigate the deduction possibilities using constraints in ontologies or relations among elements in tasks and learner models following the principles of Learning Design ontological models [8, 10].

References

1. De La Passardière, B., Dufresne, A.: Adaptative Navigational Tools for Educational Hypermedia. Computer Assisted Learning, Berlin (1992)
2. Brusilovsky, P.: Methods and Techniques of Adaptive Hypermedia. User Modeling and User-Adapted Interaction 6, 87–129 (1996)
3. Cristea, A., Aroyo, L.: Adaptive Authoring of Adaptive Educational Hypermedia. In: De Bra, P., Brusilovsky, P., Conejo, R. (eds.) AH 2002. LNCS, vol. 2347, pp. 122–132. Springer, Heidelberg (2002)
4. Dufresne, A.: From adaptable to adaptive interface for distance education. In: Workshop on Intelligent Educational Systems on the World Wide Web, Artificial Intelligence in Education: Knowledge and Media in Learning Systems, Kobe, Japan (1997)
5. Dufresne, A., Paquette, G.: ExploraGraph: A flexible and adaptive interface to support distance learning. In: EdMEdia 2000, Montreal (2000)
6. Dufresne, A.: Conception d'une interface adaptée aux activités de l'éducation à distance – ExploraGraph. Sciences et Techniques Éducatives 8, 301–320 (2001)
7. Dufresne, A.: Explor@Graph Scenarios Editor – Designing a Collaborative Task. In: ICALT 2006, Kergrade, The Netherlands (2006)
8. Paquette, G., Tchounikine, P.: Contribution à l'ingénierie des systèmes conseillers: une approche méthodologique fondée sur l'analyse du modèle de la tâche. Sciences et Techniques Éducatives 9 (2002)
9. Bourdeau, J., Mizoguchi, R., Psyché, V., Nkambou, R.: Selecting Theories in an Ontology-Based ITS Authoring Environment. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 150–161. Springer, Heidelberg (2004)
10. Mizoguchi, R., Bourdeau, J.: Using Ontological Engineering to Overcome AI-ED Problems. International Journal of Artificial Intelligence in Education 11, 107–121 (2000)
11. Abecker, A., Van Elst, L.: Ontologies for Knowledge Management. In: Staab, S., Studer, R. (eds.) Handbook on ontologies International handbooks on information systems, Berlin, pp. 435–454. Springer, New York (2004)
12. ISO/IEC 13250:2000 Topic Maps: Information Technology (September 2007)
13. Dicheva, D., Dichev, C.: Creating and Browsing Educational Topic Maps. British Journal of Educational Technology - BJET 37, 391–404 (2006)
14. Bra, P.D., Calvi, L.: AHA! An open adaptive hypermedia architecture. The New Review of Hypermedia and Multimedia 4, 115–139 (1998)
15. Domingue, H., Dzbor, M., Motta, E.: Semantic Layering with Magpie, ISCW-SWEL 2004, Hiroshima, Japon (2004)
16. Dufresne, A.: Modèles et outils pour définir le soutien dans les environnements hypermédias d'apprentissage. In: Vries, E.d., Perning, J.-P., Peyrin, J.P. (eds.) Hypermédias et Apprentissage, Grenoble, pp. 13–24 (2001)
17. Paquette, G., De La Teja, I., Dufresne, A.: Explora: An open virtual campus. In: EdMEdia 2000, Montreal (2000)
18. Dufresne, A., Guerdelli, F., Rouatbi, M.: A Generic Framework to Support Users in Various E-Learning Applications. In: EDMedia 2005, Montréal, Canada (2005)

19. Dufresne, A., Rouatbi, M., Nonnon, P., Touma, G., Guerdelli, F.: Soutien à l'Apprentissage en Sciences une Solution Générique – l'Exemple de l'EXAO. In: Colloque DIVA 2005, Montréal (2005)
20. Dufresne, A., Rouatbi, M., Villiot-Leclercq, E., Guerdelli, F.: Chapitre 3 Architecture Intégrée pour l'Accès et le Soutien à l'Utilisation des Ressources pour l'Apprentissage. In: Pierre, S. (ed.) Développement, intégration et évaluation des technologies de formation et d'apprentissage, pp. 63–88. Presses Internationales Polytechnique, Montreal (2005)
21. Dufresne, A., Schlienger, C.: The ExploraGraph advising system: an ergonomical evaluation of the editor. In: TICE 2002. Lyon, France, pp. 299–306 (2002)
22. Dufresne, A., Rouatbi, M.: Adaptive Support to e-Learning Using Formal Specifications of Applications, Tasks, Content and User Models. In: SW-EL 2004 Applications of Semantic Web Technologies for E-Learning, Hiroshima, Japan (2004)
23. Dufresne, A., Rouatbi, M.: Ontologies, Applications Integration and Support to Users in LOR. In: SWEL@AIED 2007 Workshop on Ontologies and Semantic Web for Learning, Los Angeles (2007)
24. Bourdeau, J., Henri, F., Dufresne, A., Tchetagni, J., Ben Ali, R.: Collaborative Learning and Research Training: Towards a Doctoral Training Environment. In: Les Cahiers Leibniz: Laboratoire Leibniz-IMAG, Grenoble, France, vol. 157, pp. 38–47 (2007)
25. Knight, C., Gašević, D., Richards, G.: Ontologies to integrate learning deséign and learning content. Journal of Interactive Media in Education 07 (2005)
26. Jovanović, J., Knight, C., Gašević, D., Richards, G.: Learning Object Context on the Semantic Web. In: Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies - ICALT 2006, Kerkrade, The Netherlands, pp. 696–673 (July 2006)
27. Dufresne, A., Basque, J., Paquette, G., Leonard, M., Lundgren, K., Prom Tep, S.: Vers un modèle générique d'assistance aux acteurs du téléapprentissage. Sciences et Techniques Éducatives 10, 57–88 (2003)
28. Villiot-Leclerc, E., Dufresne, A.: Supporting the Design of Socio Constructivist Scenarios with ExploraGraph. In: EDMedia 2005, Montréal, Canada (2005)

Leveraging the Social Semantic Web in Intelligent Tutoring Systems

Jelena Jovanović[1], Carlo Torniai[2], Dragan Gašević[3], Scott Bateman[4], and Marek Hatala[2]

[1] University of Belgrade, Serbia
[2] Simon Fraser University, Canada
[3] Athabasca University, Canada
[4] University of Saskatchewan, Canada
jeljov@gmail.com, carlo_torniai@sfu.ca, dgasevic@acm.org, scott.bateman@usask.ca, mhatala@sfu.ca

Abstract. Today's technology enhanced learning practices cater to students and teachers who use many different learning tools and environments and are used to a paradigm of interaction derived from open, ubiquitous, and socially-oriented services. In this context, a crucial issue for education systems in general, and for ITSs in particular, is related to the ability of leveraging these new paradigms for creating, maintaining and sharing the knowledge that these systems embed. This will enable learning environments to benefit from shared information from disparate systems, which is related to learning content and student activities, so that the overall complexity of system development and maintenance would be reduced while at the same time improving the capability of personalization, context-awareness, collaboration, and feedback provisioning. In this paper, we investigate how the Social Semantic Web can be leveraged for enabling and easing this process. This paper analyzes each ITS module, showing how it can benefit from the Social Semantic Web paradigm.

Keywords: Intelligent Tutoring Systems, Semantic Web, Social Web, Ontologies, Folksonomies, E-learning.

1 Introduction

Intelligent Tutoring Systems (ITSs) are computer-based instructional tools that rely on artificial intelligence (AI) techniques to generate individualized interactions tailored to a student's learning needs. They have emerged from AI at the very time that the field was struggling to move beyond the goal of equaling human intelligence by creating machines that could "think" like humans [1]. ITSs have demonstrated significant results in the instruction of specific domains (e.g., algebra). However, today's technology enhanced learning practices indicate that the education process of each particular learner is not limited to only one learning tool or environment (e.g., ITS), but instead students are using many different learning tools and services (e.g., learning management systems or discussion forums). It is natural to assume that, for example, some knowledge and skills that students gain in the other learning environments

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 563–572, 2008.

would be beneficial to bootstrap a student's model in an ITS. Moreover, the reuse of some of the best pedagogical practices across different learning environments or learning content may also be useful for ITS developers.

Equally important is the fact that learning is not an isolated process, and it happens in parallel with the regular day-to-day activities, which students and educators have. Students and educators now live in the world of Facebook, Wikipedia, YouTube, del.icio.us, and SecondLife. Most of these technologies are collected in a common framework, of the so-called Social Web, where the notions of social interaction, human computing, and collective intelligence are major assets. Today's students have spent most of their lives surrounded by, and using, computers, videogames, digital music players, video cameras, cell phones, and the Web itself. We envision that the learners activities performed using these ubiquitous technologies should be leveraged in ITSs (e.g., for creating more informed student models, or to increase the collaborative features of ITSs).

Having the above examples in mind, the main question is: How might ITSs be integrated with the aforementioned technologies? A potential approach could be to leverage learning technology standards such as the IEEE Learning Object Metadata and the IMS Learning Design definitions. However, those standards only provide a syntactical layer for the integration of different learning tools. What they do not provide are effective and reliable mechanisms for managing (i.e., capturing, representing, and evolving) various types of knowledge (i.e., domain, user, and pedagogical) which should be shared among various environments and ITSs. To enable such knowledge sharing, we present important conditions that should be satisfied by knowledge sharing solutions, to meet the requirements of ITSs:

- a formal representation of the knowledge being shared, so that its semantics are fully preserved;
- to be a low-cost, or less expensive solution than the current mechanisms commonly used for knowledge capturing and maintenance in ITSs;
- to be in compliance with current Web and learning technology standards.

In this paper, we analyze how the Social Semantic Web can be used as a means to satisfy the above conditions. We propose a synergetic space built on top of formal ontology-based representations of shared knowledge and social computing mechanisms (e.g., folksonomies). We first introduce the fundamental concepts of the Social Semantic Web and then proceed to examine its benefits for each particular component of the ITS architecture.

2 The Social Semantic Web

The Semantic Web has been introduced as the evolution of the current Web in which "information is given well-defined meaning, better enabling computers and people to work in cooperation" [2]. The building blocks of the Semantic Web are *ontologies*. Ontologies are formally described conceptualizations of shared domain knowledge. They can be combined, shared, extended and used to semantically annotate different kinds of resources, such as web pages, documents, and multimedia content, to name a

few. By leveraging ontological infrastructure, one can build various *intelligent* services that are capable of *i)* inferring new knowledge based on relationships specified in ontologies, and *ii)* correlating pieces of content according to their semantic annotations, and thus interpreting meanings with respect to the underlying ontologies. Despite having many promising aspects, the Semantic Web is still not widely adopted yet. This is mainly due to the difficulties in ontology creation and maintenance, and the process of semantic annotation.

A new wave of so-called *social* applications has emerged as a culmination of technology and interaction techniques, and has been labeled the Social Web or Web 2.0 [3]. The Social Web transforms the 'old' model of the Web as a container of information accessed passively by users - into a platform for social and collaborative exchange in which users meet, collaborate, interact and most importantly create content and share knowledge. Popular social websites, such as Facebook, Flickr and YouTube, enable people to keep in touch with friends and share content. Other services such as blogs and wikis allow novice users to easily create, publish and share their own content. Further, users are able to easily annotate and share Web resources using social bookmarking and tagging; thus creating metadata for web content commonly referred to as "folksonomies". However, Social Web technologies in general, and collaborative tagging in particular, suffer from the problems of ambiguous meanings. For instance, collaborative tags are often ambiguous due to their lack of semantics (e.g., synonymous meanings for a tag). Moreover, they lack a coherent categorization scheme, and require significant time and a sizeable community to be used effectively [4].

Despite the initial perception that the Social Web and the Semantic Web oppose each other, the two efforts are jointly being used to create a common space of semantic technologies. In fact, the Semantic Web can not work alone. It requires society-scale applications (e.g., advanced collaborative applications that make use of shared data and annotations) [5]. The paradigm of knowledge creation derived from the Social Web can be effectively used to refine/update ontologies generated according to Semantic Web standards and best-practices. At the same time the Social Web can benefit from the paradigm of structured knowledge, represented with standard languages adopted in the Semantic Web vision. Such standards will facilitate the sharing and application interoperation through the collectively created knowledge.

The idea of merging the best of both worlds has converged in the concept of the Social Semantic Web, in which socially created and shared knowledge on the Web lead to the creation of explicit and semantically rich knowledge representations. The Social Semantic Web can be seen as a Web of collective knowledge systems, which are able to provide useful information that is based on human contributions, and which improves as more people participate [6]. Table 1 summarizes the key features of the Semantic, the Social, and the Social Semantic Web.

Specific examples of the Social Semantic Web are being undertaken in a wide number of projects. For instance, DBpedia is a large-scale semantic knowledge base which structures socially created knowledge of the Wikipedia, a wiki-based encyclopedia. DBpedia takes advantage of the common patterns and templates used by Wikipedia authors to gather structured information into a knowledge base of socially created structured knowledge. The result is a huge database of shared knowledge which allows 'intelligent' queries such as: "List the 19th century poets from England" [7].

Table 1. Comparison of the key features of the Semantic, the Social and the Social Semantic Web

Semantic Web	Social Web	Social Semantic Web
Structured Knowledge	Semi-structured/ unstructured knowledge	Structured or Semi-structured knowledge
Standardized and machine understandable knowledge representation and annotations	Knowledge and annotations not expressed in standard forms	Standardized and machine understandable knowledge representation and annotations
Knowledge creation process requires engagement of experts for knowledge provision and formal modeling: expensive	Knowledge creation process based on social activities and informal knowledge modeling: inexpensive	Simplified creation/refining of formalized knowledge based on social software sources
Semantic annotation (annotation with ontological concepts): expensive	Annotation based on social tagging, knowledge agreement in wikis: inexpensive	Annotation based on inexpensive social sources of knowledge creation

The implications of such technologies are significant for the educational domain, where students can find immediate answers to their detailed questions. Further than finding answers to questions, though, is the possibility of a web of pedagogically focused learning materials; where activities are easily created, shared, and used by students and teachers, without the need for detailed knowledge engineering skills or know-how of advanced technologies. Specifically, for the field of ITS, we see several immediate benefits of incorporating the exisiting capabilities of the Social Semantic Web.

3 Bringing Social Semantic Web to ITS Systems

The Semantic Web and ITSs have a lot in common: besides common roots in AI, both are based on well defined principles, and have well defined structures. However, the Semantic Web is designed to enable (among other things) integration and interoperability on a web-wide scale between applications. ITSs, in contrast, are typically designed as closed systems. Semantic Web technologies can lead to the opening of ITSs, by allowing the sharing of learner interactions and of knowledge embedded in their modules. While some work has investigated the promise that Semantic Web technologies offer for 'opening' ITSs (e.g. [8]), the trend of closed systems continues. A common point of note between the Semantic Web and ITSs is that neither of them are widely adopted by the Web community, due to their complexity for laymen. The Social Web paradigm is enabling a gradual acceptance of the Semantic Web among layman users - hopefully leading to its wider adoption. In the same way, we see the Social Web paradigm as potentially promoting broader acceptance of ITSs.

There are several challenges in the ITS field which can either fully or partially be addressed with technologies offered by the Social Semantic Web. In the rest of the section, we look at each module of ITS systems and explain the benefits that can stem from the inclusion of the Social Semantic Web paradigm.

3.1 Benefits for the Domain Module

The main problem with the development and maintenance of the domain modules of traditional ITS systems stems from the fact that despite the significant effort put into the definition of domain knowledge models, these models cannot be reused or shared between systems. Moreover, the process of their definition or evolution, even within the same ITS, can be accomplished only by domain experts. This problem can be addresses by expressing such models in a standard and interoperable format, which would enable the reuse of these models as well as facilitate sharing and combining models from different ITSs focused on the same domain; thus easing the process of model evolution.

The Semantic Web offers technologies to address these needs through the Resource Description Framework (RDF) as an extensible and reusable data representation format. In addition, it provides means for formally specifying the semantics of the represented data (i.e., defining ontologies as sharable and machine-processable knowledge representation formats).

For more than two decades, researchers in the ITS field have been on the road of using ontologies for domain knowledge modeling and representation [9], [10]. However, the problem is that these first endeavours were restricted to local ontologies that were usable only in systems for which they were developed, and not used for knowledge sharing among ITSs covering the same or similar knowledge areas. Therefore, the problem of enabling (semi-)automatic knowledge sharing, reuse, and exchange among several different tutors covering the same or similar domain, is still open. M-OBLIGE [11] was the first ontology-based knowledge model focused on enabling the interoperability of ITSs. However, at the time when this model was proposed (in 2002), the Semantic Web infrastructure was not mature enough to provide the required support. Since the proposal of M-OBLIGE, Semantic Web technology has made significant progress, and the next generation of Semantic Web applications [12] can now take advantage of the vast amount of semantic data and ontologies, which are available online. For instance, there are now infrastructures (such as Watson[1]) for indexing semantic data and ontologies on the Web.

Another problem related to the usage of ontologies for representing domain models is the constant need for ontology evolution (maintenance and updating). This is not a trivial task, because current approaches and tools assume a background in knowledge engineering, or familiarity with ontology languages; this is true even when a (semi-)automatic approach is proposed. In general, tools are too complex to be used by most teachers. The social side of the Social Semantic Web paradigm offers a possibility to simplify the evolution process of ontology-based domain models. Student activities, which can be enabled within ITSs, such as annotating, tagging, and the like, can be leveraged for the maintenance of a domain model. Further, intrinsic motivation and trust of students in a system that derives knowledge from their activities is certain to increase, since they are aware that they are contributing to the system and that their contribution counts.

In our latest work, we have suggested a novel method of interactive visualizations that provide an intuitive and practical way for instructors to incorporate the implicit

[1] http://watson.kmi.open.ac.uk

feedback available from student folksonomies for evolving domain ontologies [13]. In addition, we leverage algorithms for computing the semantic relatedness to further facilitate the teachers' task of ontology maintenance. In this way, we combine several approaches to leverage student contribution for providing support in ontology evolvement. Such tasks of ontology refinement are constant, and our method allows support to be given which is consistent with the course content, and with the conceptualizations that instructors and students have of that content.

3.2 The Benefits for the Student Module

The use, within the Student Module, of interoperable representation of student models would enable ITS systems to construct and update their student models using information coming from all of the different systems with which a student interacts. This would lead to an increased capability and consistency of personalization in ITSs.

A lot of research effort has already been put into development of interoperable ontology-based user models that can be shared among different systems. For example, Dolog et al. [14] have proposed an ontology-based framework for manipulating and maintaining sharable learner profiles. Niederee et al. [15] have introduced a metaphor of a 'context passport' that accompanies users on their travel through the information space. When interacting with a system, the relevant "context-of-use" is extracted from this context passport and is used for improved support of related activities. There is also a proposal for an XML-based user model exchange language, called UserML, aimed at enabling decentralized systems to communicate through their user models [16]. The same research group has developed GUMO – the General User Model Ontology - which allows for the uniform interpretation of decentralized user models.

The Friend of a Friend (FOAF) ontology[2], an ontology of social networks, provides a unified way for describing people and their relations. Due to its popularity and wide acceptance among Web users and communities, this ontology has become the basis for building domain/application specific ontologies for user and group modeling. For example, Ounnas et al. [17] has recently proposed a semantic learner model based on the FOAF ontology and aimed at supporting automation of the process of grouping students while preserving the individual's personal needs and interests. They have actually extended the FOAF ontology with a set of student-properties, which are relevant for the formation of different types of learning groups. Since it presents a common part of many of the application-specific user models commonly used, the FOAF ontology serves as a good base for sharing user models among diverse learning systems. In the context of ITSs, this offers the potential to allow learners to seek peer-support while studying certain topics and by leveraging successful learning paths and/or knowledge of friends. Extending these functionalities even further is possible. For instance, a student would be able to export information about their achievements in an ITS into their e-portfolio. In this way, a user-centered profile management can be enabled allowing students to benefit from personalization, feedback provisioning and interaction while moving across different learning systems.

[2] http://xmlns.com/foaf/spec/

3.3 The Benefits for the Teaching Module

Design of teaching strategies and their representation in a computer executable form is a challenging task, and requires the engagement of both pedagogical experts (having knowledge of instructional and learning theories as well as best teaching practices) and knowledge engineers (capable of representing pedagogical knowledge in a machine executable form). Therefore, it is highly important to enable sharing and reuse of the pedagogical knowledge as much as possible.

One approach towards the reuse and the exchange of pedagogical knowledge is based on the reliance on standards and official specifications. The most relevant one is the IMS Learning Design[3] (IMS LD) specification. The primary aim of IMS LD is to provide a common information model for representing pedagogy that is conceptually abstracted from context and content, so that proven pedagogical patterns can be shared and reused across instructional contexts and subject domains; as well as shared among different learning systems. Still, this specification can be improved by using ontologies. Giving a formal definition of semantics for such information models provides a stronger basis for integration into different systems. For example, Amorim et al. [18] developed an OWL ontology based on the IMS LD information model in order to address the limited expressivity of the official specification.

Not only are the semantics of learning designs more precise through the use of ontologies, but it is possible to relate elements of learning designs with various aspects characterizing specific contexts of their use. For example, learning activities can be connected with domain knowledge, learning content, and learner models of previous learners who participated in learning activities. In fact, the LOCO framework exactly addresses this problem, by providing a set of ontologies for defining learning context as an interplay of learning activities, learning content (and concepts), and users [19]. It makes the first steps towards materialization of the ecological approach [20] to the learning environments by fully leveraging formal semantics of ontologies and collective experiences of the learning content usage in previous learning contexts. For ITSs, this offers a tremendous potential for evaluating the quality of the shared learning designs, as well as all other shared learning resources (e.g., as it is shown in the LOCO-Analyst system [19]). For example, ITS developers may benefit from integration of an existing learn design into the teaching module. The teaching module itself may have heuristics and rules that reason over such shared learning designs for generating an instructional plan (e.g., to refer students to how their friends (FOAF) successfully completed learning activities on a certain topic).

Not all research efforts aimed at developing ontological representations of instructional knowledge are based on IMS LD. Mizoguchi et al [21] have developed a comprehensive ontology that incorporates different theories and paradigms about instructional and learning design. The ontology came as a result of ten years of research commitment for providing a comprehensive and sharable model of instructional design knowledge. It is built based on the philosophical consideration of all the necessary concepts for understanding learning, instruction and instructional design, and, as such, should enable increased theory-awareness in authoring tools.

[3] http://www.imsglobal.org/learningdesign/

Based on a student's current learning context and the system's knowledge about the student stored in his/her student module, the teaching module schedules content selection and presentation, tutoring feedback, learner assessment, and other teaching actions. This means that the teaching module also stores procedural knowledge that describes the dynamics of the system. This knowledge often takes the form of different kinds of rules, which are typically represented in well-known rule-based languages (e.g., Jess, Lisp). Recently, some researchers have proposed the use of the Semantic Web Rule Language[4] (SWRL) for designing a system's dynamics. For example, Wang & Kim [22] proposed an extension of SWRL aimed at making this language sufficiently expressive for practical ITS development. They have also developed a concrete implementation of a teaching strategies engine using the extended SWRL and a standard rule engine.

All of the above ideas promote sharing different types of knowledge among different learners. However, as this may also affect the privacy of learners, it is also important to explore how policies can be integrated in the teaching module. For example, policies can be used for defining access rights to certain resources based on the student's current context and/or role or by negotiating trust (e.g., PeerTrust used in the ELENA project) [23]. This is especially relevant for mobile learning contexts, where the given context of learning (e.g., in a classroom) may be used by the teaching module to suggest collaboration or sharing of experiences with some peers.

3.4 Benefits for Student-Tutor Interaction

Some modern ITSs leverage improvements in natural language understanding techniques to accurately understand student utterances and to respond in productive and realistic ways. Beginners who have yet to refine their domain vocabularies - but are typically, surprisingly, consistent in their language patterns - are ideal targets for this technology [24]. Collaborative tagging can be leveraged to further improve the performance of these systems: folksonomies resulting from the students' tagging activities reflects the vocabulary they will typically use when communicating with the system, so it follows that those tags can be used for training the dialog component of an ITS.

4 Conclusions

The Social Semantic Web offers new approaches and technologies for making use of the not-so-formal contributions of users in many different systems. The educational domain is particularly well suited for first investigations of these approaches and technologies, since the goals and paths of its users are well defined and understood. Further, learners in online, distance, and independent learning situations – such as those targeted by ITSs – stand to benefit greatly from the new possibilities. ITSs in particular are complex to design, implement and maintain. Even though they are usually focused on specific domains and address particular needs, knowledge still remains embedded in the different ITS modules. Making this information accessible and reusable will reduce the costs of future ITS development, and, at the same time, improve the quality of the always growing quantity of information they contain.

[4] http://www.w3.org/Submission/SWRL/

Through leveraging socially created knowledge and system interaction data from learner activities, we cite the most immediate benefit to ITSs as increased feedback and awareness of learner actions. This will increase pedagogical understanding of learner activities and enable timely and accurate evolution of domain ontologies. Further, through structuring system and social interaction data, this information can be easily shared between ITSs and other educational systems to enable more accurate and complete personalization possibilities. Finally, students benefit from increased interactions with peers, the intrinsic motivations from contributing to their own learning, and new possibilities in ITS functionalities.

In this paper, we have taken the first steps at outlining why new research and technologies in the Social Semantic Web should be leveraged in ITSs; given the traditional architecture and approach of ITS development. Our current endeavours include creating the infrastructure to support the incorporation of the Social Semantic Web paradigm into ITSs. To achieve this, we have already defined an extensive ontology which captures aspects of not only learners and instructional design, but also complex methods of representing and capturing social interactions and socially created knowledge. Further, we will continue to develop the tools and software libraries that will allow for (semi-)automatically migrating social sources of information into detailed structured knowledge representations, which can be shared among, and used in, educational systems such as ITSs.

References

1. Urban-Lurain, M.: Intelligent Tutoring Systems: An Historic Review in the Context of the Development of Artificial Intelligence and Educational Psychology (1996), http://www.cse.msu.edu/rgroups/cse101/ITS/its.htm#_Toc355707506
2. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American 284(5), 34–43 (2001)
3. O'Reilly, T.: What Is Web 2.0 – Design Patterns and Business Models for the Next Generation of Software (2005), http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html
4. Mikroyannidis, A.: Toward a Social Semantic Web. Computer 40(11), 113–115 (2007)
5. Breslin, J., Decker, S.: Semantic Web 2.0: Creating Social Semantic Information Spaces. Tutorial at the World Wide Web Conference 2006, Edinburgh, Scotland (2006), http://www2006.org/tutorials/#T13
6. Gruber, T.: Collective Knowledge Systems: Where the Social Web meets the Semantic Web. J of Web Semantics (in press, 2008)
7. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: A Nucleus for a Web of Open Data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ISWC 2007. LNCS, vol. 4825. Springer, Heidelberg (2007)
8. Zouaq, A., Nkambou, R., Frasson, C.: An Integrated Approach for Automatic Aggregation of Learning Knowledge Objects. Interdisciplinary Journal of Knowledge and Learning Objects (IJKLO) 3, 135–162 (2007)
9. Murray, T.: Authoring Knowledge-Based Tutors: Tools for Content, Instructional Strategy, Student Model, and Interface Design. J. of Learning Sciences 7(1), 5–64 (1998)

10. Mizoguchi, R., Bourdeau, J.: Using Ontological Engineering to Overcome Common AI-ED Problems. Int. J. AI in Education 11, 1–12 (2000)
11. Mitrović, A., Devedžić, V.: A model of multitutor ontology-based learning environments. Int'l J. of Continuing Engineering Education and Life-Long Learning 14(3), 229–245 (2004)
12. Motta, E., Sabou, M.: Next Generation Semantic Web Applications. In: Asian Semantic Web Conf (keynote), China (2006)
13. Torniai, C., Jovanović, J., Gašević, D., Bateman, S., Hatala, M.: E-learning meets the Social Semantic Web. In: 8th IEEE Int'l Conference on Advanced Learning Technologies (ICALT 2008), Santander, Cantabria, Spain (to appear, 2008)
14. Dolog, P., Schaefer, M.: A Framework for Browsing, Manipulating and Maintaining Interoperable Learner Profiles. In: Proc. of the 10th Int'l Conf. on User Modeling, Edinburgh, UK (2005)
15. Niederee, C., Stewart, A., Mehta, B., Hemmje, M.: A multi-dimensional, unified user model for cross-system personalization. In: Proc. of the AVI Workshop on Environments for Personalized Information Access, Italy, pp. 34–54 (2004)
16. Heckmann, D., Schwartz, T., Brandherm, B., Kröner, A.: Decentralized User Modeling with UserML and GUMO. In: Proceedings of the Workshop on Decentralized, Agent Based and Social Approaches to User Modelling, Edinburgh, Scotland, pp. 61–65 (2005)
17. Ounnas, A., Davis, H.C., Millard, D.E.: Semantic Modeling for Group Formation. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511. Springer, Heidelberg (2007)
18. Amorim, R.R., Lama, M., Sánchez, E., Riera, A., Vila, X.A.: A Learning Design Ontology based on the IMS Specification. Educational Technology & Society 9(1), 38–57 (2006)
19. Jovanović, J., Gašević, D., Brooks, C., Devedžić, V., Hatala, M., Eap, T., Richards, G.: Using Semantic Web Technologies for the Analysis of Learning Content. IEEE Internet Computing 11(5), 45–53 (2007)
20. McCalla, G.: The Ecological Approach to the Design of E-Learning Environments: Purpose-based Capture and Use of the Information about Learners. Journal of Interactive Media in Education. Special Issue on the Educational Semantic Web, 2004/1 (2004)
21. Mizoguchi, R., Hayashi, Y., Bourdeau, J.: Inside Theory-Aware and Standards-Compliant Authoring System. In: Proc. of the 5th Int'l Workshop on Ontologies and Semantic Web for E-Learning (SWEL 2007), Marina del Rey, CA, USA, pp. 1–18 (2007)
22. Wang, E., Kim, Y.S.: Using SWRL for ITS through Keyword Extensions and Rewrite Meta-Rules. In: Proc. of the 5th Int'l Workshop on Ontologies and Semantic Web for E-Learning (SWEL 2007), Marina del Rey, CA, USA, pp. 101–106 (2007)
23. Bonatti, P., Olmedilla, D.: Rule-Based Policy Representation and Reasoning for the Semantic Web. In: Antoniou, G., Aßmann, U., Baroglio, C., Decker, S., Henze, N., Patranjan, P.-L., Tolksdorf, R. (eds.) Reasoning Web. LNCS, vol. 4636, pp. 240–268. Springer, Heidelberg (2007)
24. Lane, H.C., VanLehn, K.: Teaching the tacit knowledge of programming to novices with natural language tutoring. Computer Science Education 15(3), 183–201 (2005)

Structurization of Learning/Instructional Design Knowledge for Theory-Aware Authoring Systems

Yusuke Hayashi[1], Jacqueline Bourdeau[2], and Riichiro Mizoguchi[1]

[1] ISIR, Osaka University, 8-1 Mihogaoka, Ibaraki, Osaka, 567-0047 Japan
{hayashi,miz}@ei.sanken.osaka-u.ac.jp
[2] LICEF, Télé-université, 100 Sherbrooke W., Montréal, (QC) H2X 3P2 Canada
jacqueline.bourdeau@licef.teluq.uqam.ca

Abstract. Currently, there are little guidelines on building quality standard-compliant learning courses. Although educational theories can be guidelines, there are difficulties in the practical use. An approach to the problem is to build a theory-aware and standard-compliant authoring system based on an ontology that establishes a multi-paradigm conceptual basis for learning/instructional theories. This paper discusses the feasibility of building such an ontology, as well as the functionality of a theory-aware authoring tool based on it.

1 Introduction

Standard technologies in the field of Technology-Enhanced Learning (TEL) are currently undergoing remarkable development [5]. However, a significant problem has been noted: in most of the learning/instructional design of TEL courses, learning objects (LOs) are combined with little justification, which can result in low-quality learning courses. A cause of this problem is in fact a disjunction between the learning/instructional theories and the standard technologies. Although the theories may not be always true, they can provide guidelines for quality design.

As a key to the solution of the problem, the concern with ontologies has been growing [6]. We have made a study on an ontology for educational theories and a theory-aware and standard-compliant authoring system based on it [19, 20]. Issues addressed in this study are 1) to make computers understand and utilize a variety of educational theories and 2) to keep sharability of designed scenarios with theoretical justification. To achieve the understanding and the utilization requires a conceptual basis to ensure the compatibility among representations of theories. On the other hand, the sharability with justification requires a design environment for authors to articulate the design rationale of a learning/instructional scenario and to output the resultant scenario in a standard format such as IMS Learning Design (LD) [1]. As the ongoing results we have published OMNIBUS ontology and an authoring system SMARTIES on the project website[2].

[1] http://www.imsglobal.org/learningdesign/
[2] The OMNIBUS ontology and SMARTIES can be downloaded for free from the OMNIBUS project web site (http://edont.qee.jp/omnibus/)

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 573–582, 2008.

This paper discusses the confirmation of the working hypothesis on which the OMNIBUS ontology is built and the feasibility of a theory-aware and standard-compliant authoring system SMARTIES. This paper is structured as follows. Section 2 describes the overview of the OMNIBUS ontology and a modeling framework for learning/instruction based on it. Section 3 presents the system structure and the functionalities of SMARTIES. Section 4 considers the above results from three viewpoints: confirmation of working hypothesis, theory-blending support on SMARITES and related studies about authoring tools. Finally, the last section concludes this paper and shows other potential of the OMNIBUS ontology than SMARTIES.

2 A Framework for Modeling Learning/Instructional Theories

2.1 OMNIBUS: A Learning-Support-Related Theory Ontology

The underlying philosophy of building OMNIBUS ontology is that all the learning/instructional actions can be defined relying on the state of learners that has been changed according to such actions. For example, cognitivism pays attention to cognitive processing inside a learner while constructivism pays attention to interaction with others or the environment. Those paradigms seem to deal with different types of changes of state because cognitivism deals with a learner's internal changes in the cognitive process whereas constructivism treats the external state affected by interaction. In practice though, the educational strategies proposed by these theories are compatible and even complementary (e.g. [24]).

The hypothesis in this study is that establishing a multi-paradigm set of states of the learner, for example, states concerning change of the cognitive structure as a result of the learning process, can help to make a connection among the various internal and external states of learners with which each paradigm deals [9]. The most serious issue is, of course, whether such a multi-paradigm set can be found or not. It is obvious that there are many different states specific to each learning theory at a detailed level. However, if based on the important guiding principle of "engineering approximation," the states can be categorized into several groups such as those in common within some of the paradigms, those in common within all the paradigms or ones specific to each theory.

2.2 A Framework for Modeling Learning/Instructional Scenario

In the OMNIBUS ontology, learning/instructional processes are modeled from two viewpoints: "what" to achieve and "how" to achieve [9]. Figure 1 shows an example of the model. The oval nodes represent *I_L events*, in which "I_L" stands for the relationship between the Instruction and the Learning. An I_L event is composed of state change of a learner and actions of the learner and the instructor related to the change. This describes what learner state is achieved by the actions. Black squares linking the macro and the micro I_L events represent ways of learning/instructional goals achievement (hereafter, WAYs). A WAY means the relation in which the macro is achieved by the micros. The micros basically have smaller grain-sized state to be achieved than the macro's. A WAY describes how the state in the macro can be achieved by the sequence of smaller grain-sized states.

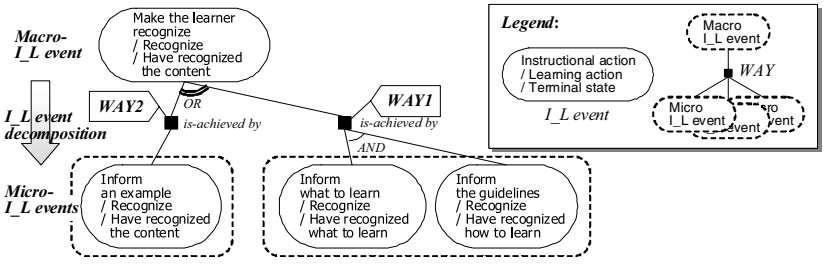


Fig. 1. Example of WAY to achieve learning/instructional goals

One of the characteristics of this framework is "OR" relation between WAYs. As shown in Figure 1, the macro has two WAYs; WAY1 and WAY2. It indicates that there are two alternative ways to achieve the macro (where the leaner recognizes what to learn). WAY1 is an instructor-led process in which the instructor directly informs what and how to learn, on the other hand WAY2 is a learner-led process in which the instructor gives a demonstration without explanations. These two WAYs have a common goal (expected state-change of the learner) but take different approaches to achieve it.

In this manner, making a distinction between "What" and "How" to achieve will further clarify the differences and commonalities of ways for learning/instruction. Moreover, this also works as guidelines to consider alternative ways to achieve the same goal. WAY is defined as the relational concept [16] on Hozo ontology editor[3] and an example of the detailed definition is shown in [20].

2.3 Modeling Strategies from Theories in the Form of WAY

Each strategy from a theory (hereafter called strategy) describes relationships between relevant learning/instructional approaches for possible situations and expected learning outcome. Moreover, a learning/instructional strategy is considered as an aggregation of such relationships. The relevant modeling of the theories would be declaratively describing these relationships. Consequently, this study proposed to model the schemes described by theories as WAYs. Such WAYs are called WAY-knowledge.

In this study, a hundred pieces of WAY-knowledge are defined based on the eleven theories. Table 1 is a summary of the amount of WAY-knowledge roughly sorted into four categories of the theories/models. *Cross-paradigm*, *Cognitivism* and *Constructivism* are in the grouping axis based on the differences in the paradigm of "Learning (mechanism)" but *Instructional management* is in a different axis because it deals with preparing learning conditions such as motivation. Another typical paradigm is the *Behaviorism*, but we excluded it from Table 1 since its WAY-knowledge has not been defined currently. We discuss the content of WAY-knowledge in 3.1.

[3] Hozo ontology editor can be downloaded for free from the Hozo web site (http://www.hozo.jp/)

Table 1. Content of WAY-knowledge Definition

	Categories of theory/model			
	Cross-paradigm	Cognitivist	Constructivist	Instruction management
Number of Theories/models	1	3	6	1
Amount of WAY-knowledge	2	30	51	16
Amount of WAY-knowledge by a theory/model	Dick and Carey's I-model [7] 2	Component display theory [18] 21	Constructivist learning environment design [13] 22	Keller's I-Theory (ARCS model) [14] 16
		Gagne's I-Theory [8] 8	STAR LEGACY model [25] 18	
		Merrill and Tennyson's I-Theory [17] 1	Scaffolding theories [11, 12] 3	
			Cognitive apprenticeship [4] 8	

3 SMARTIES: A Theory-Aware and Standard-Compliant Authoring Tool

SMARTIES is a prototype of an authoring tool that supports authors to design learning/instructional scenarios based on the OMNIBUS ontology [20]. At the same time, this is a standard-compliant system that can output the models in the IMS LD format. All the information in scenario models designed using SMARTIES, which includes the theoretical justification of the scenario, can be referred to from IMS LD-compliant tools such as Reload LD player [10]. Unlike other systems in which theories are embedded in a procedural manner, the assistance that SMARTIES offers is provided based on the declarative knowledge defined by the OMNIBUS ontology which enables scenario generation by the flexible use of multiple theories and automatic generation of explanations about the theories and scenarios generated.

3.1 An Overview of SMARITES

The current scope of SMARTIES is the design phase, one of the five major phases of Instructional design: analysis, design, development, implementation and analysis. SMARTIES assist the scenario design from the abstract level to the concrete level, in other words, from goal setting of a scenario to assignment of learning objects (LOs) to it. In SMARTIES the scenario design is done by decompositions of the goal of a scenario into sub-goals as WAYs. This process is to externalize the design rationale of the scenario as well as to specify LOs used in it. Finally, the resultant scenario model is output in IMS LD format and can be executed on IMS LD compliant tools.

Figure 2 illustrates the system structure of SMARTIES. SMARTIES supports three kinds of authors: Scenario authors, Knowledge authors and Ontology authors. Scenario authors are instructional designers or teachers for example, who designs scenario models through the scenario editor with reference to concepts defined by the ontology and the educational theories described as WAY-knowledge. Knowledge

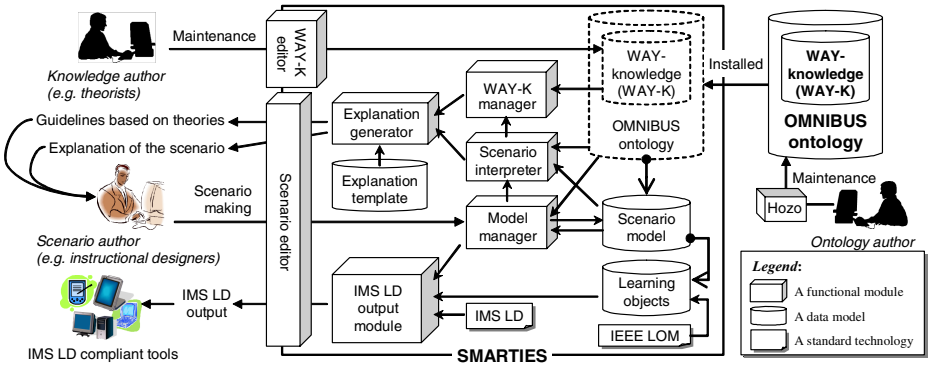


Fig. 2. The system structure of SMARTIES

authors describe learning/instructional design knowledge such as theories, best practices and their own heuristics as a set of WAY-knowledge based on the ontology. The WAY-knowledge editor supports the task and stores the resultant pieces of WAY-knowledge are stored in WAY-knowledge base. Finally, ontology authors maintain the OMNIBUS ontology through the Hozo ontology editor, which is located outside of SAMRTIES.

3.2 The Scenario Design Support in SMARTIES

Figure 3 shows the screen shots of the scenario editor where the author designs a scenario model for learning micro scopes based on STAR Legacy model [25]. This screen shot shows how an author makes a scenario model using "WAY-knowledge". The scenario editor (Fig. 3(1)) is the main window, which provides a scenario author (hereafter, author) with an environment to describe a scenario model (Fig. 3(a)) as a tree structure of I_L events in which the root describes the goal of the whole scenario. In this window, the author decomposes the root I_L event step-by-step. For each I_L event, the setting panel is provided to the author (Fig. 3(2)). In the panel he/she can refer to the ontology and choose concepts to describe I_L event (Fig. 3(3)). The authors can describe the decomposition of each I_L event by the author-defined WAY (Fig. 3(4)) or WAY-knowledge stored in SMARTIES.

The Way proposal window (Fig. 3(5)) provides an author with applicable WAY-knowledge in order to assist the author to decompose each I_L event in the scenario model. The list of applicable pieces of WAY-knowledge is shown in Fig 3(d). This is the result of the I_L event pattern matching based on OMNIBUS ontology. The I_L event selected in the scenario editor is compared with the macro I_L event of all the pieces of WAY-knowledge. If these match up, the WAY-knowledge is applicable. When the author chooses one of them, a proposed decomposition by the WAY-knowledge is displayed on the viewer (Fig. 3(e)). If the author decides to adopt the selected Way, the proposal is applied to the scenario editor. By repetition of such a process, an author makes scenario model from abstract levels to concrete ones.

At the end of the scenario design, learning objects are linked to a scenario model. The leaf nodes depicted by rounded rectangles in the scenario model stand for LOs. In

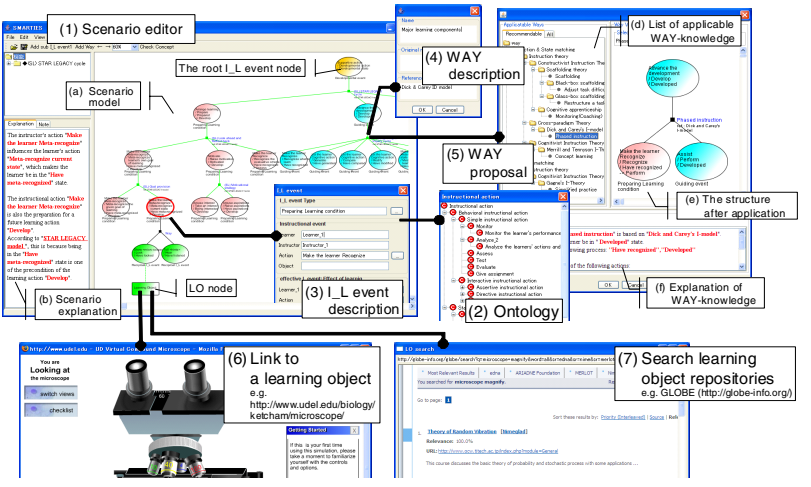


Fig. 3. Screenshot of SMARTIES

this example, a simulation of a micro scope (Fig. 3(6)) is set in order to materialize the leaf I_L event. This is because the goal in this I_L event is to remind the learners of the procedure (the manipulation of micro scopes) and the transformational media, which are visual images illustrating changes in time or over space, is appropriate to such content according to Clark's multimedia principle [3]. Authors can set an LO that they made or know as well as search LO repositories for LOs appropriate to the requirement (Fig. 3(7)). Although just the content type is used to discuss the requirement of LO in the current implementation, much more properties are considered to be used to specify the requirement, for example, learner characteristics such as age and prior knowledge, domain characteristics of the content and context characteristics such as mode of instruction and delivery listed in [20]. Enumerating such properties and linking them to LOM elements for LO search is currently in progress.

4 Discussion

In this section, we discuss the confirmation of the working hypothesis of the OMNIBUS ontology and the feasibility and the advantages of SMARTIES.

4.1 Confirmation of the Working Hypothesis

In order to confirm the working hypothesis discussed in 2.1, we summarized the usage of "state" in WAY-knowledge in Table 2. In this section, we examine if the characteristics and the common ground of each paradigm are properly extracted.

The states are classified roughly into six groups. The *Learning stage* is a state related to the progress of learning such as "Preparation," "Development" and "Assessment." The *Cognitive process state* is a state regarding the learner's recognition process. The *Meta-cognitive process* state is a state regarding the metacognition process. The *Attitudinal state* is a state regarding the learner's attitude and interest such as

Table 2. Distribution of the States used in the WAY-knowledge

		The categories of theory/model			
		Cross-paradigm	Cognitivism	Constructivism	Instruction management
The statistics of theory/model	Number of Theories/models	1	3	6	1
	Amount of WAY-knowledge	2	30	51	16
	Number of states	8	77	132	39
Percentage of each category of states	Learning stage	**71.4**	4.8	6.3	0.0
	Cognitive process state	14.3	**61.9**	36.7	35.9
	Meta-cognitive process state	0.0	15.9	**41.4**	12.8
	Attitudinal state	0.0	9.5	4.7	**43.6**
	Developmental state	**14.3**	0.0	0.8	0.0
	External state	0	7.9	**10.2**	7.7

"Motivation." The *Developmental state* is a state regarding the developmental stages of knowledge and skills and is defined following Bloom's taxonomy [2]. Lastly, the *External state* is a state regarding the learner's communication with others or the environment and has the subclasses such as "Informed" or "Asked."

In Table 2, the theory/model classification mostly used in each state classification is depicted in boldface. This result indicates that various learning/instructional strategies extracted from each theory reflect their characteristics. For example, the cognitivism uses many *cognitive states* because it focuses on the knowledge processing process while the constructivism uses many *meta-cognitive states* that are related to the meta-cognition process. The cross-paradigm theory uses many *learning stages* because it is directed to general learning/instructional processes in which the differences in paradigms are minor.

As discussed above, by modeling strategies extracted from theories as WAY-knowledge focusing on the learner state, the similarities and differences in paradigms are clarified. As Table 2 shows, there are the overlaps of the usage of state categories among paradigms. Therefore, multiple theories from multiple paradigms might be blended in one scenario with the common state concept working as the common ground. Since there do not exist theories that support theory-blending, the validity of blending is not always assured. However, we consider that its feasibility can be presented by the framework of WAY-knowledge that we propose in this study and the theory-blending function discussed below will provide useful information for the instructional designers and teachers.

4.2 Consideration of the Alternative Strategy Applicable for a Scenario Model

To consider the feasibility of the theory-blending support of SMARTIES, we have modeled a scenario on SMARTIES and examine the possibility of theory-blending. This scenario is from [24] and based on the theory by Gagne and Briggs [8]. The original scenario consists of 11 steps. On the other hand, the scenario model built on SMARTIES has 15 leaf nodes (I_L events) therefore consists of 15 steps. The reason for the difference is that several steps in the original scenario include multiple interactions between the instructor and the learner. One interaction is described as one I_L event in the scenario model therefore one step of the original scenario may be decomposed into several steps in the scenario model depending on the granularity of interaction.

Most of the WAYs in the scenario model were able to be described by the pieces of WAY-knowledge extracted from the principles by Gagne and Briggs. The only part

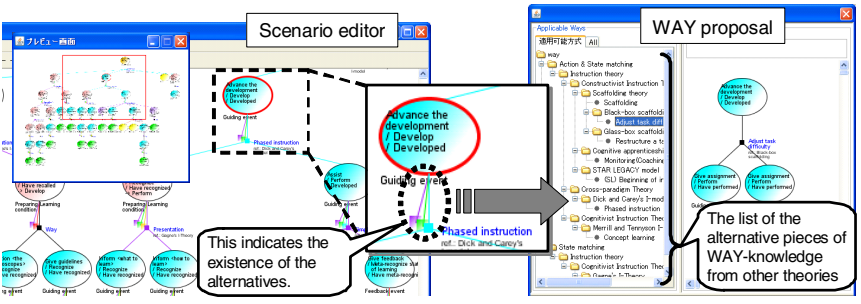


Fig. 4. The experimental result displayed on SMARTIES

that was not described by the pieces of WAY-knowledge is the lower layer of the scenario model, which mainly consists of an I_L event where actions are decomposed into concrete actions (i.e., Tell and Listen, which cannot be decomposed more in the OMNIBUS ontology.) Therefore, we believe the WAY-knowledge prepared for this study was sufficient to build an overall storyline of the scenario.

In order to know the applicability of other theories to this scenario model, we checked all the WAYs in the scenario model to see if they had any other pieces of WAY-knowledge applicable to themselves. This experiment was conducted on SMARTIES and Figure 4 shows the result displayed on SMARTIES. The result showed that one or more substitutable pieces of WAY-knowledge were detected from 99 pieces of WAY-knowledge for each of the 23 WAYs out of the 27 WAYs in the scenario model. These alternatives include some pieces of WAY-knowledge extracted from other theories or paradigms. This can be considered that SMARTIES can pose the possibility for the application of other theories than the Gagne and Briggs's to this scenario model. Of course, as mentioned in the previous section, not all the alternatives are assured to be pedagogically relevant to be used in this scenario model, but they can work as helpful information for scenario authors. From this point of view, we may suggest the possibility of designing learning/instructional scenarios based on the multiple theories beyond paradigms through the modeling framework based on the OMNIBUS ontology.

4.3 Related Studies Concerning the Authoring Systems

A number of authoring systems have been suggested in the area of learning/instruction support systems [21]. In this section, concerning the support functions for learning/instructional scenario design, we make a comparison between SMARTIES and the other major theory-based authoring systems.

As for the authoring systems that include theoretical knowledge, the CREAM tools [22] and CTAT (Cognitive Tutor Authoring Tools) [15] can be named as representative examples. They are based on the theories by Gagne [8] and by Anderson [1] respectively. By designing the support functions based on each theory, these tools can provide the author with detailed support. However, these two are based on a single theory and the author has to use another authoring tool if he/she wishes to use other theories because the content of the single theory is deeply embedded in the functions.

Furthermore, the clear correspondence between the assistance functions and the theory is kept only in the system developer's head and the authors cannot come to know it clearly when using the system.

SMARTIES has the advantages of these authoring tools as well as complementing their disadvantages. It can provide design guidelines based on multiple theories accumulated as WAY-knowledge. Focusing on only one theory, of course, has the advantage that an authoring system has theoretical consistency from the behavior of the system to the user interface. However, the problem is the scalability concerning the accumulation of knowledge. Some sort of general-purpose framework to systematize theories is required in order to make the authoring tool continuously usable in response to the change of theories. The WAY-knowledge is scalable in SMARTIES and knowledge authors can increase the number of the theories to be dealt with by increasing the number of pieces of the WAY-knowledge.

5 Conclusions

We have discussed the feasibility of building a multi-paradigm conceptual basis for learning/instructional theories, as well as the functionality of a theory-aware authoring tool based on it. As discussed in Section 4, the OMNIBUS ontology works as a conceptual basis for describing learning/instructional strategies from theories and SMARTIES has the scalability concerning theories to be dealt with. Although the definitions of the ontology need to be refined from the experts' perspective of learning/instructional theories, its significance lies in the fact that it presented the feasibility of structurizing various learning/instructional theories. In addition, it is considered that the framework of WAY-knowledge can deal with not only theories but also some other types of design knowledge such as best practices and teachers' heuristics.

Furthermore, the OMNIBUS ontology suggests other possibility of the theory-awareness than SMARTIES. CIAO [23] is a scenario analysis agent that also works based on the OMNIBUS ontology. This interprets the scenario described in the IMS LD format and infers the design rationale of it. [26] proposes an ITS based on the OMNIBUS ontology. In the ITS, pieces of WAY-knowledge are converted to SWRL rules and used by the ITS to select an instructional strategy appropriate to the learner.

References

1. Anderson, J.R.: Rules of the Mind. Erlbaum, Hillsdale (1993)
2. Bloom, B.S., Hastings, J.T., Maclaus, G.F.: Hanbook on Formative and Summative Evaluation of Student Learning. McGraw-Hill (1971)
3. Clark, R.C., Mayer, R.E.: e-learning and the science of instruction, 2nd edn., Preiffer (2007)
4. Collins, A., Brown, J.S., Newman, S.E.: Cognitive apprenticeship: Teaching the crafts of reading, writing and mathematics. Knowing, learning, and instruction: Essays in honor of Robert Glaser, 453–494 (1989)
5. Devedzic, V.: Semantic Web and Education. Springer (2006)
6. Dicheva, D.: O4E Wiki, http://o4e.iiscs.wssu.edu/xwiki/bin/view/Blog/Articles
7. Dick, W., Carey, L., Carey, J.O.: The systematic design of instruction, 5th edn. Addison-Wesley Educational Publisher Inc. (2001)

8. Gagne, R.M., Briggs, L.J.: Principles of Instructional Design, 2nd edn., Holt, Rinehart and Winston, New York (1979)
9. Hayashi, Y., Bourdeau, J., Mizoguchi, R.: Ontological Support for a Theory-Eclectic Approach to Instructional and Learning Design. In: Nejdl, W., Tochtermann, K. (eds.) EC-TEL 2006. LNCS, vol. 4227, pp. 155–169. Springer, Heidelberg (2006)
10. Hayashi, Y., Bourdeau, J., Mizoguchi, R.: Theory-aware Explanation Support for Standard-compliant Scenario Building. In: Proc. of the Workshop on Semantic Technology for Learning (held in ICCE 2007), pp. 104–109 (2007), http://www.ei.sanken.osaka-u.ac.jp/hayashi/ICCE2007WS-CR-hay-final.pdf
11. Hmelo, C.E., Guzdial, M.: Of black and glass boxes: Scaffolding for doing and learning. In: Proc. ICLS 1996, pp. 128–133 (1996)
12. Hogan, K., Pressley, M.: Scaffolding Student Learning: instructional approaches and issues. Brookline Books, Cambridge (1997)
13. Jonassen, D.: Designing constructivist learning environment. Instructional-design theories and models A new paradigm of instructional theory, 215–239 (1999)
14. Keller, J.M., Kopp, T.W.: An application of the ARCS model of motivational design. Instructional theories in action: Lessons illustrating selected theories and models, 289–320 (1987)
15. Koedinger, K.R., Aleven, V.A.W.M.M., Heffernan, N.T.: Toward a Rapid Development Environment for Cognitive Tutors. In: Proc. of AIED 2003, pp. 455–457 (2003)
16. Kozaki, K., Kitamura, Y., Ikeda, M., Mizoguchi, R.: Hozo: An Environment for Building/Using Ontologies Based on a Fundamental Consideration of "Role" and "Relationship". In: Gómez-Pérez, A., Benjamins, V.R. (eds.) EKAW 2002. LNCS (LNAI), vol. 2473, pp. 213–218. Springer, Heidelberg (2002)
17. Merrill, M.D., Tennyson, R.D.: Concept Teaching: An Instructional Design Guide. Educational Technology, Englewood Cliffs (1977)
18. Merrill, M.D.: Component display theory, Instructional-design theories and models: An overview of their current status, Hillsdale, pp. 279–333 (1983)
19. Mizoguchi, R., Bourdeau, J.: Using Ontological Engineering to Overcome Common AI-ED Problems. IJAIED 11(2), 107–121 (2000)
20. Mizoguchi, R., Hayashi, Y., Bourdeau, J.: Inside Theory-Aware and Standards-Compliant Authoring System. In: Proc. of SWEL 2007, pp. 1–18 (2007)
21. Murray, T., Blessing, S., Ainsworth, S.: Authoring Tools for Advanced Technology Learning Environments: Toward Cost-Effective Adaptive, Interactive and Intelligent Educational Software. Springer (2003)
22. Nkambou, R., Gauthier, G., Frasson, C.: CREAM-Tools: An Authoring Environment for Curriculum and Course Building in an Intelligent Tutoring System. In: Proc. of CALICSE 1996, pp. 186–194 (1996)
23. Psyché, V.: CIAO, an Interface Agent Prototype to facilitate the use of ontology in intelligent authoring system. In: Proc. of I2LOR 2004 (2004)
24. Reigeluth, C.M. (ed.): Instructional Theories in Action: Lessons Illustrating Selected Theories and Models. Lawrence Erlbaum Associates, Hillsdale (1987)
25. Schwartz, D., Xiaodong, L., Brophy, L., Bransford, S., J.D.: Toward the Development of Flexibly Adaptive Instructional Designs, Instructional-design theories and models A new paradigm of instructional theory, pp. 183–213 (1999)
26. Wang, E., Kim, Y.S.: Issues in Integrating Teaching Strategies from Learning Theories and Extended SWRL Rules. In: Proc. of the workshop on Semantic Technology for Education (held in ICCE 2007), pp. 110–113 (2007), http://credits.skku.edu/credits/publications/WangKimSTfL07IssuesIntegrating.pdf

Expanding the Plausible Solution Space for Robustness in an Intelligent Tutoring System

Hameedullah Kazi[1,3], Peter Haddawy[1], and Siriwan Suebnukarn[2]

[1] Computer Science & Information Management Program,
Asian Institute of Technology, Thailand
[2] School of Dentistry, Thammasat University, Thailand
[3] Department of Computer Science, Isra University, Pakistan
{hameedullah.kazi,haddawy}@ait.ac.th, ssiriwan@tu.ac.th,
hkazi@isra.edu.pk

Abstract. The knowledge acquisition bottleneck is a problem pertinent to the authoring of any intelligent tutoring system. Allowing students a broad scope of reasoning and solution representation whereby a wide range of plausible student solutions are accepted by the system, places additional burden on knowledge acquisition. In this paper we present a strategy to alleviate the burden of knowledge acquisition for building a tutoring system for medical problem-based learning (PBL). The Unified Medical Language System (UMLS) is deployed as domain ontology and information structure in the ontology is exploited to make intelligent inferences and expand the domain model. Using these inferences and expanded domain model, the tutoring system is able to accept a broader range of plausible student solutions that lie beyond the scope of explicitly encoded solutions. We describe the development of a tutoring system prototype and report the evaluation of system correctness in accepting such plausible solutions. The system evaluation indicates an average accuracy of 94.59 % when compared against human domain experts, who agreed among themselves with a statistical agreement based on Pearson Correlation Coefficient of 0.48 and $p < 0.05$.

Keywords: Robustness, intelligent tutoring systems, medical problem-based learning, UMLS, knowledge acquisition bottleneck.

1 Introduction

Intelligent tutoring systems typically present a problem scenario to the students, who solve the problem and receive hints from the system to help them in reasoning towards the correct solution. Solutions presented by students are evaluated by comparing them against a particular solution accepted by the tutoring system as being correct. Thus plausible student solutions that do not match that particular solution or a small set of stored solutions recognized by the system are often rejected by the system as being incorrect. This forces students to memorize expert solutions and stifles student creativity. Students should be able to use their understanding of concepts and concept relationships and learn how to apply their knowledge to given problems. This is particularly relevant in medical PBL, where a diverse set of solutions may be acceptable instead of a single perfect solution.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 583–592, 2008.

Plausible solutions may differ along a number of dimensions such as comprehensiveness, level of detail and the choice of alternate and synonymous terms to describe a concept. Different solutions to a problem may also be acceptable depending on the perspective one chooses to analyze from and this may particularly be applicable to ill-defined domains such as medical education [1]. The value of being able to accept a broader set of solutions is that it supports an approach to learning that promotes free thinking and novel solutions. To the best of our knowledge, the task of accepting a range of solutions larger than the explicitly encoded scope of solutions, is yet to be addressed in intelligent tutoring systems.

In order for a tutoring system to exhibit robust human-level tutoring, it needs broad knowledge to allow students to explore a large space of solutions and work creatively. Authoring tutoring systems typically requires knowledge acquisition in the three areas of domain expert knowledge, student model and pedagogical model [2]. Domain expert knowledge is acquired to equip the system with the curriculum knowledge that is to be taught to the students; pedagogical knowledge is acquired to equip the system with teaching techniques and strategies; and knowledge of the student model is acquired to help the system assess the knowledge level of the student. Acquiring and encoding the relevant knowledge can lead to a large overhead in the development time of a tutoring system [3, 4]. Manually encoding all knowledge into the system so that it can accept the full range of plausible solutions is not feasible and places great burden on human domain experts, whose time is very costly. A natural choice to overcome this knowledge acquisition bottleneck is to make use of existing knowledge available for reuse and sharing. The use of ontologies is a viable alternative in reducing the burden of knowledge acquisition for knowledge based systems.

Ontologies have been employed in the design of various tutoring systems [5, 6, 7], which often require cumbersome encoding of the ontology. The Constraint Acquisition System [8] uses a more automated approach of encoding the ontology constraints by learning from examples using constraint based modeling. However, it still requires the initial design of the ontology to be defined manually.

In the next few sections we describe how the burden of knowledge acquisition can be lightened for a medical tutoring system, through the use of the broad and widely available medical knowledge source UMLS, distributed by the U.S. National Library of Medicine [9]. We also describe a mechanism of exploiting the information structure in UMLS, through which the tutoring system can accept a range of plausible solutions larger than the explicitly encoded scope of solutions.

2 Related Work

The issue of brittleness and burden of knowledge acquisition has been addressed in the design of various intelligent tutoring systems [10, 11]. Kumar [10] discusses the use of model-based reasoning for domain modeling in the context of web-based tutoring system for helping students to learn to debug C++ programs. Arguing that rule-based systems are not flexible and are not adaptable to varying system behavioral discrepancies, he proposes model-based reasoning as an alternative.

The KASER [11] design is implemented in a tutoring system that teaches the science of crystal-laser design. They argue that production rules, when found to be in error, are corrected through explicitly encoding the revised rule back into the system.

They propose a production system, which initially requires explicit encoding of rules and can later self generate other rules as extension to the ones created earlier, easing the burden of knowledge acquisition.

The designs of medical tutoring systems built to date, have typically been based on customized knowledge bases that offer students a limited set of medical terms and concepts, to form their solution. The CIRCSIM-Tutor [12] teaches cardiovascular physiology by describing a perturbation of a cardiovascular condition, and initiating a question answer dialog with the student, to help the student in reasoning towards the correct solution. The system design lays emphasis on qualitative reasoning, but the scope of hypothesis (solution) representation is narrow, as students are confined to assigning values to a small set of variables for forming their hypothesis.

The SlideTutor [5] teaches students dermatopathology by presenting a visual slide as a problem scenario and asks students to classify the diseases. After observing the visual evidence presented in the slide, students present their hypothesis through a mouse driven menu selection, identifying features and their attributes from an ontology that has been manually encoded for the problem scenarios fed to the tutoring system. Solutions accepted by the tutoring system are also based on the ontology customized for the system. Thus students are not allowed to present alternative plausible hypotheses that may lie beyond the scope of this customized ontology.

This motivates the need to have a medical tutoring system that offers students a broad knowledge base of medical concepts and terms, such as the UMLS [9], to select their hypothesis concepts. This tutoring system should also be able to accept a wide variety of plausible hypotheses to a given problem.

3 Medical PBL and System Prototype

In a typical PBL session in the medical domain, a problem scenario is presented to a group of 6-8 students, who form their hypothesis in the form of a causal graph, where graph nodes represent hypothesis concepts and directed edges (causal links) represent cause effect relationships between respective concepts. The hypothesis graph is based on the Illness Script, where hypothesis nodes may represent enabling conditions, faults or consequences [13]. Enabling conditions are factors that trigger the onset of a medical condition, e.g., aging, smoking, etc.; faults are the bodily malfunctions that result in various signs and symptoms, e.g., pneumonia, diabetes, etc.; consequences are the signs and symptoms that occur as a result of the diseases or disorders, e.g., fatigue, coughing, etc.

There is difference of opinion on whether experts or non-experts should be used as a PBL tutor or facilitator [14, 15]. Different PBL tutors may also disagree over the extent to which a causal link is acceptable. While one tutor may find a causal link perfectly acceptable, another tutor may only be inclined to accept it with reservation. Quite often a PBL tutor may accept varying solutions that may differ in the choice of alternate terms or also in the level of detail with which the solutions are presented.

Our work is based on the extension of the COMET system [16, 17] designed to cover medical PBL for various domains. In the COMET system, each problem scenario is first referred to human domain experts who provide an expert solution that is

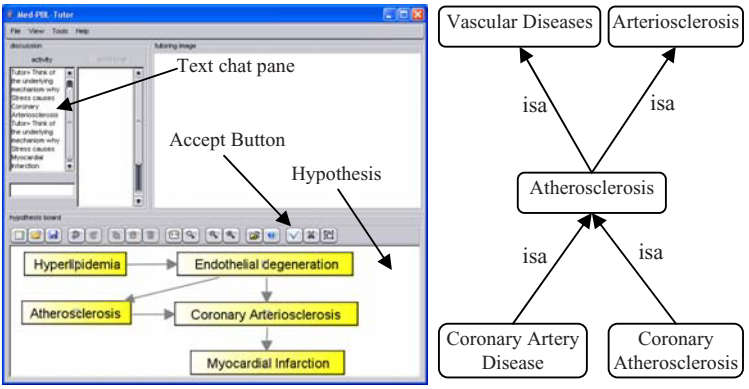


Fig. 1. (a) System Prototype Interface (b) Relationships in UMLS

eventually fed to the system. Student solutions are compared against this expert solution for evaluation. Thus a plausible student solution that does not match the expert solution is not entertained. The system allows students to form their hypothesis by choosing medical concepts from a repository manually encoded into the system.

We have developed a tutoring system for PBL in the medical domain. Problem solutions collected from experts are combined with UMLS tables to form the domain model. The pedagogical module of the system comprises of a hint generation mechanism that leverages off of the UMLS concept hierarchy and provides students a measure of partial correctness of their hypotheses [18]. Since the hint generation employs the rich domain knowledge of the UMLS in lieu of a student model, the design of our tutoring system does not include a student model.

The problem representation in our system is the same as that in COMET of a directed acyclic graph for forming the hypothesis. The student user is provided with a workspace as a hypothesis board to form the hypothesis, along with a text chat pane that returns hints from the system to guide the student in his/her clinical reasoning, as shown in Figure 1 (a). The student chooses concepts from the UMLS Metathesaurus [9] as hypothesis nodes and draws edges between nodes, using a mouse. The problem solving activity begins as the student is presented a problem scenario, such as:

"Mr. Heng-heng is a 48-year-old admitted with severe chest pain" … "His father died of heart attack aged 55. He smokes 20 cigarettes a day. He is still in pain"…

After studying the above problem description related to heart attack, the student hypothesizes that *endothelial degeneration* is a cause of *coronary arteriosclerosis*, which is shown to be a cause of *myocardial infarction*, as shown in Figure 1 (a).

4 System Knowledge Base and UMLS Knowledge Source

The UMLS [9] is a widely available medical knowledge source and is essentially a collation of various medical ontologies and terminologies (MeSH, SNOMED-CT, Gene Ontology, etc). The broad and diverse UMLS contains over 1 million medical

concepts covering various medical domains and about 135 semantic types, where each medical concept is assigned at least one semantic type [9].

The UMLS has been studied for use in many intelligent system applications [19, 20, 21, 22]. However, to the best of our knowledge, UMLS has not been used as the main knowledge source for inference purposes in an intelligent tutoring system. The Docs 'n Drug tutoring system [23] employs the use of medical terminologies that are a subset of UMLS, to allow students to choose concepts from these incrementally expandable terminologies. However, this system does not exploit the knowledge structure within these terminologies and make inferences for reasoning purposes.

The design of our system knowledge base comprises of UMLS tables and an additional table that is henceforth referred to as the *expert knowledge base*. The *expert knowledge base* is encoded with the help of human domain experts, and it contains the causal relationship between various medical concepts, such as:

Hyperlipidemia → *Endothelial Degeneration*
Endothelial Degeneration → *Coronary Arteriosclerosis*
Endothelial Degeneration → *Atherosclerosis*
Coronary Arteriosclerosis → *Myocardial Infarction*

Student solutions that are considered acceptable by human experts are merged into the *expert knowledge base*. The *expert knowledge base* is formed through the collation of expert solutions to various problem scenarios, along with the student solutions that are certified by the experts to be correct. The construction of an expert solution requires about 2-3 hours. Since each solution is in the form of a hypothesis graph, the collation of different solutions implies the incremental addition of the causal links in each solution, to the *expert knowledge base*. The goal is to expand the *expert knowledge base* over time. This is achieved through a convenient mechanism of knowledge acquisition, when student hypotheses are intermittently referred to a human tutor. While examining a plausible student hypothesis, the human tutor has the option to click the *Accept Button*, as shown in Figure 1 (a). This adds all links in the student hypothesis being examined, to the *expert knowledge base*.

5 System Reasoning and Hypothesis Evaluation

Each hypothesis causal link drawn by the student is evaluated by the system. The system refers to its knowledge base to check whether the link is acceptable. If the link is found to be acceptable, the system allows the directed edge (causal link) to be drawn; otherwise the system disallows the edge to be drawn and returns an appropriate hint as feedback to the student. The acceptability of the student hypothesis link is evaluated by comparing it against the *expert knowledge base*. If there is a match, the link under evaluation is considered acceptable; however if the link is not found in the *expert knowledge base*, the system makes use of a heuristic method to see if the link is close to acceptable.

This heuristic method makes use of the information structure within UMLS to make inferences. We use the relationships *alike* and *child-parent* for our purpose of heuristic based evaluation of the student solutions. A hierarchy based on *parent-child* relationships between UMLS concepts is shown in Figure 1 (b). While evaluating a

causal link, we consider the *alike* and *parent-child* relationships of each of the parent node and the child node in the causal link. All concepts that are found to have *alike* relationship or *parent* relationship with the parent node are considered equivalent to the parent node, while all concepts that have *alike* or *parent* relationship with the child node are considered equivalent to the child node. For a causal link under evaluation, let set *A* contain all concepts considered equivalent to the parent node and let set *B* contain all concepts considered equivalent to the child node. If the system finds a causal link in the *expert knowledge base* whose parent node is a concept from set *A* and whose child node is a concept from set *B*, then the link under evaluation will be considered acceptable.

Our experiments with expert ratings of causal links, revealed that links with concepts belonging to the four terminologies of Medical Subject Headings (MSH), UMLS Metathesaurus (MTH), Systematized Nomenclature of Medicine-Clinical Terms (SNOMEDCT) and National Cancer Institute (NCI), were found to be more acceptable. Thus in order to reduce the number of erroneous links, the system only accepts causal links whose concepts belong to any of these four terminologies.

The tutoring system starts by evaluating each causal link in the hypothesis separately, and then evaluates the graph as a whole. A hypothesis is considered valid if it contains a chain of reasoning, where nodes representing enabling conditions lead to other nodes in succession that eventually lead to the symptoms.

5.1 Example

Consider the problem scenario related to heart attack described above. Two different solutions shown in Figure 2, considered acceptable by human experts are merged into the *expert knowledge base*. Thus the student hypothesis shown in Figure 3 (a) is accepted by the system, since this hypothesis is justified through the merging of the expert solutions shown in Figure 2.

For illustration of the heuristic method of accepting inferred links, consider the causal link from an expert solution, shown in Figure 3 (b). The expert knowledge base contains this causal link leading from *hypoinsulinism* to *glucose metabolism disorder*. For this link, two lists are generated. The first list L1 comprises of concepts that have *alike* or *parent* relationship with *hypoinsulinism*, so that L1 = {*hypoinsulinism*, *diseases of endocrine pancreas*}. The second list L2 comprises of concepts that have *alike* or *parent* relationship with *glucose metabolism disorder*, so that L2 = {*glucose metabolism disorders*, *metabolic diseases*, *disorder of carbohydrate metabolism*}.

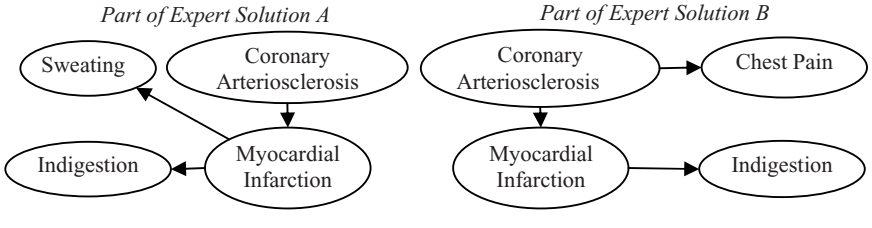


Fig. 2. Expert Solutions merged into the *expert knowledge base*

The space of plausible solutions is expanded, as the system accepts all hypothesis links that are formed through L1 → L2, such as:

hypoinsulinism → *glucose metabolism disorder*
hypoinsulinism → *metabolic diseases*
hypoinsulinism → *disorder of carbohydrate metabolism*
diseases of endocrine pancreas → *glucose metabolism disorder*
diseases of endocrine pancreas → *metabolic diseases*
diseases of endocrine pancreas → *disorder of carbohydrate metabolism*

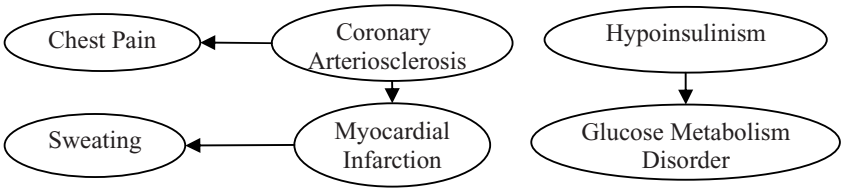


Fig. 3. (a) Student solution accepted by the system (b) Expert Causal Link

6 Results

We conducted system evaluations of the acceptability of causal links beyond those explicitly encoded into the system. We randomly selected 14 causal links from expert solutions to three problem scenarios related to diseases and disorders such as heart attack, diabetes and pneumonia. For each node in the causal link we generated a list of concepts that were found to have *alike* and *parent* relationship with the nodes in the causal link. Thus we generated a pair of lists of concepts for each causal link. We then formed links between each pair of concepts found in the respective lists.

From an initial number of 14 links, the system generated a total of 228 links based on the *alike* and *parent* relationships. These system generated causal links were then presented to a total of 10 medical experts from Thammasat University Medical School, who had at least 5 years of experience in conducting PBL sessions. The experts were asked to rate the acceptability of each link on a scale of 1-5, where 1 implied unacceptable, 2 implied not quite acceptable, 3 implied not sure, 4 implied close to acceptable and 5 implied acceptable. The ratings were so chosen to accommodate the difference of opinion often found among PBL tutors as mentioned in section 3. Figure 4 shows part of the expert evaluation form based on the links generated from the causal link shown in Figure 3 (b).

These 228 links were short listed by medical experts to comprise only of links that could conceivably be formed by medical students according to expert judgment. This resulted in 213 links, which were further short listed to comprise of only those links whose concepts belonged to any of the above mentioned four terminologies: MSH, SNOMEDCT, MTH and NCI. Thus a total of 111 system generated causal links were considered for evaluation. Based on the ratings 1-5 assigned by the medical experts, we computed the average score for each causal link. The overall mean of the ratings came out to be 4.11 with a standard deviation mean of 0.69.

Diabetes case

Causal Links	Acceptability
Hypoinsulinism-->Glucose Metabolism Disorders	5 4 3 2 1
Hypoinsulinism-->Metabolic Diseases	5 4 3 2 1
Hypoinsulinism-->Disorder of carbohydrate metabolism	5 4 3 2 1
Disorder of endocrine pancreas-->Glucose Metabolism Disorders	5 4 3 2 1
Disorder of endocrine pancreas-->Metabolic Diseases	5 4 3 2 1
Disorder of endocrine pancreas-->Disorder of carbohydrate metabolism	5 4 3 2 1

Fig. 4. Part of Expert Evaluation Form showing inferred links

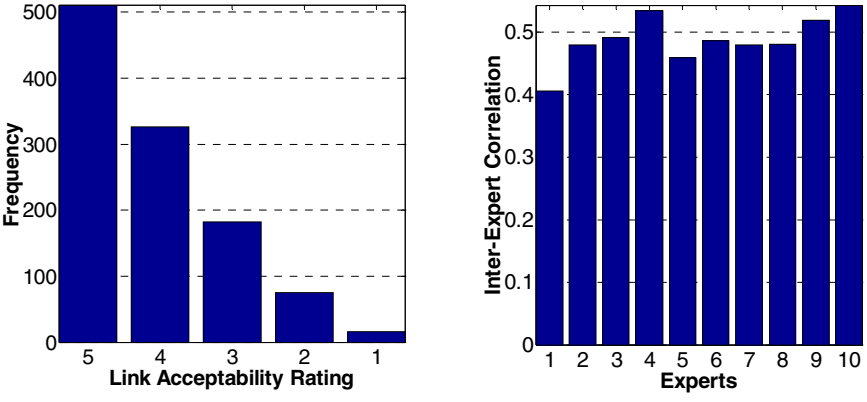


Fig. 5. (a) Distribution of Acceptability Ratings (b) Inter-Expert Correlation

To get a measure of acceptability we collapsed the rating scale to divide the links into acceptable or unacceptable. All links that had an average rating above 3 were considered acceptable, while the rest were considered unacceptable. Based on this criterion, 105 links were found acceptable, whereas 6 links were found unacceptable, leading to an overall accuracy of 94.59 %. The experts were found to agree with each other with a good degree of statistical agreement (Pearson Correlation Coefficient = 0.48, $p < 0.05$). Figure 5 (a) shows the frequency distribution of the 1110 acceptability ratings assigned by 10 experts for 111 samples, whereas Figure 5 (b) shows the correlation values of individual experts with rest of the experts.

7 Discussion

The average score of 4.11 out of 5.0 and the overall accuracy of 94.59 % together with good statistical agreement among expert ratings, indicate that the system is mostly correct in accepting inferred links. As can be observed from the correlation values in Figure 5 (b), the experts were not in perfect agreement with one another over the acceptability of the causal links. Furthermore, the expert evaluations also revealed that in some cases, links inferred by the system scored higher than the corresponding original links created by human experts. For example, the inferred links: *heredity* → *vascular diseases* and *heredity* → *arteriosclerosis* received mean scores

of 4.5 and 4.2 respectively, whereas the original expert link *heredity* → *atherosclerosis* received a mean score of 3.3. This evidence further points to the inherent variation involved in the evaluation of PBL hypothesis links and reinforces the need to have a tutoring system that evaluates student solutions in a broad context.

8 Conclusion

In this paper we have described how to reduce the burden of knowledge acquisition in an intelligent tutoring system. We have described how a broad and easily available medical knowledge source such as UMLS can be deployed as the domain ontology for a tutoring system for medical PBL. We have presented a strategy of making the tutoring system more robust by broadening the scope of solutions that are accepted by the tutoring system as being correct. The inference mechanism for expanding the solution space can also be applied to other domains, where the problem representation is in the form of causal graphs.

We intend to evaluate the effectiveness of the system's tutoring hints vis-à-vis the two major components of hint generation in our system: the measure of partial correctness and the leveraging off of the UMLS concept hierarchy. Finally we intend to conduct evaluations of learning outcomes by assessing the clinical reasoning gains acquired by student users as a result of using this medical tutoring system.

Acknowledgments. We thank the medical experts at Thammasat University Medical School for generously donating their time and effort for the system evaluations.

References

1. Pople Jr., H.E.: Heuristic Methods for Imposing Structure on Ill-Structured Problems: The Structuring of Medical Diagnostics. In: Szolovits, P. (ed.) Artificial Intelligence in Medicine, ch.5. Westview Press, Boulder (1982)
2. Murray, T.: Authoring Intelligent Tutoring Systems: An Analysis of the State of the Art. International Journal of Artificial Intelligence in Education 10, 98–129 (1999)
3. Anderson, J.R., Corbett, A., Koedinger, K., Pelletier, R.: Cognitive tutors: Lessons learned. Journal of the Learning Sciences 4(2), 167–207 (1996)
4. Mitrovic, A.: Experiences in implementing constraint-based modelling in SQL-Tutor. In: 4th International Conference on Intelligent Tutoring Systems, pp. 414–423 (1998)
5. Crowley, R., Medvedeva, O.: An Intelligent Tutoring System for Visual Classification Problem Solving. Artificial Intelligence in Medicine 36(1), 85–117 (2006)
6. Day, M.Y., Lu, C., Yang, J.D., Chiou, G., Ong, C.S., Hsu, W.: Designing an Ontology-Based Intelligent Tutoring Agent with Instant Messaging. In: Fifth IEEE International Conference on Advanced Learning Technologies, pp. 318–320 (2005)
7. Lee, C.H., Seu, J.H., Evens, M.W.: Building an ontology for CIRCSIM tutor. In: 13th Midwest AI and Cognitive Science Society Conference, MAICSS, pp. 161–168 (2002)
8. Suraweera, P., Mitrovic, A., Martin, B.: A Knowledge Acquisition System for Constraint Based Intelligent Tutoring Systems. In: Conference on Artificial Intelligence in Education (2005)
9. U.S. National Library of Medicine, `http://www.nlm.nih.gov/research/umls/`

10. Kumar, A.: Model-Based Reasoning for Domain Modeling in a Web-Based Intelligent Tutoring System to Help Students to Learn to Debug C++ programs. In: 6th International Conference on Intelligent Tutoring Systems (2002)
11. Rubin, S.H., Rush Jr., R.J., Smith, M.H., Murthy, S.N.J., Trajkovic, L.: A Soft Expert System for the Creative Exploration of First Principles of Crystal-Laser Design. In: IEEE International Conference on Systems, Man, and Cybernetics, SMC 2002, Hammamet, Tunisia (October 2002)
12. Mills, B., Evens, M., Freedman, R.: Implementing directed lines of reasoning in an intelligent tutoring system using the atlas planning environment. In: International Conference on Information Technology, pp. 729–733 (2004)
13. Feltovich, P.J., Barrows, H.S.: Issues of generality in medical problem solving. In: Schmidt, H.G., De Volder, M.L. (eds.) Tutorials in problem-based learning: A new direction in teaching the health professions. Van Gorcum, The Netherlands (1984)
14. Hay, P.J., Katsikitis, M.: The 'expert' in problem based and case-based learning: necessary or not? Medical Education 35, 22–26 (2001)
15. Albanese, M.A.: Treading tactfully on tutor turf: Does PBL tutor content expertise make a difference? Medical Education 38, 916–920 (2004)
16. Suebnukarn, S., Haddawy, P.: Modeling individual and collaborative problem-solving in medical problem-based learning. User Modeling and User Adapted Interaction 16(3), 211–248 (2006)
17. Kazi, H., Haddawy, P., Suebnukarn, S.: Enriching Solution Space for Robustness in an Intelligent Tutoring System. In: 15th International Conference on Computers in Education, Hiroshima, Japan (2007)
18. Kazi, H., Haddawy, P., Suebnukarn, S.: Towards Human-Like Robustness in an Intelligent Tutoring System. In: 8th International Conference on Cognitive Modeling, Ann Arbor, Michigan, USA (2007)
19. Achour, S.L., Dojat, M., Rieux, C., Bierling, P., Lepage, E.: A UMLS-Based Knowledge Acquisition Tool for Rule-Based Clinical Decision Support Systems Development. Journal of the American Medical Informatics Association 8(4), 351–360 (2001)
20. Burgun, A., Bodenreider, O.: Methods for exploring the semantics of the relationships between co-occurring UMLS concepts. MedInfo 10(Pt 1), 171–175 (2001)
21. Mendonca, E.A., Cimino, J.J.: Automated Knowledge Extraction from MEDLINE Citations. In: AMIA 2000 Fall Symposium, pp. 575–579 (2000)
22. Srinivasan, S., Rindflesch, T.C., Hole, W.T., Aronson, A.R., Mork, J.G.: Finding UMLS Metathesaurus concepts in MEDLINE. In: AMIA Symposium, 2002, pp. 727–731 (2002)
23. Martens, A., Bernauer, J., Illmann, T., Seitz, A.: Docs 'n Drugs – The Virtual Polyclinic: An Intelligent Tutoring System for Web-Based and Case-Oriented Training in Medicine. In: AMIA Fall Symposium (2001)

Using Optimally Selected Drill Practice to Train Basic Facts

Philip I. Pavlik Jr.[1], Thomas Bolster[1], Sue-mei Wu[2], Kenneth R. Koedinger[1], and Brian MacWhinney[3]

[1] Human Computer Interaction Institute
[2] Deparment of Modern Languages
[3] Department of Psychology, Carnegie Mellon Univeristy, Pittsurgh, PA 15213
{ppavlik,tib,suemei}@andrew.cmu.edu, {keodinger,macw}@cmu.edu

Abstract. How to best sequence instruction in a collection of basic facts is a problem often faced by intelligent tutoring systems. To solve this problem, the following work details two tests of a system to provide drill practice (test trials with feedback) for foreign language vocabulary learning using a practice schedule determined to be optimal according to a cognitive model. In the first test, students chose between an optimized version and a version that merely cycled the vocabulary items. Examination of the time on task data revealed a preference for practice based on the decisions of the cognitive model. In the second test, the system was used to train the component parts of Chinese characters and measure the transfer of knowledge to subsequent learning of Chinese characters. Chinese character learning was improved for students with the relevant optimized training.

Keywords: computer assisted instruction, practice scheduling, prerequisites.

1 Introduction

Because many domains rely on basic facts, this paper addresses a general method for how deficits in basic facts can be addressed through efficient scheduling of practice. To illustrate, consider the case of vocabulary in foreign language learning. The importance of vocabulary knowledge to success in foreign language learning is emphasized by experts in foreign language instruction [1]. However, students are not always motivated to spend time in the repetitive exercises necessary to produce fast and accurate recall of vocabulary items in a language they are learning. Indeed, while taking advantage of the spacing effect (the advantage to long-term learning when practice is distributed in time) is recommended by many authorities [e.g. 2], the high numbers of errors produced during learning that uses spaced repetition might further reduce motivation for extended practice thus making spaced practice a difficult method to apply [3].

To address this dilemma, we have been developing a system that delivers practice of facts scheduled according to the predictions of a cognitive model. This optimal practice scheduling algorithm provides more efficient practice in the laboratory [4] and this paper reports two classroom experiments with the system in a college level Chinese I language class.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 593–602, 2008.

2 Practice Model and Optimization Algorithm

We will begin by describing the ACT-R (Adaptive Control of Thought – Rational) model variant that we use to make scheduling decisions [5]. While ACT-R is best known for its production rule system that models the flow of procedural execution, the model in this paper uses the ACT-R memory equations to predict performance based on a history of learning events.

2.1 ACT-R Variant Practice Model

The model characterizes the strength of an item in memory (vocabulary pairs in the experiments in this paper) by a quantity referred to as "activation". Activation is a continuous real valued quantity specified by Equation 1.

$$m_n(t_{1..n}) = \beta_s + \ln\left(\sum_{i=1}^{n} b_i t_i^{-d_i}\right) \tag{1}$$

In equation 1, n is the number of prior practices for an item for which activation is being computed. The t_i values are the ages (in seconds) for each prior practice of the item by a learner. The summation of these t_i values captures the benefit of frequency of practice while the power function decay parameter (d) models how more recent practice contribute more to the summation. The b value multiplied by each t_i captures the influence of the result of each practice in later predictions. In the case where the past practice was a successful recall, b is typically high, whereas in the case of a failure, b is typically low. The β_s value is initially set at 0 for each student and is estimated during learning to dynamically improve the overall fit of the model for individual differences. These incremental adjustments occur every 50 trials. The d values in the superscript are computed according to Equation 2.

$$d_i = ce^{m_{i-1}} + a \tag{2}$$

In Equation 2 a and c are fitted parameters and m_{i-1} is equal to the activation at the time practice i originally occurred. For example, if we are computing d_7 (decay value for the 7th practice drill), we need to know the activation at the time this drill occurred m_6. Keep in mind that this is recursive since to compute m_6 we need to know ds 1 thru 6. (Since m_0 = -infinity, $d_1 = a$, according to Equation 2.) Equation 2 captures the spacing effect. It represents practice being forgotten more quickly when an item is easier due to narrow spacing of prior practices.

For activation, it has also proven necessary to scale the times between sessions as if it passes more slowly than time within practice sessions [5]. So, if a practice occurred during a previous session its t_i is modified by subtracting a fraction of the intersession from the actual t_i value. For example, if t_6 occurred 100000s ago and 99000s of this period was spent outside the practice sessions, the modifier (0.00046 for the experiments here) is multiplied by the inter-session time and the result (45.5s) is added to the within session practice duration (1000s). For this example t_6 is computed to be 1045.5s according to this procedure. Theoretically this mechanism captures the idea that memory interference (from other items in the set of items being

learned) may be much less intense when practice is not occurring as compared to when it is. Further, at least in the case of classroom experiments where students can be expected to practice the items outside the tutor for classroom work, this parameter also captures some of this learning that occurs without the tutor between sessions. This picking up of classroom learning in this parameter is unintentional and a more principled solution to issue will be sought in future versions.

Equations 3 and 4 are the functions that model recall probability and recall latency as a function of activation. In Equation 3, s captures the variance of the logistic transformation of activation into probability while τ captures the threshold of recall. If s = 0 it implies that activations above threshold result in perfect recall, while activation below threshold means recall always fails. As s increases the transition from 0% to 100% recall becomes increasingly graded. In Equation 4, F scales the latency, which is an exponential function of activation. Equation 4 captures the variable time necessary to perform a correct recall. Fixed costs of each practice are often summed to this variable cost function to capture perceptual motor costs of responding. Like the β_s parameter, F is incrementally adjusted every 40 trials.

$$p(m) = \frac{1}{1 + e^{\frac{\tau - m}{s}}} \tag{3}$$

$$l(m) = Fe^{-m} \tag{4}$$

2.2 Optimized Practice Scheduling

These ACT-R equations model the effect of practice history (including practice spacing, frequency, recency, and success) on both success and latency of later performances. This model allows fine-grained trial by trial predictions of which item of a learning set is optimal to practice next. These predictions are made by computing the long-term efficiency of practice for each item in the set, and then selecting items for practice when they are at a time when their efficiency is maximal. Efficiency is a value computed directly from the model and is equivalent to the long term learning gain divided by the expected time cost of practice for a particular item. Long term learning gains are shown by increase in the "activation" value, which is the strength of an item in memory. Expected time cost depends on activation's effect on latency of recall and on the probability of failure (which results in feedback time). Equation 5 shows the efficiency score equation used for the experiments in this report. The variable r is the retention interval desired for the optimization and is scaled like the t values for the reduced effect of between session forgetting. We set the raw r equal to 30 days, which, scaled by the 0.00046 between session adjustment fixed r at 1191s.

$$eff_m = \frac{p_m b_{suc} r^{-d_m} + (1 - p_m) b_{fail} r^{-d_m}}{p_m (Fe^{-m} + fixedcosts) + (1 - p_m) fixedfailcosts} \tag{5}$$

Figure 1 graphs this function at the parameter values used and shows the inverted u-shaped relationship between efficiency and activation (memory strength). The initial

increase in efficiency, as activation increases, is due to the reduction in failed drill opportunities with higher memory strength. Failed drills consume more time because they require the presentation of feedback for the item, and also because failure itself is typically slower than recall.

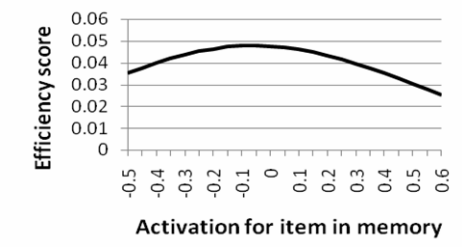


Fig. 1. Graph of the efficiency function for optimized practice

We can also see in Figure 1 that at some point increasing activation no longer provides benefits, but rather leads to less efficient learning. This effect is due to the model of spaced practice, which assumes that the learning from each drill would be more permanent if the drill occurs when activation is less. Each repetition in widely spaced practice reduces activation, as there is more time to forget between repetitions relative to more massed practice. Therefore, spaced practice causes more long-term gain for each trial despite resulting in lower correctness during learning.

Figure 1 illustrates how the interaction of the speed advantage effect for narrow spacing and the long-term learning spacing advantage translate to predict an optimal activation point at which a drill is optimal. The optimal scheduling in the following experiments used an algorithm created using this model to schedule items for practice at this point of maximally efficient learning. To do this, before each trial, the change in the efficiency score as a function of time (the first derivative of efficiency) is computed for every item. Items are selected for immediate practice when the change in the efficiency score approaches 0. If no items have approached 0 either because the derivatives are all positive (in which case more will be learned by waiting for spacing of practice to increase) or because no items have been introduced, the algorithm introduces a new item into the set. After all items have been introduced and the change in efficiency score for all items is positive the algorithm selects the item with the smallest change in efficiency score.

One interesting consequence of Equation 5 is that given a value for r it specifies a specific activation that corresponds to a specific percent correct performance level that should result in maximal efficiency. For Figure 1 this corresponds to -0.04 activation and a percent correct level of 92.6%. However, while the algorithm predicts this specific level at which practice should be optimal, the spacing of practice for each item increases as the learner accumulates practice. Spacing increases because of the increasing stability of the activation value as practice accumulates. This increasing stability is caused by the power function model of forgetting, which indicates that learning from older practices decays more slowly than recent learning. Figure 1 was computed for the following parameter values: s = .261, τ = -0.7, F = 2.322, b_{suc} = 2.497, *fixed success cost* = 0.63s, *fixed failure cost* = 9s, a = .17, c = 0.21 and

r = 1191s. These parameters were estimated by fitting the model for past classroom experiments not reported here. b_{fail} was estimated to be 1.526 for the first practice of any item with later practices $b = 0$.

3 Experiment 1

The first between-subjects experiment described here compares two computerized practice delivery systems in a Chinese I university level course. The control system drills the course vocabulary by cycling through each unit items in random order. The experimental system uses an identical drill procedure, but cycles through items according to the predictions of the model based algorithm designed to maximize learning. In laboratory studies, this optimized model results in large gains in performance (effect size ≈ 1), but it has proven hard to show similar performance advantages in the classroom. This difficulty seems to be mainly due to the relatively small amount of time for which students are assigned to use the tutor.

As a partial resolution to this problem, the following study compares two conditions using time on task as the dependent variable rather than final performance. By doing this we can avoid the problem of having limited classroom time allocated to either condition by simply looking at the total time students spend in each condition. An advantage for time on task is then taken to be evidence for improved motivation and compliance in that condition.

The two practice systems were made available to students by the professor, and students were asked to complete 15 minutes per unit for approximately 1-2% of their semester grade. The webpage that students used to access the two conditions gave simple instructions that were identical in each system. Further, the webpage randomized the order the two tutors appeared on the page so that one condition would not be selected more frequently because of its position on the page. However, students were not blind to condition since the optimized condition was described as, “Optimized Version -- In this version of the practice software, a model of learning is used to choose which flashcard to give for each trial. The model is set to provide approximately optimal spacing and repetition. You can choose to go through either the full set of flashcards for the class, or any particular unit. Your progress is saved, and you can return to where you left off at any time”, while the flashcard condition was described as, “Flashcard Version -- In this version of the practice software, flashcards are delivered in random order, but drop out of the deck when you get them right. You can choose to go through either the full set of flashcards for the class, or any particular unit. Your progress is saved, and you can return to where you left off at any time”.

Students were free to switch back and forth between systems by saving their data and returning to the webpage to continue with another version. This meant that students were not locked into their choice, but rather could change their preference at any time. Other than the practice scheduling the only difference between the versions was that the optimized version listed immediate and one month recall predictions based on the model while the control version kept track of the remaining pairs to be answered correctly to finish one repetition of the selected learning set.

3.1 Participants, Stimuli and Procedures

We only analyzed data from students that had produced complete data on the pre-quiz and post-quiz. According to this criterion there were 90 participants each of which distributed their practice between the two practice methods.

The practice items in each condition were identical for each between subject condition, however the item set varied from 540 pairs in 8 of the 9 class sections (the sections that included regular classroom meetings) to 555 pairs in 1 section of the 9 (the online only section with limited face to face meetings). Since the online section had a correspondingly small sample size it was aggregated with the classroom sections. There were three vocabulary pairs of stimuli and responses for each semantic item: Chinese sound file → English response, Chinese sound file → pinyin response, Hanzi character → pinyin response. (Hanzi are the Chinese characters, while pinyin is the English character orthography for the Chinese pronunciation.) This indicates there were 180 (540 / 3 pairings) classroom semantic items and 185 online semantic items. Every pairing was modeled by an activation value in the optimized condition.

There were 7 units of practice in the one online sections and 10 units of practice in the eight classroom sections. Order of introduction of items was randomized within unit for each version. Additionally, the optimized condition was set so that no related pairs (where the underlying semantic item was the same) were presented with a spacing of less than 2 intervening items. Further, while the flashcard condition randomized all pairings independently to determine the introduction order, the optimized condition randomized units by the groupings of 3 related pairs for each item. Having items introduced in groups was not an explicit model-based decision, but respects the spirit of the model since the model implies that related items should be spaced narrowly at introduction. In contrast, the standard spacing effect suggests practice should be maximally spaced as items were in the flashcard condition.

Practice was distributed according the algorithm in each condition. In the flashcard control condition practice for a particular unit simply involved randomizing the order of the vocabulary items and presenting them one by one. If an item was responded to correctly it was "discarded" and if an item was responded to incorrectly it was put at the end of the order. This procedure continued for each unit until all items were answered correctly for the unit (at which time the order was rerandomized and practice began again) or until the student quit using the tutor.

The drill procedure used for each trial was identical in both conditions. Each drill presented a stimulus on the left side of the screen and allowed the student 20 seconds to respond. If the response was correct there was a 0.5s presentation of a "check mark" to indicate correctness. If the response was incorrect there was a 3s presentation of the correct stimuli and response. If the response was incorrect but the student provided an answer to another item the system gave a 6s study opportunity of both the pair tested and the pair which the student provided a response for.

Pre- and post-quizzes tested the ability to translate 54 items randomly selected from the full item set for each student. Items did not repeat from pre-quiz to post-quiz.

3.2 Results

The main effect of interest was time spent practicing by students in each condition. Using the raw data there was no significant effect (M = 1.26 hours for optimized practice and

M = 1.12 for flashcard practice). However, these values overestimate the preference for the flashcard version since some students merely allowed the flashcard tutor to run without practicing and this happened more frequently for the flashcard condition. To alleviate this, we choose to only consider practice in a condition if probability correct was greater than 0.1 overall. This conservative analysis filtered out students that allowed the tutor to run without practicing. Using this filtered data there was a preference for the optimized version (M's equal 1.24 and 0.86 for optimized and flashcard conditions, t = 2.37, p = .020, Cohen's d effect size = .25).

While the post-quiz results could not be used to establish whether practice was more effective in one condition, it was possible to determine the correlation of practice in each version with gain in post-quiz scores. To do this we computed the correlation of 8 measures of practice in the tutor with the improvement in performance from pre-quiz to post-quiz. These measures included for each condition: total time (raw value), total time filtered to remove p(success) < 0.1 values, total count of correct responses, and probability correct during practice. Only 2 correlations were significant. First, the count of correct responses in the optimized condition correlated with pre-test to post-test gain, r = 0.421 (p = 0.000036). Second, the raw time spent in the flashcard condition was negative correlated with r = -0.366 (p = 0.00121) pre-test to post-test gain. This negative correlation was driven by the few subjects that used the flashcard condition but did not attempt to respond to the drills as discussed above.

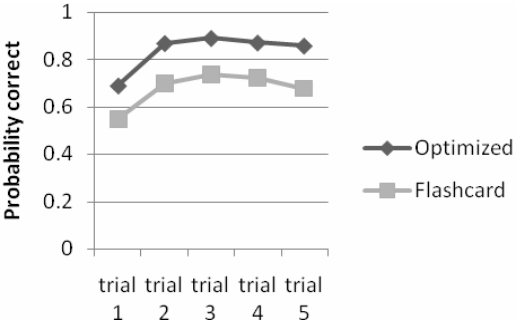


Fig. 2. Average learning curves across the first 5 practices for items in either condition

Figure 2 (created from the raw data) helps us understand why the preference occurred. The figure illustrates the average correctness for items in each condition as a function of the repetition. As the figure illustrates, students found practice in the optimized condition to be easier due to the narrower scheduling used by the optimization condition. In contrast, the lower performance for the flashcard condition showed it was more difficult, which we also take to be an effect of scheduling. Curiously, we also see an advantage for the first drill of practice when the algorithm was simply introducing the item. This benefit is different than the optimized scheduling benefit and was probably due to the procedure of randomizing the related pairings in groups of three and introducing them relatively close together (minimum of 2 intervening trials) in the schedule. However, this grouping would not have significantly affected

later trials because item schedules were dependent on performance after the first trial in the optimized condition.

4 Experiment 2

This between-subjects experiment applied the optimization algorithm to teach learning components rather than to directly train the items that would be tested. To do this experiment we relied on the structure of Chinese characters. Each Chinese character contains one radical item that forms part of the character (sometimes characters have more than one radical, but one radical is always primary). Figure 3 provides an example. As we can note the "see" radical forms part of the verb for "to think". This experiment tested the idea that learning these radical components would improve learning of the characters. While this particular benefit has been found before [6], and theory of part-whole transfer makes it seem likely it can be reproduced [7], we thought it would be interesting to explore this paradigm as a prototype for one type application of the optimal training of basic facts. So, while no single part of this experiment is novel, it remains an important test because it shows real world applicability by bringing a part whole training paradigm into the classroom using a design that measures long term effects on future learning rates.

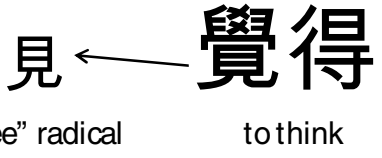


Fig. 3. Example of a Hanzi character and a constituent radical it contains

This experiment was run concurrently with Experiment 1; however, this study independently randomly assigned subjects into either an experimental "radical" training condition or a control "Hanzi" training condition. Both conditions used optimally scheduled practice. The hypothesis was that the "radical' components learned in the experimental condition would produce better learning of Hanzi characters. In contrast, practicing in the Hanzi condition should not improve learning since assessment used a different set of Hanzi than was used in practice. (The Hanzi control condition was intended to rule out the possibility that experience with the software by itself might cause any effects found.) The students were asked to complete one hour practice in the condition they were placed; however, some students practice more or less than that amount.

4.1 Participants, Stimuli and Procedures

We only analyzed data from students that had produced complete data on the pre-quiz and post-quiz. According to this criterion there were 94 participants, 46 of which were randomized into the radical condition and 48 of which were randomized into the Hanzi condition.

The radical set used for training was provided by the Chinese professor and included 154 radical pairs that were identified as components of the Hanzi characters students had to learn for Chinese I. Since some radicals appeared in multiple characters, we choose to introduce the radicals in the order from most frequent to least frequent. For the Hanzi control condition the practice set corresponded to the Hanzi characters for the first three units of the course. In the classroom version of the experiment, the Hanzi set contained 90 practice items, while the online only version contained 106. In both radical and Hanzi practice conditions, the practice items were both radical/Hanzi → pinyin trials and radical/Hanzi → English trials. Thus, for example, there were 77 radicals trained, each of which appeared in 2 pairings.

For the pre and post-quizzes, we tested randomly selected Hanzi items from the last 3 units of the course. Since the post-quizzes occurred at mid semester, this meant that the items were unfamiliar and unlearned for the majority of students. Both pre-quizzes and post-quizzes had the same structure, with 27 items tested each with 2 drill trials. Items did not repeat from pre-quiz to post-quiz. The goal of the quizzes was to produce a learning rate score for each quiz that was a measure of the average correctness on the second drill of a character minus the average correctness of a first drill. For example, on a pre-quiz, a student might get only 1 of 27 items correct for the first drills on the pre-quiz and then get 10 of 27 items correct for the second drills. This would indicate a 33% learning rate for the pre-quiz for this student.

4.2 Results

We were interested in comparing learning rates from the pre-quiz and post-quiz to see if there was an advantage for the post-quiz learning rate as compared to the pre-quiz learning rate. First we ran an ANOVA that compared the gain in learning rate from pre-quiz to post-quiz using the pre-quiz learning rate result as the covariate. This result showed the significant advantage for radical training ($F(1, 91) = 5.62$, $p = 0.020$, $d = 0.49$). The mean gain in learning rate was 12.8% in the radical condition and 6.6% in the Hanzi practice condition. Raw pre-quiz learning rates were 28.8% for radical training and 27.2 for Hanzi training. Raw post-quiz learning rates were 41.6% for radical training and 33.8% for Hanzi training.

Additionally, we were interested in whether this more accurate learning also translated to faster performance on the post-quiz. To look for this we compared the reduction in time for the post-quiz compared to the pre-quiz using the pre-quiz duration as a covariate. This result showed a significant benefit for the radical condition ($F(1, 91) = 4.04$, $p = 0.048$, $d = 0.42$). The mean time saved on the post-quiz was 46.1s for the radical condition and 17.6s for the Hanzi condition. These values are considerable since the average completion time on the pre-quiz was only 394 seconds.

Finally, we wanted to make sure that the result was not driven merely by better time on task in the radical condition. The difference for total practice time was not significant, nor was it significant when pre-quiz duration was used as a covariate ($M = 3771$s for the optimized condition and $M=3428$s for the flashcard condition).

5 Discussion

To address the importance of these basic facts, this paper described a theoretically based, algorithmic method of scheduling performance for such basic facts. This

method takes advantage of well know properties of memory such as the benefits of recency, frequency and spacing to optimize the efficiency of such fact learning. We tested this method of practice in 2 experiments. In Experiment 1 we were able to show that students tend to use the optimized practice more often when given the opportunity to choose an alternative more conventional schedule of practice. This result suggests that students either found the system more effective or more enjoyable. The higher level of correct performance for the optimization condition shown in Figure 2 may be the one reason why people prefer the optimized practice. Experiment 2 focused on efficacy, showing that learning using this method may automatically transfer to new contexts in a naturalistic classroom setting.

Acknowledgments. This research was supported by NIMH R01 MH 68234 and a grant from Ronald Zdrojkowski for educational research.

References

1. Nation, I.S.P.: Learning vocabulary in another language. Cambridge University Press, Cambridge (2001)
2. Dempster, F.N.: The spacing effect: A case study in the failure to apply the results of psychological research. American Psychologist 43, 627–634 (1988)
3. Rea, C.P., Modigliani, V.: The effect of expanded versus massed practice on the retention of multiplication facts and spelling lists. Human Learning: Journal of Practical Research & Applications 4, 11–18 (1985)
4. Pavlik Jr., P.I., Anderson, J.R.: Using a model to compute the optimal schedule of practice. Journal of Experimental Psychology: Applied (accepted)
5. Pavlik Jr., P.I., Anderson, J.R.: Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. Cognitive Science 29, 559–586 (2005)
6. Taft, M., Chung, K.: Using radicals in teaching Chinese characters to second language learners. Psychologia 42, 243–251 (1999)
7. Singley, M.K., Anderson, J.R.: The transfer of cognitive skill. Harvard University Press, Cambridge (1989)

Eliminating the Gap between the High and Low Students through Meta-cognitive Strategy Instruction

Min Chi and Kurt VanLehn

Learning Research and Development Center & Intelligent System Program
University of Pittsburgh, PA, 15260
{mic31,vanlehn+}@pitt.edu

Abstract. One important goal of Intelligent Tutoring Systems (ITSs) is to bring students up to the same level of mastery. We showed that an ITS teaching a domain-independent problem-solving strategy indeed closed the gap between High and Low learners, not only in the domain where it was taught (probability) but also in a second domain where it was not taught (physics). The strategy includes two main components: one is solving problems via Backward-Chaining (BC) from goals to givens, named the BC-strategy, and the other is drawing students' attention on the characteristics of each individual domain principle, named the principle-emphasis skill. Evidence suggests that the Low learners transferred the principle-emphasis skill to physics while the High learners seemingly already had such skill and thus mainly transferred the other skill, the BC-strategy. Surprisingly, the former learned just as effectively as the latter in physics. We concluded that the effective element of the taught strategy seemed not to be the BC-Strategy, but the principle-emphasis skill.

Keywords: Intelligent Tutoring Systems, meta-cognitive skills, domain-independent Problem-Solving Strategies.

1 Introduction

Bloom [2] argued that human tutors not only raised the mean of scores, but also decrease the standard deviation of scores. That is, students generally start with a wide distribution in test scores; but as they are tutored, the distribution becomes narrower—the students on the low end of the distribution begin to catch up with those on the high end. Another way to measure the same phenomenon is to split students into High and Low groups based on their incoming competence. One then measures the learning gains of both groups. According to Bloom, a good tutor should exhibit an aptitude-treatment interaction: both groups should learn, and yet the learning gains of the Low students should be so much greater than the High ones' that their performance in the post-test ties with the High ones. That is, one benefit of tutoring is to narrow or even eliminate the gap between High and Low.

Previously, we found that Pyrenees [11], an ITS that explicitly taught a problem-solving strategy, was more effective than Andes [12], an ITS that did not explicitly teach any strategy not only in the domain where it was used, but in a second domain where it was not used [3]. The strategy seemed to have lived up to our expectations

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 603–613, 2008.

and transferred from one domain to another. In this paper, we investigated whether explicit strategy instruction exhibited an aptitude-treatment interaction, that is, whether it narrows or even eliminates the gap between High and Low; moreover, whether both High and Low indeed transferred the strategy to the second domain.

2 Background

A task domain is deductive if solving a problem requires producing an argument, proof or derivation consisting of one or more inference steps, and each step is the result of applying a domain principle, operator or rule. Deductive domains are common parts of math and science courses. Two common problem-solving strategies in deductive domains are forward chaining (FC) and backward chaining (BC) [7]. In FC, reasoning proceeds from givens toward goals; while in BC, it works backward from goals to givens. FC and BC have been widely used in computer science; however, they are rarely observed in a pure form in natural human problem solving. Early studies suggested that novices used BC and experts used FC [5], but later studies showed that both used fairly similar mixtures [6]. It appears that most human solvers use a mixture of strategies, heuristics, and analogies with past solutions as well as other general knowledge. Although human solvers don't seem to use FC and BC in their pure form, the strategies' success in guiding computer problem solvers suggests that teaching human solvers to use FC or BC might improve their problem-solving performance. Several ITS-based studies were conducted to test this hypothesis.

Trafton and Reiser [10] tested the benefits of explicit strategy instruction on an ITS called Graphical Instruction in Lisp. Three forms of instruction were compared: FC-only, BC-only or freely. After 13 training problems were completed in less than one hour, all three groups achieved the same learning gains. Scheines and Sieg [8] gave students over 100 training problems in sentential logic and they found students who were taught and required to use FC or BC learned just as effective as those who were not taught any strategy. VanLehn et al. [10] compared two ITSs that teach introductory college physics. One system explicitly taught students a version of BC; while the other did not teach or require students to follow any explicit strategy. Although some outcome measures differed between groups, overall performance on the post-test was quite poor, suggesting a floor effect.

In summary, most previous studies were conducted in a single domain and contrasted students who were taught a strategy and those who were not. In this paper, we investigated the impact of explicit strategy instruction on eliminating the gap between High and Low across two unrelated domains and two different ITSs. The problem-solving strategy chosen is the Target Variable Strategy (TVS) [11], a domain-independent BC strategy, and the two selected domains were probability and physics. Probability covered 10 major principles in Axiom of Probability and Conditional Probability; and physics covered 10 principles in Work and Energy. During probability instruction, the Experimental students were trained on an ITS, Pyrenees, that explicitly taught the TVS; while the Control students were trained on another ITS, Andes, without explicit strategy instruction. During subsequent physics instruction, both groups were trained on the same ITS, which did not teach any strategy. On both probability and physics post-tests, we expect:

High-Experimental = Low-Experimental = High -Control > Low-Control.

That is, for both task domains, the Low students should catch up with the High students, but only if they were taught the TVS.

3 Methods

3.1 Participants

Participants were 44 college students who received payment for their participation. They were required to have a basic understanding of high-school algebra, but not to have taken college-level statistics or physics courses. Students were randomly assigned to the two conditions. Two students were eliminated: one for a perfect score on the probability pre-test and one for deliberately wasting time.

3.2 Three ITSs

The three ITSs involved in this study were Pyrenees, Andes-probability, and Andes-physics respectively. The first two taught probability while the third taught physics. Apart from domain knowledge, Andes-probability and Andes-physics were the same and we use 'Andes' to refer to both. Pyrenees required students to follow the TVS while Andes did not require students to follow any explicit problem-solving strategy. Next, we will compare Pyrenees and Andes from the perspectives of both the user interface and students' behaviors.

User Interfaces Perspectives: Both Pyrenees and Andes provide a multi-paned screen that consists of a problem-statement window, a variable window for listing defined variables, an equation window, and a dialog window. The tutor-student interactions are quite different for each system.

Pyrenees is a restrictive tutor-initiative ITS. It guides students in applying the TVS by prompting them to take each step as dictated by the strategy. For example, when the TVS determines that it is time to define a variable, Pyrenees will pop up a tool for that purpose. Thus the tutor-student interactions in Pyrenees take the form of a turn-taking dialogue, where the tutor's turns end with a prompt or question to which the student must reply and all interactions only takes place in the dialogue window. Andes, on the other hand, is a nonrestrictive mixed-initiative ITS. Students use GUI tools to construct and manipulate a solution, so the interaction is event-driven. Students may edit or interact with any of the four windows: by drawing vectors in vector window, writing or editing equations in the equation window, and so on. Once an entry or edit has been made successfully, Andes provides no further prompt to make the next step.

Interactive Behaviors Perspectives: Both Andes and Pyrenees provide immediate feedback. However, their standard of correctness differs. Andes considers an entry correct if it is true, regardless of whether it is *useful* for solving the problem; on Pyrenees, however, an entry is considered correct if it is true and strategically acceptable to the TVS. Moreover, students can enter an equation that is the algebraic combination of several principle applications on Andes but not on Pyrenees because the TVS requires students to apply one principle at a time.

Both systems provide hints when students asked. When an entry is incorrect, students can either fix it independently, or ask for *what's-wrong help*. When they do not know what to do next, they can ask for *next-step help*. Both *next-step help* and *what's-wrong help* are provided via a sequence of hints that gradually increase in specificity. The last hint in the sequence, called the *bottom-out hint*, tells the student exactly what to do. Pyrenees and Andes give the same *what's-wrong help* for any given entry, but their next-step help differs. Because Pyrenees requires students to follow the TVS, it knows what step they should be doing next so it gives specific hints. In Andes, however, students can always enter any correct step, so Andes does not attempt to determine their problem-solving plans. Instead, it asks students what principle they are working on. If students indicate a principle that is part of a solution to the problem, Andes hints an uncompleted step from the principle application. If no acceptable principle is chosen, Andes picks an unapplied principle from the solution that they are most likely to be working on.

3.3 Procedure

The study had 4 main parts: background survey, probability instruction, Andes Interface training, and physics instruction (shown in the left column of Table 1). All materials were online. The background survey asked for High school GPA, SAT scores, experience with algebra and other information.

Table 1. Experiment Procedure

Part	Experimental	Control
Survey	Background survey	
Probability Instruction	Pre-training	
	Pre-test	
	Training on Pyrenees	Training on Andes-Probability
	Post-test	
Andes Interface Training	Solve a probability problem on Andes-Probability	
Physics Instruction	Pre-training	
	Pre-test	
	Training on Andes-Physics	
	Post-test	

The probability and physics instruction each consisted of four phases: 1) Pre-training, 2) Pre-test,3) Training on the ITS, and 4) Post-test. During pre-training, students studied domain principles. For each principle, they read a text description, reviewed some worked examples, and solved some single-principle and multiple-principle problems. After solving a problem, their answer was marked correct or incorrect, and the expert's solution was also displayed. The students then took the pretests. All students took the same pre- and post-tests. All test problems were

open-ended and required students to derive answers by writing and solving one or more equations. In phase 3, students in both conditions solved the same problems in the same order, albeit on different ITSs. Each of the domain principles was applied at least twice in both trainings. The Experimental group learned probability in Pyrenees and physics in Andes-physics while the Control group learned both domains in Andes. Students could access the domain textbook at any time during the training. Finally, students took the post-tests. On each post-test, 5 problems were isomorphic to a training problem in phase 3. There were also 5 novel, non-isomorphic multiple-principle problems on the probability post-test and 8 on the physics post-test.

Only the Experimental students took the third part, Andes Interface Training. Its purpose was to familiarize them with the Andes GUI without introducing any new domain knowledge. The problem used was one of the twelve probability training problems that they had previously solved on Pyrenees. Pilot studies showed that one problem was sufficient for most students to become familiar with Andes GUI.

To summarize, the procedural differences between the two conditions were: 1) during the probability training, the Experimental condition trained on Pyrenees while the Control condition trained on Andes-probability; 2) the Experimental students learned how to use Andes' GUI before physics instruction.

3.4 Grading Criteria

We used two scoring rubrics: binary and partial credit. Under binary, a solution is worth 1 point if it was completely correct or 0 if not. Under partial credit, each problem score is a proportion of correct principle applications evident in the solution. If they correctly apply 4 of 5 possible principles they would get a score of 0.8. Solutions were scored by a single grader blind to condition.

4 Results

In order to measure aptitude-treatment interaction, we needed to define High and Low groups based on some measure of incoming competence. We chose to use MSAT scores because probability and physics are both math-like domains. Our split point was 640, which divide into: High (n = 20) and Low (n = 22). Except for the MSAT scores and High school GPA, no significant difference was found between High and Low on other background information such as age, gender, VSAT scores and so on. As expected, the High group out-performed the low group during the probability pre-training and the probability pre-test under the binary scoring rubric: $t(40)= 3.15$, $p= 0.003$, $d= 0.96$, $t(40)= 2.15$, $p= 0.038$, $d= 0.66$, and $t(40)= 2.27$, $p <0.03$, $d=0.70$ on single-principle, multiple-principle problems during probability pre-training and overall in probability pre-test respectively . The same pattern was found under partial rubric in the probability pretest. Thus, the MSAT score successfully predicted the incoming competence of the students, which justifies using it to define our High vs. Low split.

Incoming competence combined with conditions partitioned the students into four groups: High-Experimental (n = 10), Low-Experimental (n = 10), High-Control (n = 10), and Low-Control (n = 12). Fortunately, random assignment balanced the

Experimental vs. Control conditions for ability, and this balance persisted even with the groups were subdivided into High and Low via MSAT score. On every measure of incoming competence, no significant difference was found between the Experimental and Control groups, the Low-Experimental and Low-Control ones, or the High-Experimental and High-Control ones. These measures were: the background survey, the probability pre-test; probability pre-training scores, the time spent reading the probability textbook, and the time spent solving the pre-training problems. Averaged over all students, the total times for each training phase were: 2.4 hrs and 2.7 hrs for probability pre-training and training; 1.5 hrs and 3.0 hrs for physics pre-training and training respectively. No significant differences were found among the four groups on any of these times.

4.1 Test Scores

Figure 1 shows that the test score results are consistent with our hypothesis: after trained on Pyrenees, the Low-Experimental students scored significantly higher than their Low-Control peers on all three assessments: probability post-test, physics pre-test and physics post-tests: $t(20) = 4.43$, $p < 0.0005$, $d = 1.90$; $t(20) = 3.23$, $p < 0.005$, d = 1.34; and $t(20) = 4.15$, $p < 0.0005$, $d = 1.84$ respectively. More importantly, the Low-Experimental students even seemed to catch up with the High ones: no significantly difference was found among the High Experimental, Low-Experimental, and High-Control on all three assessments even though the two Experimental groups seemed to out-perform the High-Control in Figure 1.

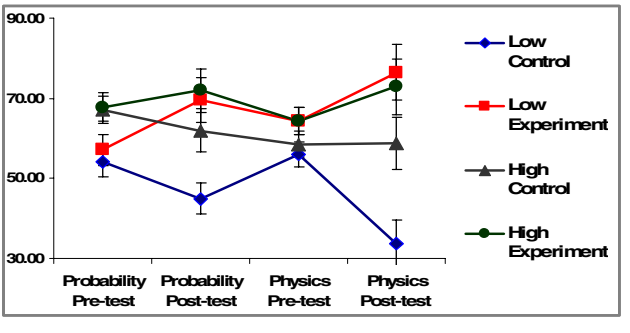


Fig. 1. Compare four groups on four tests (maximum score = 1)

Thus, explicit strategy instruction in probability caused the Low-Experimental group to learn more effectively than the Low-Control group during probability training, physics training and even physics pre-training. They seemed to have caught up to the High ones while the Low-Control ones did not. Moreover, while the High-Experimental group didn't benefit much from the TVS, they were not harmed either.

Dynamic Assessments

While test results are the most common assessment of learning performance, one can also compare students' behaviors as they learn. Such comparisons are called *dynamic*

assessments [2]. In so doing, we can identify students who are effective learners even though their test scores may be equal to or even lower than others. Here we investigated students' interactive behaviors on Andes during physics training, as all students received the identical procedure during that period.

Frequency of help requests: Andes-Physics logs every user's interface action performed, including help requests, tool usage, and equation entries. We first tried to characterize the overall difference in students' solutions via the amount of help they requested. On each of 8 physics training problems, the Low-Experimental students made *significantly fewer* next-steps help requests than the Low-Control ones. No significant difference was found among the Low-Experimental, the High-Experimental and High-Control groups. This suggests that the Low-Experimental may have transferred the TVS. However, there are other possible explanations, so we conducted several other analyses.

Triage of Logs: Solution logs were grouped into 3 categories: smooth, help-abuse, and rocky:

Smooth solutions included no help requests, except on problems that required more than eight principle applications. There students were permitted up to two *what's-wrong help* requests.

Help-abuse solutions are produced when every entry was derived from one or more *next-step helps*.

Otherwise, the solution was categorized as *Rocky* because students appeared capable of solving part of the problem on their own, but needed help on the rest.

Figure 2 shows there was a significant difference among four groups on the distribution of the three types of solutions. While no significant difference was found between the High-Experimental and Low-Experimental, there was a significant difference between the Low-Experimental and the High-Control: $\chi^2(2) = 11.74$, $p(\chi^2) < 0.003$; and between the High-Experimental and High-Control: $\chi^2(2) = 9.06$, $p(\chi^2) < 0.01$. Qualitatively, the results appear to be: High-Experimental = Low-Experimental > High-Control > Low-Control.

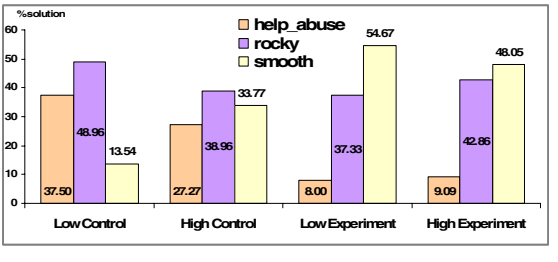


Fig. 2. Solution Percentage by Type

For a more quantitative measure, we used a smaller unit of analysis: individual equations. We coded each correct equation entry in the solution logs with 3 features:

Relevance: The equation was labeled relevant or irrelevant based on whether it contributed to the problem solution.

Help: The equation was labeled "Help" if it was entered after the student asked for help from Andes-physics. Otherwise, it was labeled "No-help".

Content: The equation's content was coded as either "a correct equation with new physics content" or "others".

We sought to find out how frequently students made progress toward solving a problem without asking for any help from Andes. In terms of the three-feature coding mentioned above, such a "desirable" equation would be coded as "Relevant", "No-help", and "Correct equation with new physics content". We called them *desirable* steps and defined the desirable steps ratio DSR:

$$DSR = \frac{Desirable\ \ Steps\ in\ the\ solution}{All\ \ Steps\ in\ the\ solution}$$

$$DSR = \frac{Desirable\ \ Steps\ in\ the\ solution}{All\ \ Steps\ in\ the\ solution}$$

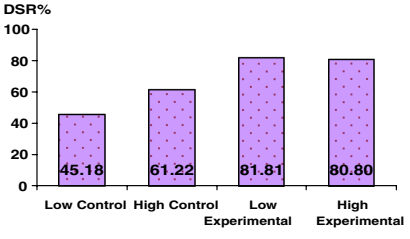


Fig. 3. DSR on overall solutions

As shown in Figure 3, the Low-Experimental had significantly Higher DSR than the Low-Control: $t(169)= 7.50$, $p<0.0001$. In fact, the former even made significantly more progress than the High-Control: $t(150)= 3.84$, $p< 0.001$. While there is a significant difference between the Low-Control and High-Control groups: ($t(171)=2.83$, $p< 0.01$), there is no such difference between the two Experimental groups. In short, this dynamic assessment showed that: High-Experimental = Low-Experimental > High-Control > Low-Control.

To summarize, both test scores and dynamic assessments show that the Low students catch up with the High ones in the Experimental condition but not in the Control condition. On some measures, the Low-Experimental students even surpass the High-Control ones. Next, we'll investigate what was transferred from probability to physics that made the Low-Experimental students so successful?

Transferring the Two Cognitive Skills of the TVS

The TVS is BC problem-solving strategy [11]. That is, it solves problems backwards from goals to givens. However, it differs from pure BC in that it requires students to explicitly identify principles before applying them. As an illustration, Table 2 presents the principle application part of a TVS solution.

Prior work on BC through equations required students to enter the equations alone [1]. Thus, they might only write the equations shown in the middle column of Table 2. Our

TVS strategy, however, also requires them to attend to the application of individual domain principles, as shown in the right column of Table 2. For example, instead of simply entering an equation with one principle application each, students need to pick an unknown variable, select a principle that apply to the unknown variable, define all the variables appearing in its equation if undefined yet, write the equation, and finally remove the "sought" mark and mark new unknown variables. Students were also asked various questions on the characteristics of the principle. For example, in last row in Table 2, after students pick the complement theorem, Pyrenees would ask: "… *To apply the principle, you must have noticed that there are a set of events that are mutually exclusive and collectively exhaustive. What are these events?*" Students should answer: ~(A∩B) and (A∩B). Therefore, the TVS is not only a BC strategy, but it draws students' attention to the characteristics of each individual domain principle, such as when it is applicable.

Table 2. Part of a TVS example solution

Problem: **Given P(A)=1/3, P(B)=1/4, P(A ∩ B)=1/6, find: P(~A ∪ ~B).**		
Step	**Equations**	**Justification**
1	P(~A ∪ ~B)=P(~(A∩B))	To find P(~A ∪ ~B), apply De Morgan's theorem. Delete "sought" from P(~A ∪ ~B) and add "sought" to P(~(A∩B))
2	P(A∩B) + P(~(A∩B))=1	To find P(~(A∩B)), apply the Complement theorem. Delete "sought" from P(~(A∩B))

In short, we argue that the TVS includes two main components: one is to solve problems via BC from goals to givens, named the BC-strategy, and the other is to focus attention to the domain principles, named the principle-emphasis skill. In order to determine the BC-strategy usage, we analyzed students' logs to see whether the order of equations in their solutions follows the BC. For the principle-emphasis skill, we used the single-principle problems as our litmus test because students who had applied the BC-strategy would have no particular advantage on them because solving these single-principle problems only need to apply one principle; while students who had learned the idea of focusing on domains principles should show an advantage on them.

Transfer the BC-Strategy: If students engaged in the BC-strategy, we expect they would apply the BC-strategy when they had difficulties, that is, on rocky solutions. Whereas on smooth solutions, students don't have any difficulties since they may solve problems mainly based on existing schemas [9]. Thus, we subcategorized each desirable step in the logs as BC or non-BC, where non-BC included FC, combined equations, and so on. We then defined BC% as the proportion of desirable steps that were coded as BC. Figure 4 showed that on Rocky solutions the High-Experimental group applied BC significantly more frequently than the other three groups: $t(40)=2.25$, $p=0.03$ while the Low-Experimental group used the BC as frequently as the two Control groups. Thus, apparently it was the High-Experimental group alone who transferred the BC-Strategy to physics.

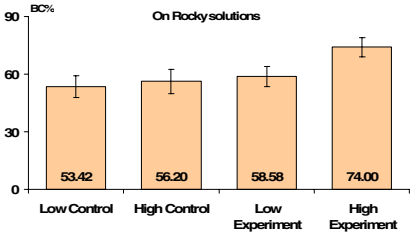


Fig. 4. BC Usage on Rocky Solutions

Transfer of the Principle-Emphasis Skill: The Low-Experimental students scored just as high as the High-Experimental ones even though they used the BC no more frequently than two Control groups. Thus, they must have transferred something else of the TVS. Our hypothesis is that they transferred the principle-emphasis skill. We divided both post-tests into single-principle and multiple-principle problems. Furthermore, we divided the multiple-principle problems into those that were isomorphic to a training problem and those that were not. If the Low-Experimental group applied the principle-emphasis skill, we expected them to out-perform the Low-Control group on all of them in both post-tests. This turned out to be the case (see Figure 5). It suggests that the main effect of teaching the TVS to the Low students was to get them to focus on the domain principles. Further analysis showed no significant difference among the High-Control, the Low-Experimental, and High-Experimental on any types of problems, which indicates that High students may already have such skill.

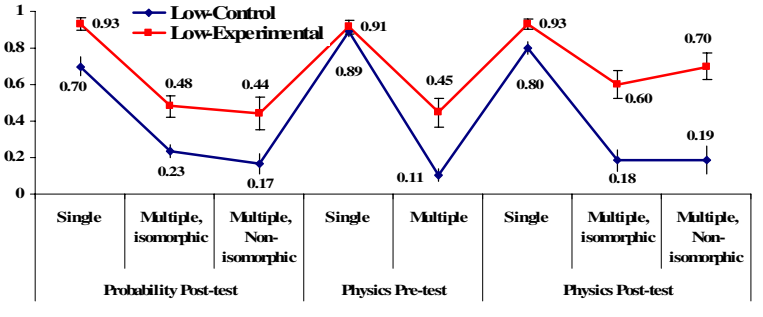


Fig. 5. Scores on Three Types of Problems in Both Tests

5 Conclusions

Overall, we found teaching students the TVS indeed exhibited an aptitude-treatment interaction in deductive domains: the gap between High and Low students in the Experimental Condition seemed to be eliminated in both probability and physics. Although the two Experimental groups performed equally well in both physics pre- and post-tests, the Low-Experimental group transferred the principle-emphasis skill to

physics while the High-Experimental apparently already possessed it and thus they mainly transferred the BC-strategy.

These results suggest that it is *not* the BC-strategy that is most important to teach Low learners. Instead, one should teach the meta-cognitive skill of focusing on individual principle applications. It could be that Low and High learners may have differed initially in that Low students lacked this "how to learn" meta-cognitive knowledge for a principle-based domain like probability or physics. Such results suggest building an ITS that does not teach the TVS explicitly, but instead just teaches to focus on principle applications in deductive domains. Perhaps it would be just as effective as Pyrenees. Indeed, because its students need not learn all the complicated bookkeeping of the BC-strategy, which may cause cognitive overload [9], it might even be more effective than Pyrenees not only for an initial domain where the ITS was used but subsequent domains where it is not used.

References

1. Bhaskar, Simon: Problem solving in semantically rich domains: An example from engineering thermodynamics. Cognitive Science 1, 193–215 (1977)
2. Bloom: The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher 13, 4–16 (1984)
3. Chi, VanLehn: Accelerated future learning via explicit instruction of a problem solving strategy. In: 13th International Conference on AIED, pp. 409–416 (2007)
4. Haywood, Tzuriel: Applications and challenges in dynamic assessment. Peabody Journal of Education 77(2), 40–63 (2002)
5. Larkin, McDermott, Simon, Simon.: Expert and novice performance in solving physics problems. Science 208, 1335–1342 (1980)
6. Priest, Lindsay: New Light On Novice-Expert Differences in Physics Problem-solving. British Journal of Psychology 83, 389–405 (1992)
7. Russell, Norvig: Artificial Intelligence: A Modern Approach, 2nd edn. Prentice-Hall, Upper Saddle River (1995)
8. Scheines, Sieg: Computer environments for proof construction. Interactive Learning Environments 4(2), 159–169 (1994)
9. Sweller: Cognitive Technology: Some Procedure for Facilitating learning and Problem-solving in mathematics and Science. Journal of EdPsych 81(4), 81–84 (1989)
10. Trafton, Reiser: Providing natural representations to facilitate novices' understanding in a new domain: Forward and backward reasoning in programming. In: Proceedings of the 13th Annual Conference of the Cognitive Science Society, pp. 923–927 (1991)
11. VanLehn, et al.: Implicit versus explicit learning of strategies in a non-procedural cognitive skill. In: 7th Conference on ITS, Maceio, Brazil (2004)
12. VanLehn, et al.: The Andes physics tutoring system: Lessons learned. International Journal of Artificial Intelligence and Education 15(3), 147–204 (2005)

Using Hidden Markov Models to Characterize Student Behaviors in Learning-by-Teaching Environments

Hogyeong Jeong[1], Amit Gupta[1], Rod Roscoe[1], John Wagster[1], Gautam Biswas[1], and Daniel Schwartz[2]

[1] Vanderbilt University & [2]Stanford University
Nashville, TN 37235. USA
{hogyeong.jeong,rod.roscoe,john.wagster,
gautam.biswas}@vanderbilt.edu, danls@stanford.edu

Abstract. Using hidden Markov models (HMMs) and traditional behavior analysis, we have examined the effect of metacognitive prompting on students' learning in the context of our computer-based learning-by-teaching environment. This paper discusses our analysis techniques, and presents evidence that HMMs can be used to effectively determine students' pattern of activities. The results indicate clear differences between different interventions, and links between students learning performance and their interactions with the system.

Keywords: Learning by Teaching environments, Metacognition, Behavior Analysis, hidden Markov modeling.

1 Introduction

We have developed exploratory learning environments called teachable agents that use a learning-by-teaching paradigm to promote learning and reasoning skills with middle school science students [1][2]. The students are typically not domain experts and have little knowledge of teaching practices. In these environments, students teach a computer agent called Betty using structured graphical representations called concept maps [3]. Since the concept maps are purported to be representations of Betty's knowledge, the students are teaching Betty and fixing her errors by revising the maps. Of course, the maps are generated by the students based on their own knowledge, thus they are actually representations of the students' own domain understanding (Fig. 1).

The teaching aspects of this environment build upon research showing that students can benefit academically by teaching other students [1][4]. Biswas, Schwartz, & Bransford have reported that students preparing to teach felt that the responsibility to teach encouraged them to gain deeper understanding of the materials [5]. Beyond preparing to teach, actual teaching taps into three critical aspects of learning interactions – *structuring, taking responsibility, and reflecting.* These interactions facilitate self-monitoring and reflective knowledge-building for the teacher [6]. Effective teaching requires the monitoring of how well students understand and use ideas. Tutors and teachers often reflect on their interactions with students during and after the teaching process in order to better prepare for future sessions [7][8].

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 614–625, 2008.

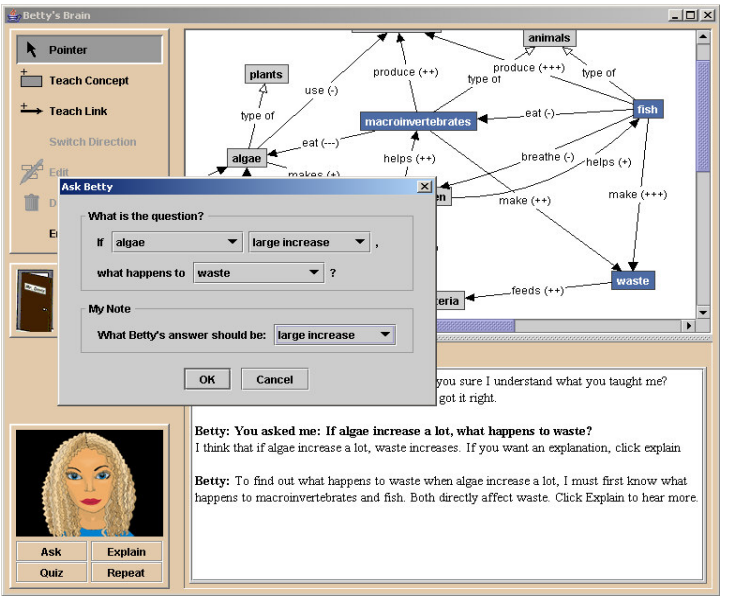


Fig. 1. Betty's Brain system with Query Window

The visual concept map structure also helps students make concepts and relationships explicit, which supports self-monitoring and knowledge organization [3]. The concept mapping also occurs in a context where the students can query Betty, ask her to explain her reasoning, and assess her knowledge by having her to take quizzes. For these reasons, we have hypothesized that working with Betty can help students to better understand science concepts, and engage in productive learning strategies that promote metacognition, organization, and reasoning with causal knowledge

Our previous work has focused on students' learning as measured by the quality of their concept maps. We found that learning-by-teaching with metacognitive support helped students learn about river ecosystems, and also better prepared them for future learning on related topics [1][9]. We compared several versions of the learning by teaching environment with a non-teaching version. Students who taught Betty developed more complete and interconnected concept maps than students who created maps for themselves (i.e., these students made concept maps and received feedback from the system on the quality of the map, but there was no cover story of teaching an agent). Learning outcomes were strongest for students who also received metacognitive feedback from Betty, in which she exhibited self-regulated learning behaviors that the student teacher could appropriate to improve their own learning. These differences persisted during a transfer phase in which students learned about a new domain and taught Betty in the absence of most feedback and prompts.

We have recently turned our attention to analyses of students' behaviors as they teach Betty and create concept maps. Such analyses are important because they shed light on students' choices of interactive behaviors that influence learning, and the strategies they bring to the learning task [4]. Preliminary analyses of prior data

showed that the quality of students' concept maps was paralleled by patterns in their behaviors [2]. These results suggest that self-regulated learning prompts and feedback from Betty helped student teachers engage in productive learning interactions.

In this paper, we discuss data from a new study testing the benefits of the Betty's Brain system. In particular, we present a refined methodology for exploring students' strategies using hidden Markov models (HMMs) to capture students' behaviors as they use the system [10]. Ours is a specific implementation of the generic process outlined by Fisher and Sanderson [11]. We discuss our methods for extracting the student interaction patterns from system log files, and describe our procedure for deriving and interpreting the HMMs of student behaviors. We then compare the HMMs across three experimental conditions in the main and transfer phases. We believe that this approach has merit for analyzing student behaviors for several reasons. First, HMMs allow us to go beyond frequency counts or proportions of individual behaviors, instead examining how these behaviors cohere in larger patterns or strategies. Similarly, this approach takes into account the entire sample of students' behaviors, rather than focusing only on specific behaviors or moments in time. The holistic nature of our analysis may provide a useful global view of how students approach the learning task.

2 Experimental Design and System Features

Our participants were 56 students in two 5th grade science classrooms, taught by the same teacher. Students were assigned to one of three conditions using stratified random assignment based on standardized test scores. The conditions varied on the type of scaffolding provided by the mentor agent and/or the Betty agent. The students first created concept maps on river ecosystems during the main phase (seven 45-minute sessions). After an eight-week delay, students participated in the transfer phase (five 45-minute sessions) in which they learned about a new domain, the land-based nitrogen cycle. All students used an identical system during the transfer phase.

The three versions of the system were: (i) a learning by teaching (LBT) version in which students taught Betty, (ii) a self-regulated learning by teaching (SRL) version in which students taught Betty and received metacognitive prompts from Betty, and (iii) an intelligent coaching system (ICS) version in which students created a map for themselves with guidance from the mentor agent.

Students' interactions with Betty include three main activities: "teaching" by generating the concept map; "querying" by using a template to ask Betty questions; and "quizzing" Betty by asking set of predefined questions that have been "assigned" by the mentor agent. Betty answers questions using qualitative reasoning methods [1] to follow chains of links to determine how changes in one concept affect other concepts. After asking Betty a question, students can ask Betty to explain her reasoning steps.

The ICS version was our control condition. Students constructed a concept map to answer three sets of quiz questions. These students had access to the same teach, query, and quiz functions, but they were not presented in terms of teaching Betty. Students directly edited and queried their own maps. When students submitted their maps for the quizzes, Mr. Davis, the mentor agent, provided corrective feedback in the form of hints on how to correct errors [1]. The LBT group received the same corrective feedback from Mr. Davis after Betty took a quiz. The feedback to the SRL

students focused on higher level concepts (e.g., read about the food chain or the waste cycle) and suggestions on how they could become better learners and teachers.

For the transfer phase, all students used a stripped down version of the LBT system with no feedback provided by Betty or the mentor. Students could still check their maps by asking questions or submitting them for a quiz.

2.1 Metacognitive Support in Betty's Brain

An important part of our system is the self-regulated learning support provided to the students. Self-regulated learning theory describes a set of comprehensive skills such as setting learning goals, selecting appropriate strategies, monitoring one's learning progress, and revising one's knowledge and strategies as necessary [11][12].

Table 1. Some Interactive Action Patterns and Betty's responses

Regulation Goal	Pattern Description	Betty Response
MONITORING BY ASKING QUERIES	**Successive quiz requests but no queries asked of Betty in between quizzes**	I'm still unsure of this material and I would like to do well. Mr. Davis said "take the quiz only if you think you will do well." *(Betty refuses to take quiz)*
MONITORING THROUGH EXPLANATIONS	**Multiple requests for Betty to give an answer but no request for explanation**	Let's see, you have asked me a lot of questions, but you have not asked for my explanations lately. Please make me explain my answers so you will know if I really understand.
TRACKING PROGRESS	**The most recent quiz score is significantly worse than the previous score**	I would really like to do better. Please check the resources, teach me, and make sure I understand by asking me questions that are on the quiz. My explanation will help you find out why I am making mistakes in my answers. Also, be sure to check out the new tips from Mr. Davis.

Betty's SRL persona incorporates aspects of this metacognitive knowledge that she conveys to the students to help them develop and apply monitoring and self regulation strategies [2]. For example, when the student is building the concept map, Betty occasionally responds by demonstrating reasoning through chains of events. She may remark (right or wrong) that the answer she is deriving does not seem to make sense. The idea of these spontaneous prompts is to get students to reflect on what they are teaching and perhaps check on their tutee's learning progress. These interactions are directed to help Betty's student-teacher understand the importance of monitoring and being aware of one's own abilities.

We have identified several recurrent sequences where metacognitive feedback might be useful. When the system detects such patterns, Betty provides suggestions the students may employ to improve their own understanding. Some of the triggering

patterns along with Betty's response are shown in Table 1. After Betty takes a quiz, the mentor agent also reminds Betty and her teacher about the importance of reading resources, and checking one's understanding after learning (teaching) new material.

3 Learning Results

In the main phase, the SRL condition generated maps with significantly more correct concepts and links than the LBT, $p < .05$, and ICS students, $p < .05$ [2]. These results suggest that the metacognitive prompting improved the students' learning. However, the LBT students also generated more correct maps than the ICS students, which suggests an overall benefit for learning by teaching.

Table 2. Concept map quality: main and transfer studies

Condition	Mean (SD) Map Scores	
	Main Phase	**Transfer Phase**
ICS	22.83 (5.3)	22.65 (13.7)
LBT	25.65 (6.5)[c]	31.81 (12.0)
SRL	31.58 (6.6)[a,b]	32.56 (9.9)[a]

[a] SRL > ICS, $p < .05$; [b] SRL > LBT, $p < .05$; [c] LBT > ICS, $p < .05$.

Students' transfer map scores provide indications whether a given version of the system better prepared students to learn in a new domain without scaffolds and prompts. Students in the SRL condition still had the highest map scores after the transfer phase, and scored significantly higher than the ICS students, $p < .05$. Interestingly, the LBT students' scores were now comparable to the SRL students. However, LBT students did not differ significantly from the ICS group, in part because of the high level of variability within that group.

Our interpretation is that working with Betty, especially with metacognitive prompts, helped students develop metacognitive strategies that supported their abilities to learn subsequently. However, another possible explanation is that ICS students were at a disadvantage because they switched from a non-teaching environment to a teaching environment in the transfer phase of the study, whereas the other students had not. By directly analyzing patterns of how students interact with the system, we explore the validity of these explanations.

4 Analysis of Behavior Patterns

We recorded log files of students' interactions with the system. From these log files, we identified the six main activities summarized in Table 3.

Sequences of these activities were then extracted from the log files, and mined using statistical learning methods to search for patterns that defined students' interactions with system. Our goal was to determine if there was evidence of different activity patterns

between groups during the main phase, and whether these patterns persisted or changed when students worked in a new environment during the transfer phase. We thus derived two sets of HMMs, one set from students' accumulated activity sequences from all main phase sessions, and the second set from the transfer phase sessions [10].

Table 3. Student Activities and Related Actions

Activity	Student Actions
Edit Map (**EM**)	adding, modifying, or deleting concepts and links
ASK QUERY (**AQ**)	asking Betty queries
REQUEST QUIZ (**RQ**)	asking Betty to take the quiz
RESOURCE ACCESS (**RA**)	accessing the resources
REQUEST EXPLANATION (**RE**)	asking Betty for an explanation to her query answer
CONTINUE EXPLANATION (**CE**)	asking Betty to provide a more detailed explanation

HMMs are so named because their states are hidden. That is, they are not directly observed in the input sequences, but provide an aggregated description of the students' interactions with the system. Sequences of states may be interpreted as the students' learning behavior patterns. The set of parameters that define a HMM comprise (i) the transition probabilities between the states, (ii) observation probabilities for detecting a particular observation in a state, and (iii) initial probabilities for each state [13]. The particular learning method used, developed by Li and Biswas [13] utilizes the Bayesian information criterion (BIC) to find the optimal number of states that define the HMM.

The HMM models derived for the three conditions in the two phases of our study are summarized in Figure 2. For convenience, each hidden state is labeled by the predominant activity (or activities) comprising that state. (Only those activities whose likelihood of occurrence exceed 10% are listed). Some states are dominated by a single activity (e.g., editing the map), whereas others represent a composite of more than one activity (e.g., requesting quizzes and accessing the resources). Figure 2 also provides the likelihood, expressed as a percentage, of a student in a given state transitioning to a different state or remaining in the current state.

4.1 Interpreting the HMMs

Much like factor analysis, it is up to the researcher to give meaning to the derived interactive states, and hypothesize strategies for learning that may be associated with these states. Our analyses suggest several interpretable patterns that are relevant to interactive metacognition. These patterns combine several activity states and transitions to define higher level behavior patterns with links to metacognitive strategies.

One pattern is *basic map building*. This activity pattern is characterized by editing the map (EM), submitting the map for a quiz (RQ), and occasionally accessing the reading resources (RA). The pattern reflects a basic and important metacognitive

strategy. Students work on their maps, check the map by taking a quiz to see if there are flaws, and occasionally refer to the readings.

A second pattern is *map probing*. Students edit the map (EM) and then ask a question (AQ) to check for specific relations between two concepts (e.g., if fish increase, what happens to algae?). This pattern exhibits a more proactive, conceptually driven strategy, because students are targeting specific relations rather than relying on the quiz to identify errors. Students also need to formulate their own questions to do so.

The third pattern is *map tracing*. This pattern reflects students asking Betty or the mentor (depending on the system) to explain the reasoning step by step (RE and CE). When Betty or the mentor initially answers a question, they state that a change in one entity causes a change in another entity and highlight the paths they followed to reach their answer. To follow the details of the inference chain, students had to ask Betty or Mr. Davis to explain their reasoning. The agents did so by hierarchically decomposing the chain of inference; for each explanation request, they showed how a particular path within the larger chain contributed to the final answer. Receiving more details about the reasoning process is particularly useful when maps become complex, and there are multiple paths between two concepts

The individual states portrayed in the original HMMs (Fig. 2) can be combined and re-represented in order to reflect these higher level aggregate states. These aggregate states are shown in Fig. 2, which are separated by condition and phase of the study. The percentages accompanying each arrow indicate the likelihood of transitioning from one aggregate to another or remaining in a given aggregate state. In addition, we exploited the stationary nature of these models to calculate the steady state probabilities of each aggregate state as the sum of the stationary probability of the individual states that make up the aggregate state (Table 4). These values indicate the probability that a student would be in a given state. For example, during the main phase, ICS students had a 62% chance of engaging in map building, but only a 7% chance of engaging in map tracing. The individual states' steady-state probabilities were also taken into account when calculating the transition probabilities between aggregate states.

4.1.1 Main Phase Patterns

One way to approach each figure is to assume that students begin with basic map building. As an example, the ICS models show that there is a greater than 90% chance that the students remain in the map building condition, and less than a 10% chance that they transition to the map probing state. The LBT behavior model is very similar to the ICS model, except that these students in this group were more likely to transition to the map probing state from the map building state (15%). Both the ICS and LBT groups rarely use Map Tracing as a learning behavior.

The SRL behavior model is different in that the map building and map probing states are tightly coupled, and thus aggregated into one state. This is not surprising because Betty's prompts required the students to ask queries and check her answers between quizzes. The aggregated model indicates that the SRL students were more likely to engage in map tracing behaviors (15% as opposed to 8% for the LBT group and 7% for the ICS group) perhaps to understand how she reasoned with the concept map to derive her answers. Overall, the SRL condition exhibited a more versatile repertoire of interactive strategies for completing the cognitive task of teaching Betty.

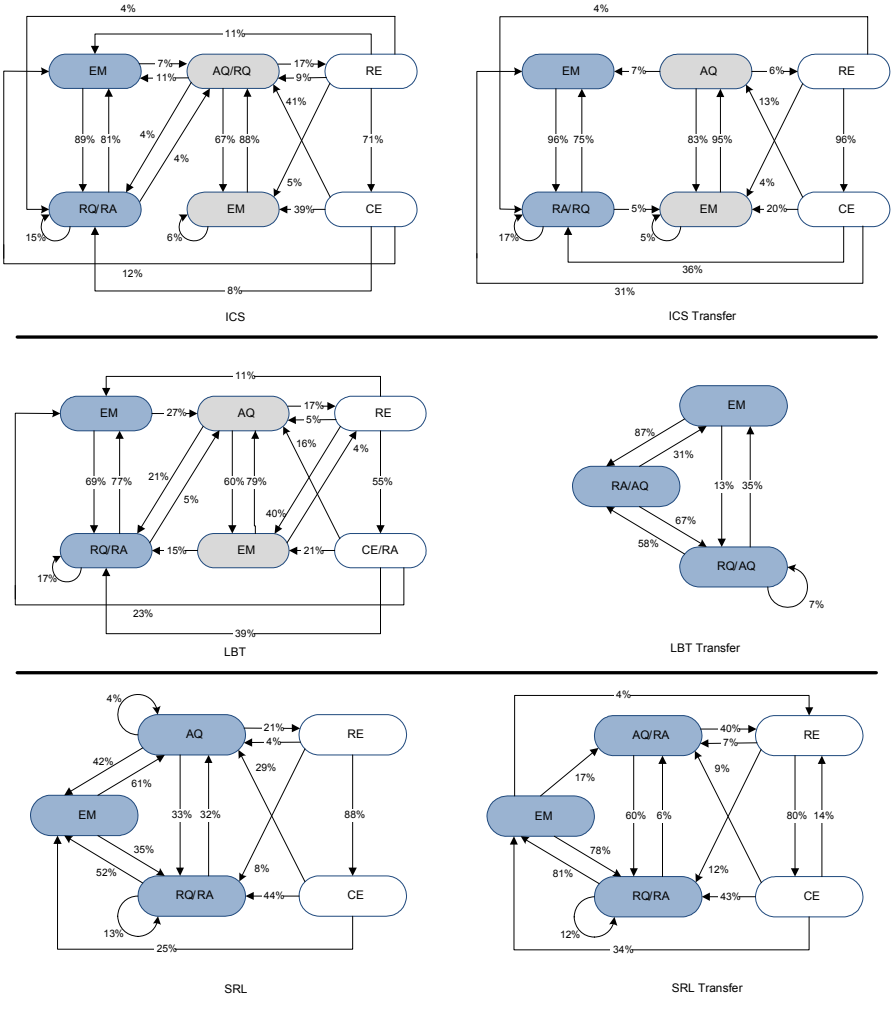


Fig. 2. HMMs for the three conditions in the main and transfer phases

This offers one explanation for why the SRL students generated higher quality maps in the main phase, even though they were never explicitly told how to correct their maps. The support for interactive metacognition, primarily in terms of seeking information from the resources, and monitoring Betty's learning helped them learn the content better than the other two conditions.

4.1.2 Transfer Phase Patterns

In the transfer phase, all students taught Betty but all scaffolds were removed. The only feedback students received was how well Betty performed on the quizzes. The question was whether there was any continuation of the patterns developed during the main phase. The ICS condition continued to focus on the basic map building pattern

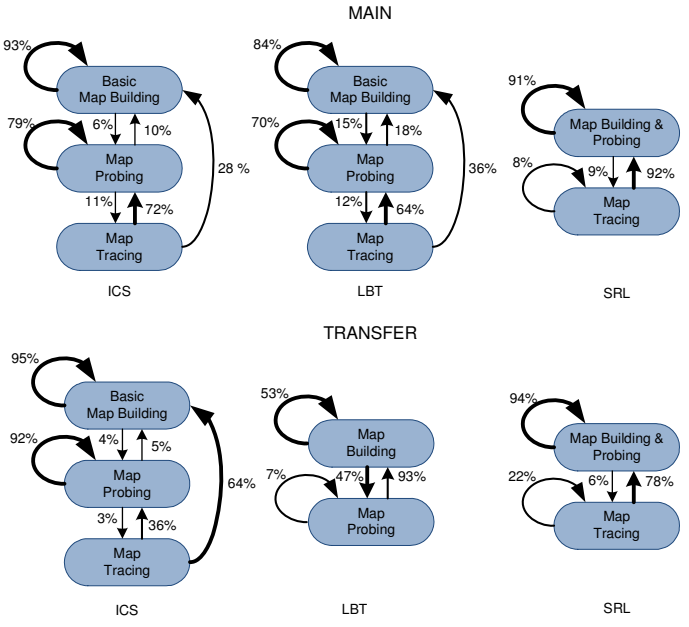


Fig. 3. Behavior Patterns for the three groups in the main and transfer study

Table 4. Aggregate state stationary probablities (i.e., probability of being in a given state)

State	ICS	LBT	SRL	Transfer ICS	Transfer LBT	Transfer SRL
Map Building	0.62	0.54	x	0.60	0.66	x
Map Probing	0.31	0.38	x	0.36	0.34	x
Map Tracing	0.07	0.08	0.15	0.04	x	0.11
Building & Probing	x	x	0.85	x	x	0.89

(60%). Their map probing behavior occurrence increased marginally (31% to 36%), and their use of the tracing mechanisms was limited (4%), even though they were now teaching Betty just like the other two groups. We inferred that the teaching aspect alone did not override the students' desire to simply get quiz answers right. The ICS students did not seem inclined to probe Betty's understanding, and by extension their own understanding.

The transfer phase behavior patterns exhibited by the SRL group were also similar to their main phase behaviors. The map building and map probing states were still aggregated, and occurred with high frequency (89%). The transitions from the building/probing state to map tracing decreased (9% to 6%). It is possible that the SRL students had internalized the reasoning mechanism and did not need to probe Betty as often or ask her to explain. However, once these students transitioned to the map tracing state, there were more internal transitions in that state (transition likelihood was

22%). This may indicate that when the concept map and the answer generation process became complex, the students did spend more time in map tracing activities.

The LBT condition behavior model also remained similar with map building and map probing dominating their learning activities. However, a more careful study of the more detailed model in Figure 3 reveals that within the map building phase these students spent almost twice as much time reading the resources as they did in editing their maps (41% to 25%). The amount of map tracing by this group seemed to decrease and all map tracing activity was integrated with the map building and map probing states. This version of Betty used in the transfer phase was most similar to the original LBT condition, except there was no corrective feedback provided by Mr. Davis. Therefore, it is reasonable to expect that the LBT condition would show the same interactive patterns across study phases. Instead, the LBT students seemed to develop a different learning strategy that included more time spent in reading the resources to learn about the new domain during the map building phase.

Unlike the SRL group the LBT group did not receive feedback on monitoring strategies in the main phase of the study. As a result, the LBT group did not seem to use map tracing as a learning strategy. Instead, their strategy was to track down Betty's wrong answers by querying her and then reading the resources to find how to change their map to get the correct answer (given that they were no longer given corrective feedback). Teaching in the main phase of the study seemed to have a positive effect on the LBT groups learning behaviors because they performed better than the ICS group in the transfer phase where both groups worked with the same system. The differences in the feedback received during the main phase of the study also produced differences in learning behaviors between the LBT and SRL groups. Whereas the LBT group combined map probing and reading of resources to learn the new domain, the SRL group also used map tracing when faced with complex reasoning situations with their concept maps.

5 Discussion and Conclusions

The Betty's Brain system is designed to leverage the benefits of learning by teaching and concept mapping to facilitate students' science learning and causal reasoning. Our hypothesis that working with Betty helped students engage in educationally productive cognitive and metacognitive processes is supported by the results reported here. Students who utilized the LBT and SRL systems constructed better concept maps with more causal relationships between entities than students who used the non-teaching ICS version of the system. Moreover, students' performance was strongest when we explicitly supported their use of self-regulated learning strategies by having Betty model and prompt for such behaviors. Not only did these students do well in the main phase of our study when the prompts were present, they also continued to outperform other groups in the transfer phase when the prompts were absent.

Although assessments of learning outcomes were in agreement with our hypotheses, it was also critical to explore students' actual behaviors during the teaching and learning process. Using HMMs to characterize students' behaviors allowed us to identify several meaningful patterns that distinguished our three experimental conditions in the main and transfer phases.

Examining the map building, map probing, and map tracing patterns across the three conditions provided interesting results. First, these results support the claim that our SRL system with metacognitive prompting was beneficial because it altered students' behaviors in positive ways. Whereas LBT and ICS students relied heavily on basic map building, we were successful in encouraging SRL students to engage in more probing and tracing. In addition, these beneficial patterns tended to persist in the transfer phase of the study. An interesting result is the learning strategy that the LBT students developed as they progressed from the main to the transfer phase. Although these students used more map building during the main study, they spontaneously showed a shift toward probing and resource reading to correct errors, but they did not develop the tracing behavior during the transfer phase. Because the LBT students used fairly similar systems in both the main and transfer phases, these results suggest that use of a learning-by-teaching system over a period of time may help students gradually *develop* better learning strategies. But there is also added value to focusing on metacognitive and self-regulated learning strategies through social interactions between tutors, their students, and agents who play the role of mentors. More research is needed to look at the benefits of extended use of our system.

Acknowledgments. This work has been supported by a Dept. of Education IES grant #R305H060089 and NSF REESE Award #0633856.

References

1. Biswas, G., Leelawong, K., Schwartz, D., Vye, N.: TAG-V: Learning By Teaching: A New Agent Paradigm for Educational Software. In: Applied Artificial Intelligence. AAI, vol. 19, pp. 363–392 (2005)
2. Wagster, J., Tan, J., Wu, Y., Biswas, G., Schwartz, D.: Do Learning by Teaching Environments with Metacognitive Support Help Students Develop Better Learning Behaviors? In: The 29th Annual Meeting of the Cognitive Science Society, Nashville, pp. 695–700 (2007)
3. Novak, J.D.: Learning, Creating, and Using Knowledge: Concept Maps as Facilitative Tools in Schools and Corporations, LEA, Mahwah, NJ (1998)
4. Roscoe, R.D., Chi, M.: Understanding tutor learning: Knowledge-building and knowledge-telling in peer tutors' explanations and questions. Review of Educational Research. 4, 534–574 (2007)
5. Biswas, G., Schwartz, D., Bransford, J.: TAG-V: Technology support for complex problem solving: From SAD environments to AI. In: Forbus, K.D., Feltovich, P.J. (eds.) Smart Machines in Education, pp. 71–98. AAAI Press, Menlo Park (2001)
6. Roscoe, R.D., Chi, M.: Tutor learning: The role of instructional explaining and responding to questions. In: Instructional Science (in press)
7. Chi, M.T.H., Siler, S.A., Jeong, H., Yamauchi, T., Hausmann, R.G.: Learning from Human Tutoring. Cognitive Science 25(4), 471–533 (2001)
8. Lin, X., Schwartz, D.L., Hatano, G.: Toward Teachers' Adaptive Metacognition. Educational Psychologist 40(4), 245–255 (2005)
9. Leelawong, K., Biswas, G.: Designing Learning by Teaching Systems: The Betty's Brain System. International Journal of Artificial Intelligence in Education (2008)

10. Rabiner, L.R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proc. IEEE, 77 (1989)
11. Azevedo, R.: Using Hypermedia as a Metacognitive Tool for Enhancing Student Learning? The Role of Self-Regulated Learning. Educational Psychologist 40(4), 199–209 (2005)
12. Zimmerman, B.J.: A Social Cognitive View of Self-Regulated Academic Learning. Journal of Educational Psychology 81(3), 329–339 (1989)
13. Li, C., Biswas, G.: A Bayesian Approach for Learning Hidden Markov Models from Data. Special issue on Markov Chain and Hidden Markov Models. Scientific Programming 10, 201–219 (2002)
14. Fisher, C., Sanderson, P.: Exploratory sequential data analysis: exploring continuous observational data. Interactions 3(2), 25–34 (1996)

To Tutor the Tutor: Adaptive Domain Support for Peer Tutoring

Erin Walker[1], Nikol Rummel[2], and Kenneth R. Koedinger[1]

[1] Human Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA, USA
[2] Department of Psychology, Albert-Ludwigs-Universitat Freiburg, Germany
erinwalk@andrew.cmu.edu, rummel@psychologie.uni-freiburg.de, koedinger@cmu.edu

Abstract. The effectiveness of intelligent tutoring systems at increasing learning might be improved if the systems were combined with collaborative activities that encouraged conceptual elaboration. We extended the Cognitive Tutor Algebra, an intelligent tutoring system for high-school mathematics, with a peer tutoring activity that was designed to encourage interaction, reflection, and accountability. Two types of domain support were provided: adaptive support, which used the intelligent tutor domain models to provide feedback to the peer tutor, and fixed support, which simply consisted of answers to the problems. We compared the two peer tutoring conditions (adaptive or fixed support) to individual use of the cognitive tutor (without peer-tutoring activities). Even though students in the individual condition solved more problems during instruction, we did not find significant differences between the individual and collaborative conditions on learning. However, we found a correlation between tutee impasses and tutor learning.

Keywords: Peer tutoring, cognitive tutor, algebra, in-vivo experimentation, adaptive collaborative learning system.

1 Introduction

The Cognitive Tutor Algebra (CTA) has been shown to increase student learning by roughly one standard deviation over traditional classroom instruction [1], and is used by about 475,000 students a year [2]. However, the impact of the intervention still falls short of the effectiveness of good human tutors, who can improve student learning by two standard deviations over classroom practice [3]. As students may acquire shallow conceptual knowledge while using tutoring systems, researchers augment cognitive tutors with activities that encourage conceptual elaboration such as self-explanation [4] and scripted collaboration [5]. However, it appears that in order for significant improvement over CTA instruction to occur, students must be able and motivated to apply the metacognitive skills targeted by an intervention [6].

In our work, we augment individual use of the CTA with a collaborative peer tutoring activity. Instead of the computer tutoring the student, students take turns tutoring each other (see Figure 1). Tutees can ask their tutors questions and self-explain, and tutors can then provide their tutees with elaborated help. Ideally, students in the tutee

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 626–635, 2008.

role should benefit from the peer instruction at least as much as students using the CTA individually, and students in the tutor role should benefit even more from the additional conceptual demands of tutoring. However, because peer tutors are also in the process of learning the domain material, they may not be able to provide the tutee with feedback that is timely or correct. The tutee may then be unable to successfully complete the curriculum problems, and will not benefit from the instruction. Therefore, we implemented a meta-tutor that provides adaptive domain support to the peer tutor. In this paper, we discuss how students learn from tutoring and how an adaptive system might support this process, describe the design of the meta-tutor, and compare the adaptive system to a fixed support system and to typical use of the CTA.

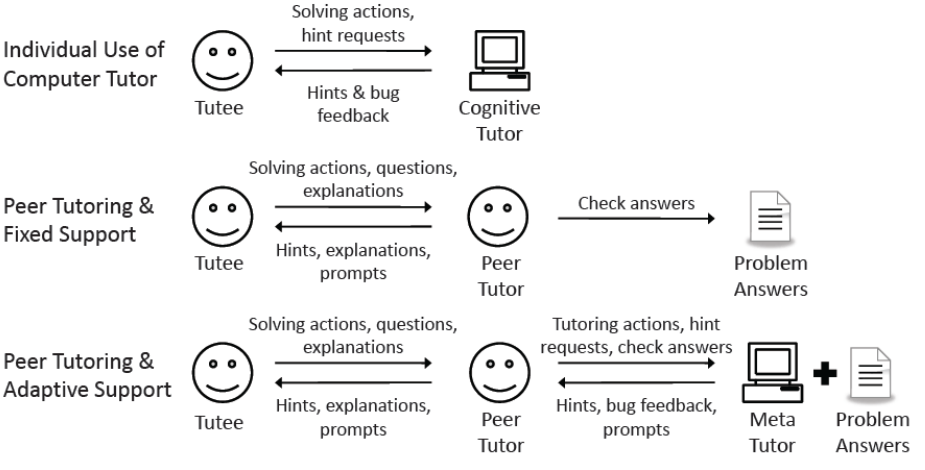


Fig. 1. Three tutoring scenarios used in the study

1.1 Peer Tutoring: Learning by Teaching

Incorporating peer tutoring into the CTA might be a way to encourage deep learning. Roscoe and Chi conclude that peer tutors benefit due to *knowledge-building,* where they reflect on their current knowledge and use it as a basis for constructing new knowledge [7]. Because these positive effects are independent of tutor domain ability, researchers implement reciprocal peer tutoring programs, where students of similar abilities take turns tutoring each other. This type of peer tutoring has been shown to increase academic achievement and positive attitudes in long-term classroom interventions [8]. Biswas et al. [9] described three properties of peer tutoring related to tutor learning: tutors are accountable for their tutee's knowledge, they reflect on tutee actions, and they engage in asking questions and giving explanations. Tutee learning is maximized at times when the tutee reaches an impasse, is prompted to find and explain the correct step, and is given an explanation if they fail to do so [10].

Peer tutors rarely exhibit knowledge-building behaviors spontaneously [7], and thus successful interventions provide them with assistance in order to achieve better

learning outcomes for them and their tutees. This assistance can target *tutoring behaviors* through training, providing positive examples, or structuring the tutoring activities. For example, training students to give conceptual explanations had a significantly positive effect on learning [11]. It is just as critical for assistance to target *domain expertise* of the peer tutors, in order to ensure that they have sufficient knowledge about a problem to help their partner solve it. Otherwise, there may be cognitive consequences (tutees cannot correctly solve problems) and affective consequences (students feel that they are poor tutors and become discouraged [12]). Domain assistance can take the form of preparation on the problems and scaffolding during tutoring [e.g., 8]. Although assistance for peer tutoring has generally been fixed, providing adaptive support may be a promising approach.

1.2 Adaptive Collaborative Learning Systems

In order to benefit from collaboration students must interact in productive ways, and collaborative activities can be structured (*scripted*) to encourage these behaviors [e.g., 13]. However, fixed scripts implemented in a one-size-fits-all fashion may be too restrictive for some students and place a high cognitive demand on others [13, 14]. An adaptive system would be able to monitor student behaviors and provide support only when needed. Preliminary results suggest that adaptive support is indeed beneficial: Adaptive prompting realized in a Wizard of Oz fashion has been shown to have a positive effect on interaction and learning compared to an unscripted condition [15]. An effective way to deliver this support would be to use an adaptive collaborative learning system, where feedback on collaboration is delivered by an intelligent agent.

Work on adaptive collaborative learning systems is still at an early stage. One approach is to use machine learning to detect problematic elements of student interaction in real-time and trigger helpful prompts. Although implementations have lead to significant learning gains, the adaptive feedback appears to be disruptive to dyadic interaction [16]. Another promising approach has explored using an intelligent agent as one of the collaborators; students teach the agent about ecosystems with the help of a mentoring agent [9]. However, the agents do not interact with the students in natural language, one of the primary benefits of collaboration.

With respect to peer tutoring, intelligent tutoring technology could be applied either to supporting tutor behaviors or domain knowledge of peer tutors. As it is very difficult to build an intelligent tutor for collaborative processes, we decided to develop a general script for the peer tutoring interaction and then focus on providing *adaptive domain assistance* to peer tutors by leveraging the existing domain models of the CTA. A condition where students tutor each other with adaptive domain support provided to the peer tutor is likely to be better than a condition where the peer tutor merely has access to an answer key, because the support would be tailored to each individual tutor's needs. It is also likely to be better than a condition where students use the CTA individually, because the students in the collaborative condition would be able to interact deeply about the domain material.

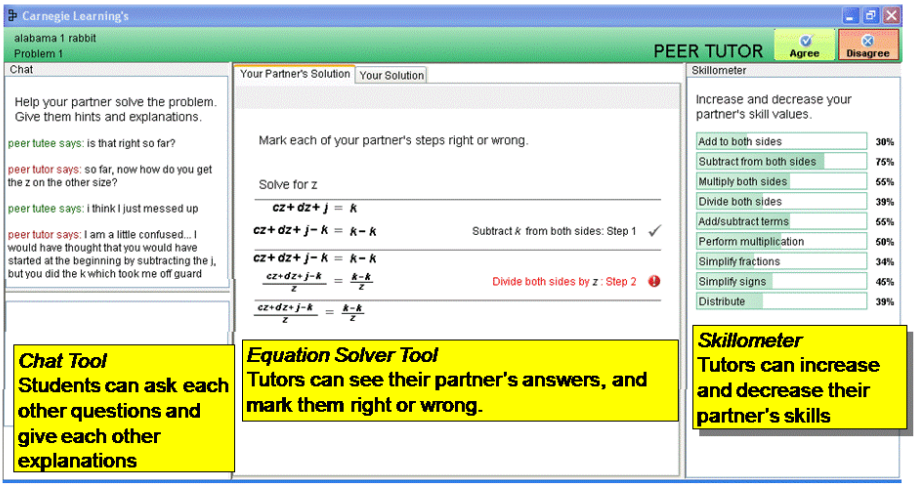


Fig. 2. Peer tutor's interface

2 Method

2.1 System Design

Peer Tutoring Script. We extended the CTA for peer tutoring using a literal equation solving unit, where students are given a prompt like "Solve for x," and then given an equation like "ax + by = c." Students went through a preparation and collaboration phase. In the *preparation phase*, students individually solved the problems they would later tutor. They used an equation solver to perform operations on the equation, were given immediate feedback from the CTA when making a mistake, and could ask for a hint from the CTA at any time. They were also given feedback on their progress by a Skillometer. After each problem in the preparation phase, we gave students reflection questions to prepare them for tutoring (e.g., "A good question asks why something is done, or what would happen if the problem was solved a certain way. What is a good question to ask about the problem?").

During the *collaboration phase*, students in the same class were grouped into pairs of similar abilities and collaborated with each other at different computers, taking turns being peer tutors and tutees on alternating problems. Although they were located in the same room, they were seated far apart and discouraged from talking to each other out loud. Peer tutees solved the same problems as their tutor had solved in the preparation phase, using the same interface. The peer tutor was able to see the peer tutee's actions, but could not solve the problem themselves (see Figure 2). Instead, the peer tutor took the role of the cognitive tutor, marking the peer tutee's actions right or wrong and adjusting the values of the tutee's skill bars. There was also a chat tool, where tutees could ask questions and tutors could provide hints and feedback.

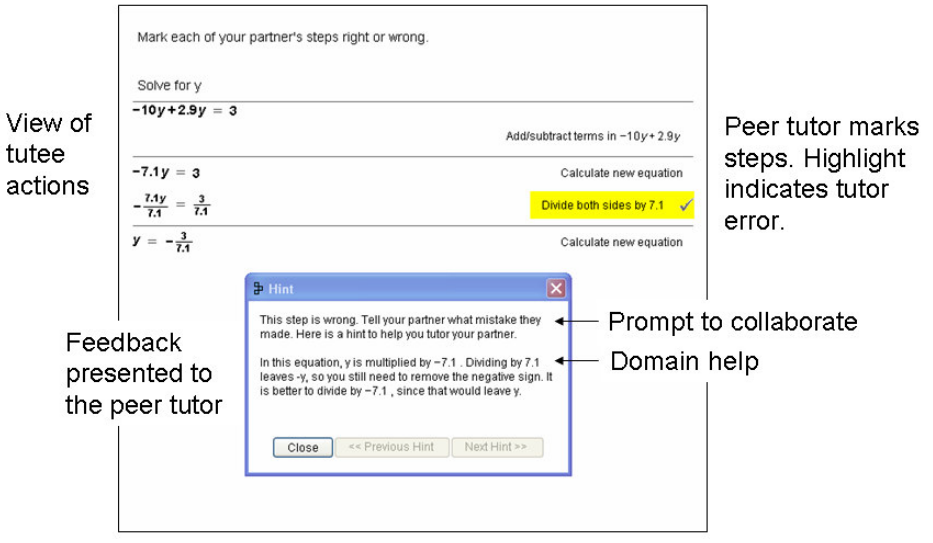


Fig. 3. Feedback presented to the peer tutor

Domain Support. We implemented two different support conditions for peer tutors: fixed domain support and adaptive domain support. In the *fixed support* condition, answers to the problem were located in a separate tab in the interface. Peer tutors could access the tab at any time, but viewing the tab they could no longer see what the tutee was doing. If both the tutee and tutor agreed that the problem was finished the students could move to the next problem, even if they were not actually done.

In the *adaptive support* implementation, peer tutors were given feedback by the intelligent tutoring system in two cases. If the peer tutee asked for a hint, the peer tutor could request it from the cognitive tutor and relay it to the tutee. If the peer tutor marked something incorrectly in the interface (e.g., they marked a wrong step by the tutee correct), the intelligent tutor would highlight the answer in the interface, and present the peer tutor with an error message. Hints and error messages were composed of a prompt to collaborate and the domain help the tutees would have received had they been solving the problem individually (see Figure 3). If both students agreed the problem was done, and were incorrect, the peer tutor would be notified and told to ask for a hint about how to complete the problem. In general, messages provided by the intelligent tutoring system were presented only to the peer tutor, and it was the peer tutor's responsibility to explain them to the tutee. Feedback was based on the peer tutor's actions, and not solely on the peer tutee's actions. As with the fixed support, peer tutors had access to the problem answers in the interface.

2.2 Experimental Design

We compared three conditions: (1) students tutored each other with adaptive domain support in addition to the peer tutoring script (*adaptive collaboration condition*), (2) students tutored each other with fixed domain support in addition to the peer tutoring

script (*fixed collaboration condition*), and (3) students used the CTA individually (*individual condition*). As argued above, we expected the adaptive collaborative condition to learn more than the fixed collaboration and individual conditions because of the combination of tutoring interaction and adaptive domain support.

Participants. Participants were 62 high school students from five algebra classes at a vocational high school in the United States, taught by the same teacher. The high school used the individual version of the CTA as part of regular classroom practice. Students from each class were randomly assigned to one of the three conditions. 11 students were excluded from analysis because either they or their partner were absent during a collaborative part of the intervention, and they were not re-paired with another student. Another 12 participants did not take the delayed posttest, but were included in all other analyses. The total number of students included in the analysis was 51 (20 in the individual condition, 14 in the fixed collaboration condition, and 17 in the adaptive collaboration condition). 39 students took the delayed posttest (18 in the individual condition, 10 in the fixed collaboration condition, and 11 in the adaptive collaboration condition).

Procedure. The study took place over five weeks. Students were given a 15 minute pretest during the first week. The intervention took place during two 70 minute class periods, each one week apart. On both intervention days, students in the collaborative conditions spent half the period in the preparation phase and the remaining time taking turns tutoring each other in the collaboration phase. Students in the individual condition used the CTA alone during both phases. The week after the intervention, students were given a 15 minute posttest. Two weeks later, students were given a 15 minute delayed test to assess their long-term retention. The pre-, post-, and delayed tests were counterbalanced, contained 8 questions, and were administered on paper.

3 Results

3.1 Learning Gains

We scored answers on the pre-, post-, and delayed tests by marking whether the solutions were correct or incorrect. If students got a completely correct solution or reached a nearly correct solution with just a copying error, they received a 1. If students performed at least one important conceptual step incorrectly they received a 0. Points on all the questions were summed, with a maximum score of 8 points. We conducted a two-way (condition x test-time) repeated-measure ANOVA, with test-time (pretest, posttest, or delayed test) as the repeated measure. There was a significant effect for test-time ($F(2,72) = 41.303$, $p < .001$), but there were no significant differences between conditions, and no interaction. A priori contrasts revealed that the effect was due to the difference between the pretest and the other two tests ($t(36) = 69.541$), $p < .001$) and not due to the difference between the posttest and the delayed posttest ($t(36) = 2.544$, $p = .119$). Table 1 contains the scores of the students who took all three tests. For the correlational analysis in the remainder of this section, we computed normalized gain scores for the posttest and the delayed test.

Our hypothesis that the adaptive collaboration condition would lead to more learning than the other two conditions was not supported by the data. We next investigated how process related to learning outcomes, by examining student progress through the unit, the effect of tutee impasses, and the feedback that students received.

Table 1. Pre, post and delayed test scores

	Pretest		Posttest		Delayed Posttest	
Condition	M	SD	M	SD	M	SD
Individual	1.28	1.60	3.00	1.75	3.67	1.78
Fixed	.90	.876	3.50	2.17	3.60	2.17
Adaptive	.82	1.08	2.36	1.57	2.82	1.78

3.2 Student Progress

We expected that the collaboration conditions might complete fewer problems than the individual condition because students spend more time interacting. However, tutees in all conditions should make similar numbers of incorrect problem-solving actions per problem if they receive comparable support from their tutors. We conducted a one-way (condition: individual, fixed, adaptive) ANOVA on the number of problems completed per hour in the collaboration phase of the study (see Table 2). For this analysis, we grouped the students in the collaborative conditions by dyad, as the number of problems that one pair member completes (and the time that they take) is dependent on the number of problems the other pair member completes. Condition was indeed significantly related to problems solved ($F(2,34) = 8.764$, $p = .001$). We then conducted a one-way (condition: individual, fixed, adaptive) ANCOVA on the average number of incorrect actions per problem (see Table 2). We included pretest as a covariate because it was significantly correlated with incorrect attempts. Because we wanted a comparable measure of individual progress across conditions, we looked at incorrect attempts per problem *for each tutee*, rather than by dyad. Pretest was significantly predictive of incorrect attempts ($F(1,47) = 5.449$, $p = .024$). Condition marginally affected incorrect attempts per problem ($F(2,47) = 2.480$, $p = .095$).

Table 2. Problems completed, incorrect attempts, help requested, and help given

		Problems Completed per hour			Incorrect Attempts per problem		Help Requested per problem		Help Given per problem	
Condition	*N*	M	SD	*N*	M	SD	M	SD	M	SD
Individual	*20*	47.0	30.2	*20*	1.46	1.26	.648	.806	1.41	1.41
Fixed	*8*	13.3	7.71	*14*	1.81	1.04	.929	.647	.943	.839
Adaptive	*9*	17.7	5.69	*17*	2.46	1.87	1.32	1.60	1.96	1.63

Table 3. Types of help requested and given

Role	Utterance Type	Utterance Content	Measure
Tutor	Explanation	you need to factor f to get it by itself	Help Given
Tutor	Hint	you need to get the 4 away from the t	Help Given
Tutor	Prompt	but where is the -1.3333 come from?	Help Given
Tutor	Instruction	divide y+r to both sides	Help Given
Tutor	Confusion	I'm not sure I going to check the hint	None
Tutee	Specific Request	now do I divide both sides by (3-x)	Help Requested
Tutee	General Request	what do I do next	Help Requested
Tutee	Specific Description	I still have the m-n on the y side	None
Tutee	General Description	I think I did it right	None
Tutee	Confusion	I have noooooooo clue	Help Requested

3.3 Tutee Impasses

The collaborative conditions differed on how easy it was for students to move to the next problem. In the adaptive condition, students could not continue unless they had successfully completed the problem, making it possible for students to get "stuck", where they repeatedly tried incorrectly to move to the next problem. The number of these incorrect done tries was negatively correlated with tutee gain scores on the delayed test (r = -.591, p = .056), but positively correlated with tutor gain scores on the delayed test (r = .463, p = .115). In the fixed condition, students were not notified when their attempts to continue were incorrect, and thus could "skip" to the next problem even if the previous problem was not done. Problems skipped were negatively correlated with tutee learning (r = -.614, p = .059) and tutor learning (r = -.369, p = .329). If problems were skipped tutors did not benefit from tutee impasses.

To further investigate, we looked at how incorrect attempts might be related to tutee and tutor learning. In the adaptive collaboration condition, total incorrect problem-solving attempts were negatively correlated with tutee gain scores on the delayed test (r = -0.614, p = .044), but *positively* correlated with the delayed gain score of the tutor (r = .428, p = .190). Posttest correlations with tutee learning (r = -.206, p = .427) and tutor learning (r = .320, p = .210) were not as strong. In the fixed collaboration condition, the pattern was still present, but slightly weaker for the tutees, perhaps because students could sidestep impasses by skipping to the next problem; incorrect attempts were negatively correlated with tutee gains on the delayed test (r = -.378, p = .281) and positively correlated with tutor gains on the delayed test (r = .472, p = .199). In this condition, posttest scores were not correlated with incorrect attempts for the tutee (r = -.046, p = .876) or the tutor (r = .034, p = .917).

3.4 Feedback Received

Even though the number of incorrect attempts made across conditions was not significantly different, there may have been differences in the way tutors gave feedback. We computed a measure for help requested by counting the number of times the students in the individual condition clicked on the hint button and the number of times peer tutees in the collaborative conditions expressed confusion or asked a question (see

Table 3). We then computed a measure of help given by the peer tutor by counting the number of times the cognitive tutor gave a feedback message and the number of times the peer tutor gave advice (see Table 2). Help given was not significantly different across conditions ($F(2,48) = 2.16$, $p = .127$), nor was help requested ($F(2,48) = 1.794$, $p = .191$).

4 Discussion

Both individual use of the CTA and peer tutoring activities lead to significant learning gains. The fact that the collaborative conditions performed just as well as the individual condition is encouraging. They achieved similar gains even though they solved fewer problems, suggesting that interacting in depth about a small number of problems might be as efficient for learning as solving a large number. Further, peer tutoring can have social and attitudinal benefits [8], so students may gain more from peer tutoring than from working individually.

It is surprising that quantitative measures of student progress and feedback exchange within problems were so similar across all three conditions. Although one would expect hint requests to be the same across tutees and individual problem-solvers, one might expect that peer tutors would have difficulty giving hints to a tutee compared to an intelligent system, either delivering more or less help than necessary. However, the results indicate that on a broad level the mechanisms of intelligent tutoring and novice tutoring are similar. It appears that to improve the effects of the tutoring, the best approach may be to focus on the details of the interaction by coding utterances for the type of help given or requested (e.g., using Webb's coding scheme [17]). We can look more closely at how different feedback provided to peer tutors affects quality and timing of their help, and how those elements might relate to tutor and tutee learning.

The tutors' apparent benefit from tutee impasses, which were negatively correlated with tutee learning gains, is problematic since it suggests that in order for the tutor to improve, the tutee must struggle. This result occurred mainly on the delayed test, which is a measure of long-term retention, and therefore a measure of deeper learning than the posttest. It is important to be cautious in interpreting this correlation, but it is consistent with the result that viewing erroneous worked examples may improve student learning [18]. If this is the case, it is important to give students the opportunity to experience these impasses. However, we need to examine in greater detail why tutors could not help tutees benefit from impasses, and provide assistance for these circumstances to better support the peer tutor in explaining the relevant concepts to the tutee. Focusing on this aspect, we may be able to use collaborative learning to improve on individual use of the CTA.

Acknowledgments. This research is supported by the Pittsburgh Science of Learning Center, National Science Foundation Grant #0354420. Thanks to Amy Ogan, Ido Roll, Dejana Diziol, and Sean Walker for their helpful comments.

References

1. Koedinger, K., Anderson, J., Hadley, W., Mark, M.: Intelligent tutoring goes to school in the big city. International Journal of Artificial Intelligence in Education 8, 30–43 (1997)
2. Carnegie Learning, http://www.carnegielearning.com
3. Bloom, B.S.: The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher 13, 3–16 (1984)
4. Aleven, V., Koedinger, K.R.: An effective metacognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor. Cognitive Science 26, 147–179 (2002)
5. Diziol, D., Rummel, N., Spada, H., McLaren, B.: Promoting learning in mathematics: Script support for collaborative problem solving with the Cognitive Tutor Algebra. In: Chinn, C.A., Erkens, G., Puntambekar, S. (eds.) CSCL 2007, pp. 39–41 (2007)
6. Roll, I., Aleven, V., McLaren, B.M., Koedinger, K.R.: Can help seeking be tutored? Searching for the secret sauce of metacognitive tutoring. In: Luckin, R., Koedinger, K., Greer, J. (eds.) AIED 2007, pp. 203–210 (2007)
7. Roscoe, R.D., Chi, M.: Understanding tutor learning: Knowledge-building and knowledge-telling in peer tutors' explanations and questions. Review of Educational Research 77(4), 534–574 (2007)
8. Fantuzzo, J.W., Riggio, R.E., Connelly, S., Dimeff, L.A.: Effects of reciprocal peer tutoring on academic achievement and psychological adjustment: A component analysis. Journal of Educational Psychology 81(2), 173–177 (1989)
9. Biswas, G., Schwartz, D.L., Leelawong, K., Vye, N.: TAG-V. Learning by teaching: A new agent paradigm for educational software. Applied Artificial Intelligence 19, 363–392 (2005)
10. VanLehn, K., Siler, S., Murray, C., Yamauchi, T., Baggett, W.: Why do only some events cause learning during human tutoring? Cognition and Instruction 21(3), 209–249 (2003)
11. Fuchs, L., Fuchs, D., Hamlett, C., Phillips, N., Karns, K., Dutka, S.: Enhancing students' helping behaviour during peer-mediated instruction with conceptual mathematical explanations. The Elementary School Journal 97(3), 223–249 (1997)
12. Medway, F., Baron, R.: Locus of control and tutors' instructional style. Contemporary Educational Psychology 2, 298–310 (1997)
13. Rummel, N., Spada, H.: Can people learn computer-mediated collaboration by following a script? In: Fischer, F., Kollar, I., Mandl, H., Haake, J. (eds.) Scripting computer-supported communication of knowledge. Cognitive, computational, and educational perspectives, pp. 47–63. Springer, New York (2007)
14. Dillenbourg, P.: Over-scripting CSCL: The risks of blending collaborative learning with instructional design. In: Kirschner, P.A. (ed.) Three worlds of CSCL. Can we support CSCL, pp. 61–91. Open Universiteit Nederland, Heerlen (2002)
15. Gweon, G., Rosé, C., Carey, R., Zaiss, Z.: Providing Support for Adaptive Scripting in an On-Line Collaborative Learning Environment. In: Proc. of CHI 2006, pp. 251–260 (2006)
16. Kumar, R., Rosé, C.P., Wang, Y.C., Joshi, M., Robinson, A.: Tutorial dialogue as adaptive collaborative learning support. In: Proceedings of the 13th International Conference on Artificial Intelligence in Education (AIED 2007). IOSPress, Amsterdam (2007)
17. Webb, N., Troper, J., Fall, R.: Constructive activity and learning in collaborative small groups. Journal of Educational Psychology 87(3), 406–423 (1995)
18. Große, C.S., Renkl, A.: Finding and fixing errors in worked examples: Can this foster learning outcomes? Learning & Instruction 17, 612–634 (2007)

Shall We Explain? Augmenting Learning from Intelligent Tutoring Systems and Peer Collaboration

Robert G.M. Hausmann, Brett van de Sande, and Kurt VanLehn

Pittsburgh Science of Learning Center, University of Pittsburgh,
3939 O'Hara Street, Pittsburgh, Pa, 15260-5179
{bobhaus,bvds}@pitt.edu, vanlehn@cs.pitt.edu

Abstract. Learning outcomes from intelligent tutoring systems (ITSs) tend to be quite strong, usually in the neighborhood of one standard deviation. However, most ITS designers use the learning outcomes from expert human tutoring as the gold standard (i.e., two standard deviations). What can be done, with the current state of the art, to increase learning from an ITS? One method is to modify the learning situation by asking students to use the ITS in pairs. To enhance performance, we drew upon the beneficial effects of structured peer collaboration. The results suggest that the intervention was successful. Pairs of students solved more problems and requested fewer bottom-out hints than individuals. To test the possibility that the effect was due to the best partner in the group directing the problem solving, a nominal groups analysis was conducted. A nominal group is a statistical pairing of the non-interacting individuals' performance. The results from the nominal groups replicated the same pattern of results, but with a reduced magnitude. This suggests that the best member may have contributed to some of the overall success of the pair, but does not completely explain their performance.

Keywords: Collaborative learning; explanation activities; studying examples.

1 Introduction

An often-heard suggestion is that students may learn more from an intelligent tutoring system (ITS) if two students worked together on the system instead of working on it alone. Early studies did not support this hypothesis, and instead suggested that having pairs of students using an ITS produced the same learning gains as having students work alone as they used it [early studies with Lisp & geometry tutors]. However, null results are often open to many interpretations, such as a lack of statistical power. This issue has been re-opened recently, and for good reasons. The simplest reason for re-opening the pair/solo hypothesis is that, despite the designers' best efforts, the hints given by an ITS are sometimes more confusing than helpful. Perhaps having two students interpret the hints may help alleviate this problem.

Another reason for studying pairs using an ITS is that learning gains can be impressive when the students in a pair actually collaborate, as opposed to one student dominating the problem solving. Collaborators can encourage each other's reasoning, notice and productively resolve conflicts in their thinking, confirm each other's beliefs, externalize

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 636–645, 2008.

their thoughts, and so on. Collaborators could use an ITS to catch mistakes that the students manage to overlook, to provide hints when they have exhausted their mutual knowledge, or to resolve conflicts that they cannot resolve themselves. In short, metacognitive strategies exist for using an ITS as an effective scaffolding for peer problem solving.

Lastly, many observers have noticed that students working alone often abuse the ITS by asking for more help or less help than they need [1]. In particular, many ITSs give increasingly specific hints when asked, and the last "bottom-out" hint tells the student exactly what step to enter. Some students ask for bottom-out hints on almost every step. Conversely, some students will enter incorrect versions of a step dozens of times without asking for a hint. Although it is possible that students do not know how to use the help system effectively, experiments with a help-seeking tutor have shown that explicitly teaching students effective help-seeking strategies did not change their long-term behavior [2]. They went back to abusing the ITS as soon as the help-seeking tutor was replaced by the regular ITS. This suggests that students know how to seek help effectively, but they sometimes choose otherwise.

One way to reduce the frequency of help misuse may be to have students work in pairs. If both students know that rapidly pressing the hint button in order to get the bottom-out hint is bad for learning, then such abuse would only occur if they both simultaneously agree to be "bad." Similarly, they would have to both agree to enter mistaken entries repeatedly without asking for help, even though they both know this could be a waste of time. So perhaps having students work in pairs on an ITS could increase the frequency of proper help usage, compared to students working alone.

In short, there are at least three reasons why pairs of student should learn more than individuals using an ITS: (1) pairs may be able to interpret the ITS's hints more successfully, (2) the ITS may help student collaborate effectively, and (3) pairs of students are less likely to abuse the ITS than individuals.

Although the first and third hypotheses are somewhat novel, there has been a considerable amount of work on the second hypothesis [3, 4]. For example, Rummel and Spada [5] contrasted learning and problem solving under four different conditions. The first was a detailed, worked-out example of successful collaboration. Participants were asked to study the example of smooth collaboration, and apply it to their own dialog. A second group was provided with a collaboration script, in which case their interactions were structured in a way that was hypothesize to promote successful collaborative outcomes. Finally, there were two control conditions that did not structure the collaborative interactions. They found that the example and scripted conditions demonstrated better understanding of successful collaboration principles, as well as a better understanding of the domain (i.e., therapy and diagnosis of medical and psychological disorders) than the control conditions.

The laboratory study by Rummel and Spada suggests that successful collaborative interactions can be instructed and scripted. Walker et al. [6] extended this finding by developing a cognitive tutor that teaches students how to interact in the context of peer tutoring. They developed a peer-tutoring script that assists students along three dimensions. The first dimension prepares students for peer tutoring by providing instruction on the domain content, as well as relevant pedagogical strategies. The second dimension involves actual tutoring, whereby the tutor is taught to set goals for the student, as well as monitor the student's progress during problem solving. The last

dimension introduces skills for effective interaction, such as providing elaborated help. These three dimensions were used to construct a cognitive tutoring system for peer collaboration, thus reifying the empirical results on effective peer tutoring and collaboration.

Although the available evidence suggests that ITSs can have a positive impact on collaborative learning (i.e., hypothesis 2), research addressing the collaborative use of ITS hints has been relatively sparse (i.e., hypotheses 1 and 3). However, the empirical work on collaboratively studying worked-out examples, which is reviewed in the next section, may be of some relevance because requesting a string of bottom-out hints can turn a problem into a worked-out example.

1.1 Collaborative Example Studying

So far, we have discussed the hypothesis that pairs would be more effective than individuals when they work on an ITS, but similar remarks apply to studying examples as well. By *example*, we mean a problem plus a presentation of the multiple steps required for its solution. When individuals study an example, they sometimes self-explain it by filling in the gaps between steps, relating the inter-step reasoning to prior knowledge, etc. [7]. Sometimes students do anticipatory self-explanation, where they try to generate the next solution step themselves, then look at the example to see if they are right [8]. Prompting can increase the amount of self-explanation [9, 10]. However, even with prompting, students can abuse an example just as they can abuse an ITS. They abuse an example by simply reading it shallowly and not trying to self-explain much of it.

When discussing pairs using an ITS, we suggested that they may be more effective than individuals for 3 reasons: (1) pairs may be able to interpret the ITS's hints more successfully, (2) the ITS may help student collaborate effectively, and (3) pairs of students are less likely to abuse the ITS than individuals. Those same three reasons apply to examples as well. (1) A pair may be more able to interpret an examples' reasoning more successfully than an individual student. (2) An example scaffolds collaborators by helping them extend their reasoning when they get stuck, resolve conflicts productively, externalize their reasoning, confirm their beliefs, etc. (3) Pairs are less likely to abuse an example than individuals.

Laboratory evidence, which suggests that pairs may be better suited for studying examples than individuals, can be found in [11]. In their experiment, participants were asked to study some instructional materials collaboratively, then solve LISP programming problems individually. The pairs' performance on the LISP problems was contrasted with individuals studying the instructional materials alone. They found that the programming performance of the pairs was significantly better than the solo student performance; however, the authors note that the advantage for collaboration diminished over time.

The present study is another step toward understanding if and when "two heads are better than one" for learning. The study compares pairs vs. solo students who are both studying examples and solving problems with an ITS. As they study examples, they are prompted to self-explain. This study is preliminary in that we did not use pre-tests and post-tests, and thus cannot measure students' learning gains. However, we did record their behavior during the training in order to determine if it was affected by the solo/pair manipulation. Thus, this study counts as a *manipulation check* for a subsequent study of learning gains.

Although verbal protocols were collected, they have not yet been fully analyzed, so this paper reports only the analyses of log files generated by the tutoring system. Using them, we found evidence that pairs abused the help system less frequently than solo students, as predicted. We also looked for but failed to find signs of collaborative facilitation, in that pairs would make fewer errors due to collaboration than nominal pairs. On the other hand, the pairs did no worse than the nominal pairs, so there was no process loss [12]. Thus, all the current evidence is positive—two heads may indeed be better than one for explaining examples and solving problems with an ITS.

1.2 Problem Solving and Example Studying in an ITS

The Andes physics tutor was initially developed as a replacement for paper and pencil homework problems. The advantage for solving problems with Andes is the adaptive support it provides to the student. One form of adaptive support is the on-demand hints, which are provided in a graded fashion. Typically, the first hint points the student's attention to a relevant feature of the problem (i.e., a *Pointing Hint*). The second hint level presents general instructional principles that are relevant to the problem (i.e., a *Teaching Hint*). Finally, at the terminal level, the bottom-out hint tells the student exactly which action to take (i.e., a *Bottom-out Hint*).

In addition to the on-demand hints, Andes provides a series of videos that the students can watch in order to learn about various solution strategies, as well as how to use elements of the Andes interface. For the purposes of the current study, we modified the available videos so that they were broken down into individual problem-solving steps. Typically, students are responsible for starting and stopping the videos, but that generally leads to shallow cognitive processing of the video content. Instead, at the juncture of each step, we prompted the students to engage in an explanatory activity. Solo students and pairs were prompted to generate explanations while studying video-based, worked-out examples. The purpose of prompting students was to increase their cognitive processing of the examples.

The use of examples in the present experiment slightly diverges from traditional studies in the sense that students were prompted to engage in explanation while studying an isomorphic worked-out example *after* solving a related problem. We used this design for two reasons. First, the ACT-R theory of learning suggests that students only learn from the correct application of knowledge [13]. Second, the cognitive load associated with problem solving can impede a deep encoding of the problem-solving goals and operators [14].

2 Method

The following experiment was designed to test the effects of collaboration on problem solving and example studying while using an ITS, primarily with an emphasis on the collaborative use of hints.

2.1 Participants

Thirty-nine undergraduates, enrolled in a second semester physics course, were randomly assigned to one of two experimental conditions: solo students (n = 11) or pairs

(n = 14). Volunteers were recruited from several sections of a second-semester physics course, which covered Electricity and Magnetism. Participants were recruited during the third week of the semester, with the intention that the experimental materials would coincide with their introduction in the actual physics course. The participants were paid $10 per hour. To ensure that the participants' motivation remained high during the entire two-hour session, they were offered an incentive of an additional $10 for doing well on the tests, which they all received.

2.2 Materials

The materials developed for this experiment were adapted from an earlier experiment [15]. The domain selected for this experiment was electro-dynamics with a focus on the definition of the electric field, which is expressed by the vector equation: $\boldsymbol{F} = q\boldsymbol{E}$. This particular topic is typically covered within the first few weeks of a second-semester physics course. Thus, it is an important concept for students to learn because it represents their first exposure to the idea that a field can exert a force on a body.

To instruct the participants, several materials were developed. Four electro-dynamics problems were created. These problems are representative of typical problems found at the end of a chapter in a traditional physics textbook. The problems covered a variety of topics, including the definition of the electric field, Newton's first and second law, the weight law, and several kinematics equations. Each of the four problems was implemented in Andes. Andes was chosen because its design allowed for both the presentation of video-based examples, as well as coached problem solving. The first problem served as a warm-up problem because none of the students had any prior experience with the Andes user interface.

In addition to the problems, three examples were created in collaboration with two physics instructors at the U.S. Naval Academy. The examples contained a voice-over narration of an expert solving the problems, and they were structured such that they were isomorphic to the immediately preceding problem.

2.3 Procedure

The procedure consisted of several activities. The first activity was to watch a short, introductory video on the Andes user interface. Afterwards, the participants read instructions on how to produce explanations, including an example. Next, participants were asked to use Andes to solve a warm-up problem. The experimenter was available to answer any user-interface questions. He was not, however, allowed to give away any domain-specific information. During problem solving, the student had access to the flag feedback, the hint sequences, and an Equation Cheat Sheet. Once the student submitted a final answer, she then watched and explained an example of an expert solution of an isomorphic problem. The example solutions were broken down into steps, and at the conclusion of each step the student was prompted to explain (either individually or collaboratively). Once the explanation was complete, the participant clicked a button to go onto the next step. Only the cover story and given values differed between the problem-solving and example problems. The students alternated between solving problems and studying examples until all four problems were solved and all three examples were studied, or until two hours elapsed.

2.4 Measures

Several dependent measures, taken from the log files, were used to assess problem-solving performance, including: the number of entries, correct entries, solution rate, and the number of bottom-out hint requests.

3 Results

The results are organized into two sections. The first reports performance differences between the solo students and pairs at the problem level. The second section then reports the same dependent measures using a "nominal group analysis," which is considered the gold standard for collaborative research [16]. A nominal group is a statistical pairing of the non-interacting individuals. To construct a nominal group, individuals from the solo condition were randomly paired, and the best performance from each individual was taken to represent the pair.

3.1 Performance Differences

Before delving into the problem-solving performance measures, we first analyzed the solution rates (i.e., whether or not a final answer was found) for two reasons. First, the students worked at their own pace; and second, the experiment was capped at two hours. Thus, there was no guarantee that all of the students would finish all of the problems in the allotted time. The pairs were much more likely to submit an answer to the final problem than the solo students ($\chi^2 = 4.81$, $p = .03$).

An analysis of the mean number of entries and correct entries for the final problem confirmed the solution rate results. The pairs ($M = 34.29$, $SD = 6.72$) demonstrated more entries than the solos ($M = 25.00$, $SD = 14.97$), $F(1, 23) = 4.32$, $p = .05$, $d = .87$. Moreover, the pairs ($M = 23.29$, $SD = 5.06$) demonstrated reliably more correct entries for the final problem than the solos ($M = 15.64$, $SD = 9.85$), $F(1, 23) = 6.36$, $p < .02$, $d = 1.06$. Taken together, these results suggest that the pairs were more efficient in solving the problems during the two-hour experiment.

To test if the participants abused the available help, bottom-out hint requests were analyzed. Requesting multiple bottom-out hints is an indication that the student required more direct instruction, and this may have translated into *gaming the system* behaviors. However, if the student requests a reasonable number of bottom-out hints, then that is an indication that she is more interested in connecting the information found in the instructional materials (i.e., examples) to the individual problem-solving steps.

The bottom-out hint usage interacted with time, such that the difference between conditions was apparent early in the experiment, but diminished over time. To correct for Type II errors due to multiple statistical tests across the four problems, a repeated measures ANOVA was used. The univariate tests, which contrasted the two conditions for each problem, indicated that the solos demonstrated marginally higher bottom-out hint requests for the warm-up problem ($F(1, 21) = 3.98$, $p = .06$), reliably more hint requests for the first problem ($F(1, 21) = 7.64$, $p = .01$), and no reliable differences for the final two problems (see Fig. 1).

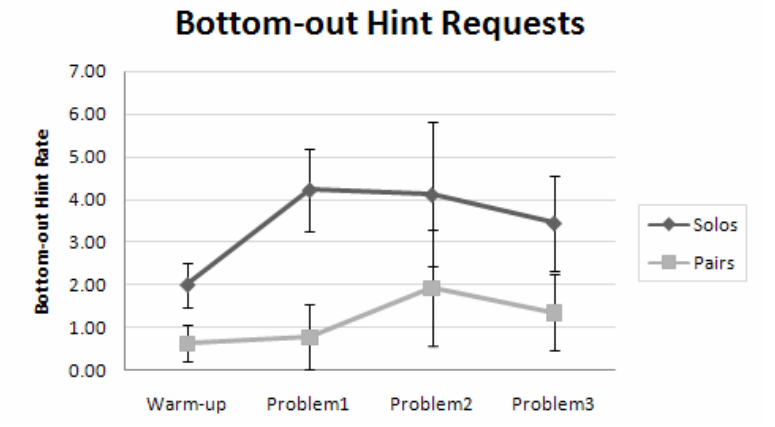


Fig. 1. The mean number of bottom-out hint requests per problem for solos and pairs

3.2 Nominal Group Analyses

One of the dangers of working in a collaborative setting is the threat of requiring more time to complete a task than it would when working alone. To test if there was a loss in efficiency, we compared real pairs to the nominal pairs by measuring the amount of time taken to input each correct entry. The results suggest there were no time penalties for working in a group. In fact, there was a small amount of evidence to the contrary. On average, real pairs (M = 47.96, SD = 17.17) demonstrated faster times between correct entries for the first problem than the nominal pairs (M = 67.05, SD = 16.45), $F(1, 23) = 7.89$, $p = .01$, $d = 1.18$.

In addition, the dependent measures used to contrast problem-solving performance between the two experimental conditions were repeated for the real pairs and the

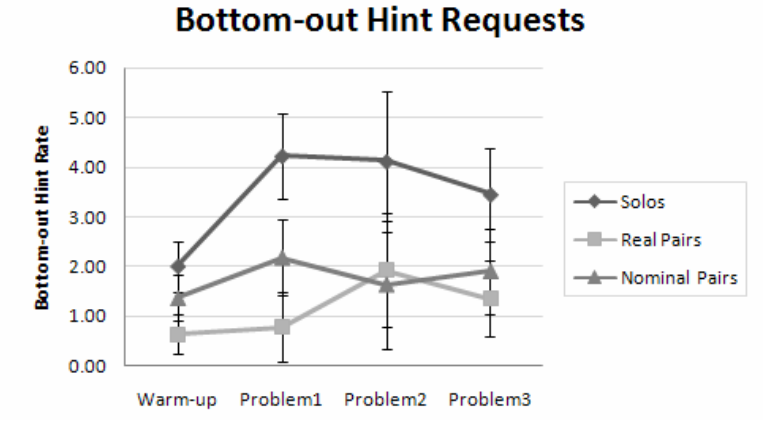


Fig. 2. The mean number of bottom-out hint requests per problem, including nominal pairs

nominal pairs. For the final problem, the nominal pairs (M = 33.54, SD = 9.15) submitted an equal number of entries for the final problem as the real pairs (M = 34.29, SD = 6.72). However, the number of correct entries made for the final problem was replicated. There was a marginal difference between the two groups in terms of the correct number of entries made on the last problem, $F(1, 23) = 3.16$, $p = .09$, $d = .75$. The real pairs (M = 23.29, SD = 5.06) entered marginally more correct entries for the final problem than the nominal pairs (M = 18.73, SD = 7.73).

In terms of the bottom-out hint requests, the pattern of results was also consistent with the solo results (see Fig. 2). The real pairs still requested fewer bottom-out hints for the Problem 1 than the nominal pairs, $F(1, 23) = 5.11$, $p = .03$, $d = .95$. None of the other contrasts reached traditional levels of statistical significance.

4 Discussion

The introduction to this paper proposed three hypotheses regarding collaboration during solving problems with an ITS. The first hypothesis stated that pairs may be in a better position to profit from an ITS's hints than individuals because each student may interpret the hint in a different way. Through the process of sharing and debugging their various interpretations, pairs of students can help each other make sense of the hints. Evidence for this claim can be found in both the completion rate and the use of bottom-out hints. The pairs progressed further into the problem set than the individuals, and they required fewer bottom-out hints to finish the problems.

The second hypothesis stated that a step-based ITS may help students collaborate more effectively. Although, the present study did not directly test this hypothesis (i.e. by contrasting the frequency of successful collaborative processes for step-based tutoring with a more open learning environment), we indirectly tested the hypothesis by conducting a *nominal groups analysis*. Nominal groups were formed by randomly pairing individuals from the solo condition and taking the best performance from each member. For example, if Solo Member A asked for 3 bottom-out hints, and Solo Member B asked for 2, then the score for that nominal group on the bottom-out hint measure was "2." However, if Solo A correctly imputed 8 steps, and Solo B entered 5 correct steps, then the score for that nominal pair was "8." Therefore, the source of the nominal pair's score could come from a different individual for the different measures of problem-solving performance.

The results from the nominal-groups analysis replicated the set of results from the solos. Although the magnitude of the differences between pairs and solos was reduced, the same trend of results was observed. This suggests that there was something special about participating in a collaborative discussion while solving problems with an ITS. That is, the tutoring system helped to scaffold the dialog between interacting students above and beyond the performance of the non-interacting individuals.

Finally, the third hypothesis stated that pairs of students should be less likely to abuse the ITS than individuals because students have a general sense of the proper use of a learning environment. Stated differently, having a partner keeps the individuals honest. Evidence for the third hypothesis was most directly demonstrated with the bottom-out hint requests. Pairs of students requested an average of 67.6% fewer bottom-out hints across the entire 2-hour experiment. The difference in bottom-out hint

requests between the pairs and solos was most pronounced after studying the first example (i.e., Warm-up Problem = 67.9% vs. Problem 1 = 85.6%). This suggests that the pairs may have also been less likely to abuse the examples. Instead of shallowly processing the content, they may have better comprehended and later reused the information in the examples. In the future, we plan to test this hypothesis more directly by analyzing the verbal protocols produced while studying the examples.

In summary, the results from each of the three hypotheses suggest that asking students to solve problems collaboratively, with a step-based tutoring system, is a productive way to enhance learning from an ITS. This study, which served as a positive example of a manipulation check, suggests that future experiments continue to examine the boundary conditions under which collaboration is effective in an ITS. Additional measures of learning need to be used to evaluate the strength of the learning that results from collaboration. For example, the present study does not indicate if learning from collaboration will transfer to individual problem solving and to novel domains. Additional research is needed to answer these and related questions.

Acknowledgements. This work was supported by the Pittsburgh Science of Learning Center, which is funded by the National Science Foundation award number SBE-0354420.

References

1. Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z.: Off-Task behavior in the cognitive tutor classroom: When students game the system. In: Proceedings of ACM CHI 2004: Computer-Human Interaction, pp. 383–390 (2004)
2. Roll, I., Aleven, V., McLaren, B.M., Koedinger, K.R.: Can help seeking be tutored? Searching for the secret sauce of metacognitive tutoring. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Artificial intelligence in education: Building technology rich learning contexts that work, vol. 158, pp. 203–210. IOS Press, Amsterdam (2007)
3. Barros, B., Conejo, R., Guzman, E.: Measuring the effect of collaboration in an assessment environment. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Artificial intelligence in education: Building technology rich learning contexts that work, vol. 158, pp. 375–382. IOS Press, Amsterdam (2007)
4. Biswas, G., Leelawong, K., Schwartz, D.L., Vye, N.: The Teachable Agents Group at Vanderbilt: Learning by teaching: A new agent paradigm for educational software. Applied Artificial Intelligence 19, 363–392 (2005)
5. Rummel, N., Spada, H.: Learning to collaborate: An instructional approach to promoting collaborative problem solving in computer-mediated settings. Journal of the Learning Sciences 14, 201–241 (2005)
6. Walker, E., McLaren, B.M., Rummel, N., Koedinger, K.: Who says three's a crowd? Using a cognitive tutor to support peer tutoring. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Artificial intelligence in education: Building technology rich learning contexts that work, vol. 158, pp. 399–406. IOS Press, Amsterdam (2007)
7. Chi, M.T.H., Bassok, M.: Learning from examples via self-explanations. In: Resnick, L.B. (ed.) Knowing, learning, and instruction: Essays in honor of Robert Glaser, pp. 251–282. Lawrence Erlbaum Associates, Inc., Hillsdale (1989)

8. Renkl, A.: Learning from worked-out examples: A study on individual differences. Cognitive Science 21, 1–29 (1997)
9. Aleven, V.A.W.M.M., Koedinger, K.R.: An effective metacognitive strategy: Learning by doing and explain with a computer-based Cognitive Tutor. Cognitive Science 26, 147–179 (2002)
10. Chi, M.T.H., DeLeeuw, N., Chiu, M.-H., LaVancher, C.: Eliciting self-explanations improves understanding. Cognitive Science 18, 439–477 (1994)
11. Bielaczyc, K., Pirolli, P., Brown, A.L.: Collaborative explanations and metacognition: Identifying successful learning activities in the acquisition of cognitive skills. In: Ram, A., Eiselt, K. (eds.) Proceedings of the Sixteenth Annual Cognitive Science Society Conference, pp. 39–44. Laurence Earlbaum Associates, Hillsdale (1994)
12. Steiner, I.D.: Group processes and productivity. Academic Press, New York (1972)
13. Anderson, J.R., Corbett, A.T., Koedinger, K., Pelletier, R.: Cognitive tutors: Lessons learned. The Journal of the Learning Sciences 4, 167–207 (1995)
14. Sweller, J.: The worked example effect and human cognition. Learning and Instruction 16, 165–169 (2006)
15. Hausmann, R.G.M., VanLehn, K.: Explaining self-explaining: A contrast between content and generation. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Artificial intelligence in education: Building technology rich learning contexts that work, vol. 158, pp. 417–424. IOS Press, Amsterdam (2007)
16. Lorge, I., Fox, D., Davitz, J., Brenner, M.: A survey of studies contrasting the quality of group performance and individual performance. Psychological Bulletin 55, 337–372 (1958)

Theory-Driven Group Formation through Ontologies

Seiji Isotani and Riichiro Mizoguchi

The Institute of Scientific and Industrial Research, Osaka University,
8-1 Mihogaoka, Ibaraki, Osaka, 567-0047, Japan
isotani@acm.org, miz@ei.sanken.osaka-u.ac.jp

Abstract. Group formation plays a critical role in collaborative learning (CL). It affects the acceptance of group activities by learners and the success of the collaborative learning process. Nevertheless, proposing an effective and pedagogically sound group formation is a very complex issue due to the multiple factors that influence group arrangement. The main goal of this paper is to present an ontology that works as a framework based on learning theories that facilitates group formation and CL design. To validate the usefulness and effectiveness of this ontology we present a method to use it and the results of an experiment carried out with four instructors and twenty participants. The results suggest that our ontology can be used adequately and the concepts represented on it can positively affect the performance of individuals during group learning.

1 Introduction

Collaborative learning (CL) has a long history in Education [14]. According to [13], over the past decades the numbers of technologies that enable people to learn collaboratively have increased considerably. In CL, group formation plays a critical role that affects the acceptance of group activities and the success of the learning process. Some researchers claim that an inadequate group formation has been the main reason for many unsuccessful applications that rely on CL [5;6]. Nevertheless, according to [17], only a few CSCL systems provide the functionality for group formation. The large majority focuses on techniques for sharing resources or on improvements of group performance (which does not guarantee an improvement of learning [3]). The policy used by conventional methods concerns situation-independent CL activities where the idea of groups composed by heterogeneous participants is always the best solution. Such policy (lower-level policy) is applicable to any situation without regulation of the group. While it has satisfactorily facilitated the use of group formation in CSCL systems [12], the lower-level policy has difficulties in supporting well-structured groups where each learner has a defined role and learning goal. This limitation may impair the chances of successful learning and complicates the analysis of the CL processes.

To overcome this problem our work deals with a higher-level policy that can be put on top of the lower-level policy to further increase the benefits of CL by bringing structure and context into the group. Thus, the main problem we are addressing is how to propose an *effective* group formation. By *effective* we mean the selection of appropriate information to propose a principled group formation that creates favorable

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 646–655, 2008.

conditions for learners to perform CL activities and helps instructors to more easily estimate the learning benefits of these activities. In order to identify the necessary information for *effective* group formation, our approach relies on achievements of the Learning Science Community (especially learning theories) and those of ontology engineering to support CL [6]. The use of ontologies aims to establish an engineering infrastructure for making theories more understandable, shareable and usable for both computers and humans. Then, we can propose techniques for reasoning on theories, facilitating the development of intelligent authoring tools for CL.

In this paper we, first, overview our theory-driven group formation concept developed to date. Second, we present our ontology and a method to use it to form groups. Finally, to validate the usefulness of this ontology, we present the results of an experiment performed with four instructors that have used our ontologies to form groups with the intent to sharpen the communication skills of twenty participants.

2 Theory-Driven Group Formation

Many learning theories contribute to in-depth understanding and support of CL (e.g. LPP [8]). By selecting an adequate theory, we can provide the rationale justifying that the suggested group formation can help learners to achieve the learning goals. One could disagree that it is possible to support or enhance effective group formation by using learning theories. The authors are aware that theories have some flaws and are not "*watertight.*" However, from our point of view, learning theories can provide some essential conditions in which learners are able to learn more smoothly or effectively. By explaining the learning process, besides trying to explain what happens inside of a learner, a learning theory also gives (explicitly or implicitly), for example, the ***context*** in which the ***learning activities*** have been taking place, the target ***knowledge/skill*** that has been tackled, and the ***roles*** played by learners. Others could think that the use of learning theories to adopt some regulations (suggestions to improve the quality of CL) could harm the CL process. However, according to [3] and [15], effectiveness of CL relies on how well we can understand the multiple factors that influence group interactions and use such understanding to prescribe appropriated learning groups that facilitate meaningful interactions among learners. From such an observation, the use of theories as ***guidelines*** can increase the effectiveness of CL.

To select an appropriate theory for a specific situation is a difficult and time-consuming task. One of the reasons is the difficulty in understanding the theories because of their complexity and ambiguity. Therefore, to allow the rational use of theories to support CL, we must establish a common conceptual infrastructure on which we can clarify, at least partially, what CL is and how learning theories can facilitate the identification of a well thought out group structure. In this context, ontologies have shown significant results to represent educational theories and to use them effectively [9]. In CSCL, a pioneering works in using ontologies to establish a system of concepts for CL, with theoretical support, was presented in [6]. Nevertheless, previous achievements have some room for improvement. Especially, it is difficult to propose group formation in compliance with theories. To overcome such a limitation we have been working to clarify the concepts extracted from theories and to promote the adequate use of these concepts. In the next session, we present some of these concepts and explain how we can use them to propose effective group formation.

3 Ontology-Enabled Group Formation

Our work uses ontologies as a common framework to describe learning theories and CL explicitly and formally. We aim to enable theory-driven group formation that offers guiding principles that link the design of CL activities with interaction analysis. This approach allows the identifying of intended goals, roles, and strategies for a group and its members during the design process. Then, we can more easily analyze individuals' interactions to identify whether the proposed interactions were carried out successfully or not and whether learners attained the expected benefits or not. Finally, with a good analysis of interactions it is possible to acquire knowledge about learners and propose a better group formation afterwards (Figure 1).

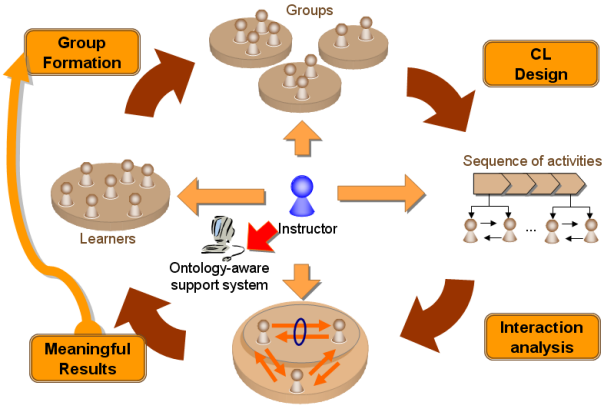


Fig. 1. A full view of the total system of the theory-based group formation and analysis

In the previous work, we extend the CL ontology to represent the relationship among interactions and learner's development, and to propose theory-compliant activities to enhance interactions among learners [7]. In this paper, we offer more expressiveness to this ontology discussing its use to support group formation.

3.1 Main Concepts for Group Formation

This section presents 3 key concepts, extracted from theories, necessary to understand how groups are formed using our ontology: *learning goal* (individual and group goal), *role* and *instructional-learning event*.

According to [1;6;10], although there is a variety of learning goals, the process of a learner's growth can be described as the process of knowledge acquisition and skill development (Table1). Thus, concerning individual goals, the CL ontology describes succinctly the learner's knowledge acquisition process and skill development process.

The process of acquiring specific knowledge includes three stages of learning: accretion, tuning and restructuring [10]. Accretion is adding and interpreting new information in terms of pre-existing knowledge. Tuning is understanding knowledge

through its application in a specific situation. Restructuring is considering the relationships of acquired knowledge and rebuilding the existing knowledge structure.

Considering the development of skills, there are also three stages: the cognitive stage (rough and explanatory), the associative stage and the autonomous stage [1]. The cognitive stage involves an initial encoding of a target skill that allows the learner to present the desired behavior or, at least, some crude approximation. The associative stage is the improvement of the desired skill through practice. In this stage, mistakes presented initially are gradually detected and eliminated. The autonomous stage is the gradual and continued improvement of the skill. In this stage, the learner can perform accurately and quickly the desired behavior. *s(x,y)* is the simplified form of representing the actual stage of the learner: *x* represents the skill development and *y* represents the knowledge acquisition. For instance, *s(0,1)* illustrates that the stage of skill development is *nothing* and the stage of knowledge acquisition is *accretion*.

Table 1. Stages of learning development [6]

Individual goals (I-goal)	Stages of development	Abbreviation	Sources
Acquisition of Content-Specific Knowledge	Nothing	s(x, 0), x=0..4	[10]
	Accretion	s(x, 1) , x=1..4	
	Tuning	s(x, 2), x=1..4	
	Restructuring	s(x, 3), x=1..4	
Development of Skill			[1]
Some Types	Nothing	s(0, y), y=0..3	
- Cognitive skills - Meta-cognitive skills - Skill for self-Expression …	Rough-Cognitive	s(1, y), y=0..3	
	Explanatory-Cognitive	s(2, y), y=0..3	
	Associative	s(3, y), y=0..3	
	Autonomous	s(4, y), y=0..3	

Concerning the description of group goals in the CL ontology, there are four types: knowledge sharing, creating a solution, spread of a skill and knowledge building (or knowledge transmission). These goals are supported by some of the theories we have analyzed. For example, the Cognitive Flexibility theory supports the sharing of knowledge; and the Cognitive Apprenticeship theory supports the spread of skills.

One of the main factors that affect learners' interactions and, consequently, the achievement of learning goals is the ***role*** played by learners. A role provides pedagogical support stating functions, goals, and responsibilities that guide learner's behavior and tend to increase group stability, satisfaction and communication [15]. For example, the role of *"Tutor"* offers benefits for a learner who has knowledge about the content, but does not have much experience in using it. It is because this learner has to explain the content using his own words in order to teach (obtaining a better understanding about it). However, the same role does not bring as much benefit for a learner who understands the content well and teaches it many times. Therefore, we need to know what roles a learner can play in order to support effective group formation. Currently, the CL ontology represents 12 roles and their pre-requisites.

Finally, a learner needs the adequate context to play a role. Context is extracted from each analyzed theory and includes sequence of activities to be performed (interaction patterns [7]), participants to interact with, and so forth. Nowadays, we have analyzed seven learning theories frequently used to support CL (*e.g.* [2;8;16]).

To express the concepts presented in this section, in Figure 2 we show an updated version of our ontological structure developed previously [7]. This structure consists of two main parts: the Learning Strategy and the CL process. The Learning Strategy, composed by the members of a group and the goals of one learner (*I-role*), specifies how (*Y<= I-goal*) the learner (*I-role*) should interact with other members of the group (*You-role*) to achieve his objectives (*I-goal*). For instance, in Cognitive Apprenticeship a learner interacts with other learners to guide them during the resolution of a problem. In this case the learning strategy (*Y<= I-goal*) used by this learner is "*learn by guiding*"; his role (*I-role*) is known as a "*master role*", the role of the learner who receives the guidance (*You-role*) is known as an "*apprentice role,*" and the goals of the learner who guide (*I-goal*) are to acquire cognitive skills (and meta-cognitive skills) at an autonomous level. To play a role effectively, a learner should satisfy some necessary and desired conditions.

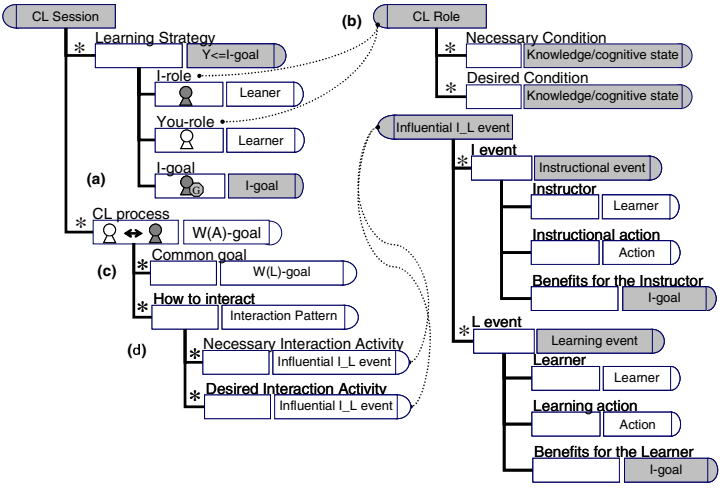


Fig. 2. Part of the Ontological Structure used for group formation

The CL Process (*W(A)-goal*) specifies the goals of the group activity (*W(L)-goal*) and the rational sequence of interactions (*interaction pattern*) provided by theories. The interaction patterns are represented by the necessary and desired interaction activities among members of a group (*e.g.* tutor and tutee). In our ontology, we describe interactions as *I_L events* (instructional-learning event), as presented by [9], for explicitly representing the interaction and its benefits from both points of view: for those who do the action and for those who receive the action. Each event is composed by an actor of an action, the action, and the benefits of the player of this action.

3.2 A Group Formation Method

The question now is how to use the ontology presented in section 3.1 to form groups. A procedural example is shown on Table 2. First of all, the ontology is used as a common vocabulary to set up the CL session. After that, we use the relationship among concepts to identify the best formation that satisfies the session requirements.

Table 2. Procedure to form a group using the ontology presented in Figure 2

Step 1 - Setup a CL session:	**Step 2 – Forming the Groups:**
1.1. To determine what the target individuals have done in the past (experience) and what they can do now (initial levels of knowledge/skills). This step aims to identify the necessities of individuals and the roles they are able to play. **1.2.** Assess the content worth learning and/or the content needed to be learned. The content should be divided in knowledge to be acquired and skills to be developed. The relationships among knowledge-knowledge, knowledge-skill and skill-skill should also be identified. **1.3.** Elect the learning goals expected to be achieved by individuals and/or by the entire group for the specific content. **1.4.** State the initial levels of knowledge/skills and the learning goals of each individual in terms of stages of learning development *s(x,y)* as indicated in Table 1. A more detailed specification of this process will be provided in future papers. Furthermore, each step described previously can be completed (at least partially) by following some instructional design strategies. Some of them can be found in [11].	There are many possibilities to form a group. Let us explore one way concerning individual goals. **2.1.** Match the individuals' goals with a CL session by looking in the *I-goal* (Figure 2a). If no match is found, it means that the theories represented in our ontology cannot help the improvement of the specific goal. However, usually there is more than one session that can help learners to achieve their goals. **2.2.** Check if learners have the necessary and desired conditions to play a role (Figure 2b). Learners with all the conditions have high-priority to join the group; learners with only the necessary conditions have low-priority; and the other learners cannot join the group, because they could harm the CL process. **2.3.** Set the group goal (*common goal*) as shown in Figure 2c; and design CL activities according to the interaction patterns that are described or prescribed by theories (Figure 2d). These patterns can be followed by learners in order to obtain the desired individual and group goals. In previous works we have shown how to design CL activities using this ontology.

Note that, unlike other approaches, the method of group formation using ontologies can provide the rationale for each choice made to form a group providing pedagogical justifications. We can support instructors by explaining why some learners should collaborate and why others should not; it is also possible to help them to set reasonable goals for learners and for the entire group considering the theoretical point of view, the learners' pre-conditions and the content to be learned; and we can ask learners to play specific roles in order to produce a more sophisticated collaboration.

4 Experiment

With the objective of obtaining information about the impact of forming groups using the theory-driven group formation with our ontologies, we designed an experiment as a proof of concepts. The main goals of the experiment were to gather information and verify (a) whether instructors can use the concepts contained in the ontology adequately, and (b) if the framework of the group formation suggested by the ontology is really relevant to the success of the CL session.

The study was carried out with 2 pairs of instructors, each pair from a different institution, and 20 participants who are expected to develop information sharing and self-expression skills. The participants are from 7 different countries of Latin

America, pursuing different degrees in Japan and between the ages of 18 and 35 years old. We chose such an ill-structured environment for two main reasons: (a) these participants have been working together since 2004, but have been suffering from problems in collaborating and sharing information; and (b) in an ill-structured environment, it is easier to identify when a set of changes in the CL settings affects the success of the CL process. We expended about 2 months to complete the whole experiment.

The experiment consists of two phases. The first phase was the planning (set up) of the CL session and the second phase was its actual execution. In the first phase, instructors were asked to deal with the group problem using their own methods. After that, they should find an agreement and select or merge some of the created CL sessions. We specifically asked the instructors to give details about the content to be learned by the participants, their choices to form groups, to define goals, and to create a sequence of activities (including tools to be used). Next, the same tasks were done using our ontology with methods similar to those proposed in section 3.2.

The second phase was the application of the proposed sessions. For each CL session, about half of the participants used the scenario proposed by instructors without support of our ontology (controlled groups), and the other half used the scenario with ontological support. All groups (experimental and controlled) received support of instructors during the activities. For each session, different participants were selected to join the experimental groups according to the necessary requirements described in the ontology. All sessions were recorded and evaluated by both instructors and participants who filled out questionnaires after the sessions.

In total, it was created four CL sessions. The first one, which the main goal was to spread a specific knowledge among participants, was performed in pairs where the more knowledgeable participant should "*teach*" the content to the less knowledgeable one. Four groups followed a Peer Tutoring based CL session [4], and six groups where controlled groups that did not have any specific guideline. In the second session, the main goal was to improve skills of self-expression. It was created five groups with four members each. Three groups followed a Cognitive Flexibility based CL session [16] where learners had to expose their opinions from different perspectives. The third and fourth sessions were based in mind maps constructions and the main goal was to improve the cognitive and meta-cognitive skills and again skills for self-expression. It was created four groups with five members each. One group

Table 3. Some Interactions and their benefits for two groups based on different theories

Interaction	**Expected benefits** (From→To)		**Learning Theory**
	Role A	**Role B**	
	Master	**Apprentice**	Cognitive Apprenticeship [2]
Demonstration	s(3, 2)→s(4, 2)	s(0, x)→s(1, x); s(1, x)→s(2, x); x=0,1,2	
Instigating thinking	s(3, 2)→s(4, 2)	s(1, x)→s(2, x); x=0,1,2	
Monitoring/Coaching	s(3, 2)→s(4, 2)	s(1, x)→s(2, x); s(2, x)→s(3, x); x=0,1,2	
	Full Participant	**Peripheral participant**	LPP [8]
Requesting details	s(3, 2)→s(3,3)	s(0,x)→s(1,x); x=0,1,2	
Instigating discussion	s(3, 2)→s(4,3)	s(1,x)→s(3,x); x=0,1,2	
Exchanging information	s(3, 2)→s(4,3)	s(1,x)→s(3,x); x=0,1,2	

followed the Cognitive Apprenticeship CL session [2] with one master and four apprentices; and another one followed the LPP CL session [8] with two full participants and three peripheral participants. The group that followed Cognitive Apprenticeship theory had activities such as demonstration and guided tasks. Although the final goals were the same, the group that followed LPP theory had activities such as discussions and exchange of ideas. In Table 3, we show some interaction between learners and their educational benefits.

5 Results and Discussion

The interface between instructors and ontologies was mediated by the authors. The intention was to capture the necessities of users and to check the usefulness of concepts in our ontologies (and not the usefulness of a system built using ontologies). With the encouraging feedback and data obtained in the experiment, we believe it will be feasible to develop a complete ontology-aware system for CL as shown in Figure 1.

Concerning the first phase (planning), all the instructors agreed that the use of the ontology was quite helpful in obtaining a good insight about the group formation. It was discovered that many unconscious choices of instructors, in fact, have been explicitly represented in our ontology. Furthermore, instructors have considered it very informative and meaningful that the concepts in our ontology were linked with the relevant theory. Besides, it gives the rationale behind each choice to form a group and to design CL activities; in some cases, the instructors could select the theory they felt more comfortable working with. Another benefit pointed out by instructors was the facility to create and to share CL sessions. When each instructor produced their own sessions/scenarios using their own vocabulary, it was quite difficult to discuss the benefits of each one in order to find a common agreement and to merge them. Using the ontology, the sessions described by one instructor were comprehensible by the others with only small misunderstandings. Finally, the ontology was used only as guideline to help instructors propose groups with theoretical justification, thus, the instructors had the flexibility to not rely too much on the theories and add the characteristics they think the groups need in order to work effectively. It shows that the use of the ontology did not restrict instructors' action or their creativity. Instead, it helped them to focus on the main problem and to make efforts in parts where their expertise was required the most. For example, after using the information in the ontology, some participants were able to join the experimental groups and, because each session required different learner's conditions, usually we had different participants in these groups. However, sometimes there were too many participants, who could join the experimental groups. Then, instructors also had to consider: the language (to facilitate self-expression), educational background and culture (to increase heterogeneity), previous relationships with other participants (to avoid meaningless interactions), and intrinsic behavior of participants.

In the second phase, we tried to verify the differences between the controlled groups and the groups formed using our ontology (experimental groups). For each CL session, instructors checked how the participants have interacted with each other, the groups' achievements, and the benefits obtained by individuals, besides other indicators. As a result, it was observed that in most of the sessions the participants in the experimental groups had more improvement in the desired skills and the performance of the whole group was better, if compared with the controlled groups. Instructors

observed that, in the controlled groups, half of the scheduled time of some sessions was filled with meaningless interactions instead of performing the necessary activities that would improve the desired skills. Furthermore, it was noted that on many occasions, members of experimental groups who had worked well together in previous sessions could not work together in controlled groups, harming the CL process. One explanation is that in the experimental groups, participants were chosen adequately (rather than randomly, as it usually happens), had defined roles and could follow well structured interaction patterns. As many studies have shown, following these regulations can decrease the chances of undesirable interactions occurring.

We observed that the experimental groups were effective in achieving the desired results. Most of the participants who joined these groups achieved their individual goals and the groups performed effectively. For example, in the session shown in Table 3, the group had as a group goal to spread the skill for building a mind map and, as one of the individual goals, the master had to develop this skill in the autonomous stage (increase his ability to build a map) while the apprentices had to develop the same skill in the associative stage (learn how to build a map adequately). The master helped the apprentices by externalizing his cognitive processes while building maps and monitoring apprentices. On one hand, the master acquired the desired goal. And on the other hand, by observing, imitating and being monitored, the apprentices developed the desired skill effectively. However, participants in the experimental groups complained that it was difficult to follow the appointed role/strategy. They argued that sometimes they had to neglect their personal behavior to get the task done as required. Those complaints are reasonable and will be taken into consideration to improve our ontology. In this same session, although some members of the controlled groups achieved their individual goals, the groups could not achieve their desired goal.

The results in this experiment suggest that the ontology-based framework of group formation can be used adequately to form effective groups. This verification is essential in order to provide intelligent systems with theoretical knowledge that clarify how learning theories can help instructors to form groups, to design CL activities and to enhance learning outcomes. The ontology presented in this work aims to represent the knowledge of intelligent educational systems that support CL, playing a central role in the decision making about ***how***, ***when,*** and ***why*** we should use theories to form groups considering the multiple factors that influence the CL process.

6 Conclusions

In this paper we focused our discussion on the necessity of sophisticated group formation to set roles, goals, and activities for learners before a CL session starts. To propose effective groups, it is helpful to have a clear and sharable understanding about many learning theories and their features. However, it is very difficult for users (e.g. instructors) to have such a common understanding. Our approach calls upon techniques of ontological engineering to build ontologies that represent, explicitly and formally, the main concepts of each theory which are obtained by our interpretation of theories from group formation perspectives. We then proposed a method for using those concepts adequately. And finally, we conducted an experiment to check the usefulness of our ontology in an ill-structured environment. The results of the experiment indicate that the concepts in the ontology helped instructors to form groups and to design CL activities with theoretical justifications. Furthermore, the results also suggest that individuals in experimental

groups, where each member was carefully selected and the interactions were partially moderated following the prescriptions in the ontology, performed and learned better than in controlled groups whose members were not selected so rigorously and could interact freely with others.

We believe this is a step forward in the development of the foundations of an intelligent authoring tool for CL, with a well grounded theoretical knowledge, that supports group formation, facilitates the design of CL activities, and minimizes the load of interaction analysis (Figure 1). Our ultimate goal is to develop this tool.

References

1. Anderson, J.R.: Acquisition of Cognitive Skill. Psychological Review 89(4), 369–406 (1982)
2. Collins, A.: Cognitive apprenticeship and instructional technology. Educational values and cognitive instruction. LEA, 121–138 (1991)
3. Dillenbourg, P.: Over-scripting CSCL: The risks of blending collaborative learning with instructional design. In: Three worlds of CSCL. Can we support CSCL?, pp. 61–91. Open University Nederland, Heerlen (2002)
4. Endlsey, W.R.: Peer tutorial instruction, Educational Technology (1980)
5. Fiechtner, S.B., Davis, E.A.: Why Some Groups Fail: A Survey of Students' Experiences with Learning Groups. Organizational Behavior Teaching Review 9(4), 75–88 (1985)
6. Inaba, A., Supnithi, T., Ikeda, M., Mizoguchi, R.: How Can We Form Effective Collaborative Learning Groups. In: Proceedings of Intelligent Tutoring Systems, pp. 282–291 (2000)
7. Isotani, S., Mizoguchi, R.: Deployment of Ontologies for an Effective Design of Collaborative Learning Scenarios. In: Haake, J.M., Ochoa, S.F., Cechich, A. (eds.) CRIWG 2007. LNCS, vol. 4715, pp. 223–238. Springer, Heidelberg (2007)
8. Lave, J., Wenger, E.: Situated Learning: Legitimate peripheral participation. Cambridge University Press, New York (1991)
9. Mizoguchi, R., Hayashi, Y., Bourdeau, J.: Inside Theory-Aware and Standards-Compliant Authoring System. In: Proceedings of the Workshop on Ontologies and Semantic Web for E-learning, pp. 1–18 (2007)
10. Rumelhart, D.E., Norman, D.A.: Accretion, Tuning, and Restructuring: Modes of Learning. Semantic factors in cognition. LEA, 37–53 (1978)
11. Romiszowski, A.J.: Designing Instructional Systems. Nichols Publishing Company, New York (1981)
12. Soh, L., Khandaker, N., Jiang, H.: I-MINDS: A Multiagent System for Intelligent Computer-Supported Collaborative Learning and Classroom Management. Journal of Artificial Intelligence in Education 18(2) (2007)
13. Soller, A., Martínez-Monés, A., Jermann, P., Muehlenbrock, M.: From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning. Journal of Artificial Intelligence in Education 15(4), 261–290 (2005)
14. Stahl, G., Koschmann, T., Suthers, D.: CSCL: An historical perspective. Cambridge Handbook of the Learning Sciences, pp. 409–426. Cambridge Press, Cambridge (2006)
15. Strijbos, J.W., Fischer, F.: Methodological challenges for collaborative learning research. Learning & Instruction 17(4), 389–393 (2007)
16. Spiro, R.J., Coulson, R.L., Feltovich, P.J., Anderson, D.K.: Cognitive flexibility theory: Advanced knowledge acquisition in ill-structured domains. In: Proceedings of the Annual Conference of the Cognitive Science Society, pp. 375–383 (1988)
17. Wessner, M., Pfister, H.: Group formation in computer-supported collaborative learning. In: Proceedings of ACM CSCW, pp. 24–31 (2001)

Self-assessment in Vocabulary Tutoring

Michael Heilman and Maxine Eskenazi

Language Technologies Institute, Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
http://www.lti.cs.cmu.edu

Abstract. To individualize instruction, a tutor must infer which knowledge components the student knows and does not know. Self-assessments, by which the student directly reports to the tutor whether a certain item is known, are a fast measure of student knowledge. We investigate their use to initialize a learner model used to individualize instruction ESL vocabulary. Experimental results indicate that self-assessments can be useful measures of knowledge for use in a tutoring system for vocabulary. Self-assessments appear to be particularly reliable when learners claim that words are not known.

1 Introduction

We explore the use of self-assessments in REAP[1], a tutoring system for vocabulary current aimed at intermediate and advanced English as a Second Language (ESL) students. REAP provides practice readings containing target vocabulary words, as well as easy dictionary access and post-reading practice vocabulary exercises. The system individualizes and adapts by choosing texts at the appropriate reading level, and by considering knowledge estimates for target words when selecting readings. The REAP tutor uses a learner model to estimate knowledge of specific target words. Evaluations of student performance on post-reading multiple choice cloze, or fill-in-the-blank practice exercises provide data for the student model. Brown, Frishkoff, and Eskenazi discuss the use and generation of cloze questions for the REAP tutor [1]. A pre-test, however, is utilized to measure initial knowledge prior to the practice readings. Heilman, Eskenazi, Collins-Thompson, and Callan provide more details on the REAP tutor [2].

The primary motivation for self-assessments is to minimize the amount of time required for initial assessments while maintaining accurate measures of knowledge. A challenge of efficiently assessing initial knowledge is that there are a very large number of words which might be taught. For example, Coxhead's Academic Word List [3] has 570 head words. Assessing knowledge of every word, or even a substantial fraction of them, with cloze questions or other performance-based methods would take a considerable amount of time. The Yes/No Test [4] is a test consisting of self-assessments for vocabulary, but it is designed to infer the overall size of a person's receptive vocabulary rather than knowledge of particular

[1] For more information on REAP, see http://reap.cs.cmu.edu.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 656–658, 2008.

words. Rather than dealing with general language skills, we investigate the use of self-assessments for measuring individual word knowledge. Specifically, we investigated whether self-assessments are faster assessments than cloze questions, and also whether self-assessments and cloze questions agree with each other.

2 Experimental Design

We conducted an *in vivo* study with the REAP Tutor involving forty-two adult ESL students of a variety of native languages whoe were part of an upper-intermediate English Reading course in the Summer of 2006 at the University of Pittsburgh. The pre-test and subsequent instruction covered ten words randomly chosen from Coxhead's Academic Word List [3]: *acknowledge*, *demonstrate*, *controversy*, *identical*, *retain*, *precise*, *undergo*, *sacred*, *cease*, and *outcome*. The pre-test included both cloze and self-assessment questions for each word, in a different random order for each student. The self-assessment questions asked students to make a binary decision as to whether or not they knew a given word (e.g., Do you know the word "sacred"?). For all cloze questions in this study, students had to choose from a word bank of twenty words that included the correct target word and a set of foils that were other target words used in REAP. The reliability of the cloze questions and self-assessments were evaluated by a follow-up cloze questions, with different base sentences, given just before the first instructional opportunity for each word. Due to time constraints, data for only about 330 of a possible 420 pre-test and follow-up question pairs were available for each type. In our analysis, we assume that an agreement of the follow-up assessment with the pre-test assessment is an indication of accuracy of a pre-test assessment, whether self-assessment or cloze. Response times were recorded automatically by the tutoring system.

3 Results and Discussion

On the pre-test, students spent 38.8 seconds (N = 329 questions, SD = 29.9) per cloze question. In contrast, students spent only 6.1 seconds (N = 331, SD = 5.2) per self-assessment. A two-tailed t-test for independent samples with unequal variance indicates the difference is statistically significant (p <0.001).

Pre-test cloze questions agreed with the follow-up cloze questions 74.7 percent of the time. That is, approximately three quarters of the time, the student either answered both correctly or both incorrectly. In contrast, self-assessments agreed with cloze questions only 56.4 percent of the time, which is barely above random chance. The upper half of Table 1 is a contingency table for pre-test cloze and follow-up cloze questions, and the lower half Table 1 is a contingency table for pre-test self-assessments and follow-up cloze questions. The cells in each table are the percentages of question pairs in a particular case. The self-assessments appear to be accurate when students claim that a word is unknown. When students claimed to not know a word, they answered follow-up cloze questions incorrectly 92% of the time, as indicated in the lower half of Table 1. When

students claimed to know a word, however, they correctly answered the follow-up cloze question for that word only 29% of the time. These findings suggest that self-assessment of knowledge of particular vocabulary words are accurate when students claim not to know a word. Recent versions of the REAP tutor use self-assessments to initialize learner models by setting different initial probabilities of knowledge based on whether the student claims to know a word.

Self-assessments of individual target vocabulary words can be much faster than other assessments such as cloze questions. Also, self-assessments appear to be particularly reliable when students claim that they do not know particular words. Thus, it appears that data from self-assessments can be effectively used with a learner model to make inferences about word knowledge in order to individualize instruction.

Table 1. Contingency Table for 331 Pairs of Pre-test Cloze and Follow-up Questions (upper) and 329 Pairs of Pre-test Self-Assessment and Follow-up Questions (lower)

	Cloze-Correct	Cloze-Incorrect	Total
Follow-up-Correct	67%	18%	20%
Follow-up-Incorrect	33%	82%	80%
TOTAL	15%	85%	100%

	Self-Assessment-Known	Self-Assessment-Unknown	Total
Follow-up-Correct	29%	8%	20%
Follow-up-Incorrect	71%	92%	80%
TOTAL	56%	44%	100%

Acknowledgements

This work was supported by the Pittsburgh Science of Learning Center which is funded by the National Science Foundation, award number SBE-0354420; by NSF grant IIS-0096139; and Dept. of Education grant R305G03123. Any opinions, findings, conclusions or recommendations expressed in this material are the authors', and do not necessarily reflect those of the sponsors.

References

1. Brown, J., Frishkoff, G., Eskenazi, M.: Automatic question generation for vocabulary assessment. In: Proceedings of HLT/EMNLP 2005, Vancouver, B.C (2005)
2. Heilman, M., Collins-Thompson, K., Callan, J., Eskenazi, M.: Classroom success of an Intelligent Tutoring System for lexical practice and reading comprehension. In: Proceedings of the Ninth International Conference on Spoken Language Processing (2006)
3. Coxhead, A.: A New Academic Word List. TESOL Quarterly 34(2), 213–238 (2000)
4. Meara, P., Buxton, B.: An alternative to multiple choice vocabulary tests. Language Testing 4, 142–145 (1987)

Automatically Generating and Validating Reading-Check Questions

Christine M. Feeney[1] and Michael Heilman[2]

[1] University of Virginia, Charlottesville, Virginia, USA
[2] Language Technologies Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA

Abstract. The number of texts used by the REAP tutor and challenges of automatically generating reading comprehension questions necessitated a simpler automated type of question. We describe an algorithm for generating such reading-check questions for arbitrary texts. Two studies investigating the utility of these questions found reliable, moderate correlations with vocabulary learning and reading comprehension.

Keywords: reading comprehension, question generation, language tutoring.

1 Introduction

Computer-based instructional systems often include text instruction as a significant component. In particular, language tutoring systems often integrate texts with practice tasks. For example, the REAP[1] tutor for English as a Second Language vocabulary asks students to read authentic texts containing target words to expose them to contextualized implicit information [1]. Estimates of student comprehension of these chosen texts can be useful for many purposes: as experimental data, input for a learner model, or as an indicator that the tutoring system should select easier texts.

One type of possible measure is the deep comprehension question, which requires inference and integration of concepts represented in the text. This question type often appears on standardized tests of verbal ability. Shallower comprehension questions might simply require the recall of facts from a text. A simpler measure which can be automated, which we call a "reading-check" question, requires recall of simple surface features of the text-for example, specific lexical items.

Ideally, a system would employ deep comprehension questions. However, generating such questions is currently extremely challenging. Systems have been designed for generating comprehension questions from texts, such as in work described by Kunichika, Katayama, Hirashima, and Takeuchi [2]. Also, Li and Sambasivam [3] describe a method for generating questions from an ontology, or domain specific knowledge base, rather than from a specific text. These systems,

[1] For more information on REAP, see http://reap.cs.cmu.edu.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 659–661, 2008.

however, are not simultaneously error-free, domain-general, and completely automatic. Tutoring systems such as the REAP tutor require automaticity and domain generality because they employ a large number of texts covering a variety of topics. Therefore, the simpler reading-check questions are employed because they are essentially error-free and thus can be completely automated.

Automatically generated reading-check questions ask the student to choose, from among set of foils, a small set of words that appear in a particular text that he has read. For a given text, the algorithm first extracts a list of unique words appearing in the text. It then calculates a measure of the salience of each word by evaluating the relative frequency of the word in the text minus the relative frequency of the word in general English, divided by the relative frequency of the word in general English. The top N words are then chosen for the answer set ($N = 8$ in these experiments). Foils are generated by taking the answer set and replacing half of the words with randomly chosen words that did not appear in the text.

2 Reading-Check Questions and Vocabulary Learning

We conducted a study to measure vocabulary learning with the REAP tutor. Forty-four students at the English Language Institute at the University of Pittsburgh participated in this experiment as part of an intermediate English as a Second Language Reading course in the Fall of 2006. After nine training sessions with REAP, students took a post-test consisting of twenty cloze, or fill-in-the-blank, vocabulary questions and ten sentence production tasks for target vocabulary words. Statistically reliable correlations were found between performance on reading-check questions during training sessions and post-test cloze scores ($r = .547$, two-tailed test of independent samples, $t(31) = 3.64$, $p = .001$) and sentence production tasks ($r = .536$, $t(31) = 3.53$, $p = .011$). These results indicate reliable associations between reading-check performance and two measures of vocabulary learning. The reading-check questions seem to measure a construct that facilitates vocabulary acquisition while reading practice texts.

3 Association of Reading-Check Questions to Comprehension Questions

We conducted a follow-up study to attempt to determine whether performance on automatically-generated reading-check questions correlates with performance on more sophisticated, manually-authored reading comprehension questions. Thirty undergraduate students attending summer research programs at Carnegie Mellon University participated in the lab study. All were native speakers of English in order to reduce the number of confounds (e.g., native language) and ensure at least partial comprehension of the texts. The study employed a between-participants design. Participants read five passages, each followed by four reading-check questions, and three to five expert-written reading comprehension questions. Passages were between 500 to 1,000 words and of varying

levels of difficulty. When looking at the participants' percentages of correct responses for both question types, the two variables had a medium positive correlation of .366 (p <.0005, one-tailed test). The R^2 value of 0.13 means that overall, thirteen percent of the variance of performance in answering reading comprehension questions can be explained reading-check performance.

4 Discussion

The first study revealed a significant correlation between performance on automated reading-check questions and vocabulary learning in the REAP tutor. A second experiment investigated the extent to which reading-check questions are associated with measures of reading comprehension. The study revealed a reliable, but not extremely strong, correlation between reading-check performance and reading comprehension, suggesting that the reading-check questions measure a construct that is associated with but not equivalent to comprehension. The most plausible explanation for the findings of the two studies seems to be that the reading-check questions measure reader attention. If students are attending to a text, then they are more likely to comprehend it. Similarly, students attending to a text are more likely but not certain to process implicit information in the text about target vocabulary, as seen in the first study. From this inference, it seems valid to continue to use reading-check questions to facilitate learning within the REAP tutoring system.

Acknowledgements

The authors would like to thank Gregory J. Mizera and Dr. Maxine Eskenazi. This work was supported by Dept. of Education grant R305G03123, the National Science Foundation grant 354420 to the Pittsburgh Science of Learning Center. Any opinions, findings, conclusions or recommendations expressed in this material are the authors', and do not necessarily reflect those of the sponsors.

References

1. Heilman, M., Collins-Thompson, K., Callan, J., Eskenazi, M.: Classroom success of an Intelligent Tutoring System for lexical practice and reading comprehension. In: Proceedings of the Ninth International Conference on Spoken Language Processing (2006)
2. Kunichika, H., Katayama, T., Hirashima, T., Takeuchi, A.: Automated question generation method for intelligent English learning systems and its evaluation. In: Proceedings of ICCE 2004 (2003)
3. Li, T., Sambasivam, S.: Automatically Generating Questions in Multiple Variables for Intelligent Tutoring. The Journal of Issues in Informing Science and Information Technology (2), 471–480 (2005)

Dynamic Browsing of Audiovisual Lecture Recordings Based on Automated Speech Recognition

Stephan Repp, Andreas Groß, and Christoph Meinel

Hasso-Plattner-Institut für Softwaresystemtechnik GmbH, Prof.-Dr.-Helmert-Str. 2-3
D-14482 Potsdam, Germany

Abstract. The number of digital lecture video recordings has increased dramatically since recording technology became available. The accessibility and the search inside of this large archive are limited and difficult. Manual annotation and segmentation is time-consuming and useless. A promising approach is based on using the audio layer of a lecture recording to get information about the lecture contents. In this paper, we are presenting a retrieval method and a user-interface based on existing recorded lectures. A deficient transcription from a speech recognition engine (SRE) is sufficient for browsing in the video-archive. A user-interface for dynamic browsing of the e-learning contents is presented and an evaluation of the supplied keywords concludes the paper.

1 System

Audiovisual recordings in terms of streaming media are used more and more for correspondence course institutions [1]. Independent of time and place learners have access to libraries of recorded lectures. But the accessibility and traceability of their content for further use is rather limited. Two major challenges arise while preparing recorded lectures for content based retrieval: automated indexing of multimedia videos and the retrieval of semantically appropriate information from a lecture knowledge base. The requested information is often covered by a few minutes of the lecture recording and is therefore hidden within a full 90 minute recording stored among thousands of others. But, how to retrieve the appropriate information in a large lecture video data base in a more efficient way? (Manual) segmentation of video lectures into smaller units, each segment related to a specific topic, is an accepted approach to finding the desired piece of information [1,3,4]. It has been shown in [1] that a keyword-based search in an imperfect transcript yields reliable results. But such solutions fail if word sense disambiguation is required.

In this paper, we present our efforts at putting together results from different fields and projects in order to create a user-interface for browsing the educational lecture video archive based on the transcripts of a SRE.

[1] e.g. http://www.tele-task.de/

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 662–664, 2008.

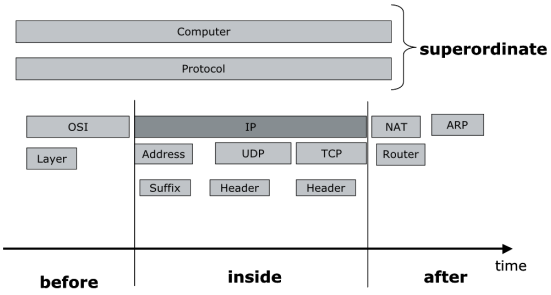


Fig. 1. The *inside*, *before*, *after* and *superordinate* area for the query *IP*

The **system** is organized in three basic functional modules:

Module 1:
This module is discussed in detail in [2]. The transcript consists of a list of words with the corresponding point in time when the word was spotted in the speaker's flow of words.

Module 2:
The input of this modul are the words from module 1 and the output data is an index of the term-chains. Clustering is used to detect cohesive areas (chains) in the transcript. A chain is constructed to consist of all repetitions ranging from the first to the last appearance of the term in the lecture. The chain is divided into subparts when there is a long hiatus between the terms [2].

Module 3:
This well-known module consists of a web-server and a web-browser.

In a retrieval **experiment** 153 topic words (e.g.: XML, topology...) are used for evaluate the first relevant chains (the chain with the maximum word number). 79 percent of all hits are correct and only 21 percent are completely wrong or not in the correct time. The generated chain-index supports a dissolving of the ambiguity: In Figure 1 the term *UDP* is used in the context of *protocol*. The sense of this word is clear from the context it is used in. Moreover, the index supports a keyword register for each video-segment. In Figure 1, the search query of the user is *IP* and the chain with the highest word-nummber of the chain and their inside chains (*Address, UDP, Suffix, Header, TCP*) is returned. The inside chains represent the content of the video-segment *IP*. In fact, a chain has a *before*, *after* and *superordinate* area too. These areas consist themselves of areas. The user has the opportunity to browse through these areas to find the semantically proper position in the video. Furthermore, the problem of word- composite can be solved with the help of the chain-index.

The **user-interface** consists of three regions: The **first region** is set for the input of the search query. The **second region** is the result of the search. The **third region** shows the summary of the *before*, *after* and *inside* areas. The user can start a video with a click on the adequate chain term. After that, an external

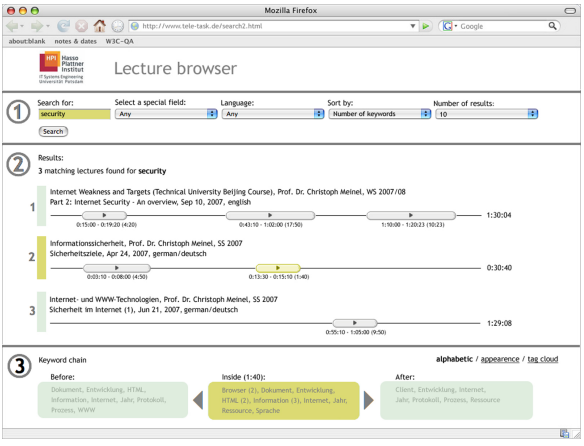


Fig. 2. User Interface with time information

player plays the video-segment and additionally, the third region of our site is expanded to this chain with its *before*, *after* and *inside* areas.

It is clear that the **results** produced by such a search tool depend on the accuracy of the SRE. But some wrongly detected words and wrong compound words are not a problem for our retrieval system and user-interface. The term occurs very often in relevant chains and so a high redundancy exists. Thus, some wrongly detected words have no influence on the generated and final result [1]. Nevertheless, the user-interface allows an exact, easy and fast navigation in the video archive. In addition, it allows the disambiguation of words. The results demonstrate how surprisingly well the browsing and disambiguation works. The result shows that it is possible to add data to the result set that supports the students with the helpful information they are searching for. Our system needs no additional resourses. The lecture-browser supports simple navigation through the corpus of lectures.

References

1. Hürst, W., Kreuzer, T., Wiesenhütter, M.: A qualitative study towards using large vocabulary automatic speech recognition to index recorded presentations for search and access over the web. In: IADIS Internatinal Conference WWW/Internet (ICWI), pp. 135–143 (2002)
2. Repp, S., Meinel, C.: Semantic indexing for recorded educational lecture videos. In: 4th IEEE PerCom 2006 Workshops, March 13-17, 2006, pp. 240–245. IEEE Computer Society Press, Pisa,Italy (2006)
3. Repp, S., Waitelonis, J., Sack, H., Meinel, C.: Segmentation and annotation of audiovisual recordings based on automated speech recognition. In: IDEAL 2007, Birmingham, UK, December 16-19,2007. LNCS, pp. 620–629. Springer (2007)
4. Yamamoto, N., Ogata, J., Ariki, Y.: Topic segmentation and retrieval system for lecture videos based on spontaneous speech recognition. In: Proceedings of the 8th European Conference on Speech Communication and Technology, EUROSPEECH, pp. 961–964 (September 2003)

Agent-Based Framework for Affective Intelligent Tutoring Systems

Mahmoud Neji[1], Mohamed Ben Ammar[2], Adel. M Alimi[2], and Guy Gouardères[3]

[1] Faculté des Sciences Economiques et de Gestion, University of Sfax, –Tunisia
Mahmoud.Neji@fsegs.rnu.tn
[2] REsearch Group on Intelligent Machines (REGIM)
University of Sfax, ENIS
P.O.Box. W-3038 – Sfax – Tunisia
adel.alimi@ieee.org, ben_ammar@acm.org
LIUPPA, IUT de Bayonne –Université de Pau et des Pays de l'Adour
64115 Bayonne, France
Guy.gouarderes@iutbayonne.univ-pau.fr

Abstract. Affective Computing is a new Artificial Intelligence area that deals with the possibility of making computers able to recognize human emotions in different ways. This paper represents a study about the integration of this new area in the intelligent tutoring system. The main goal is to analyses learner facial expressions and show how Affective Computing could contribute for this interaction, being part of the development of computer systems where information about learner' emotion would be helpful.

Keywords: Affective Computing, intelligent tutoring system, learner' emotion.

1 EMASPEL Framework

An Intelligent Tutoring System (ITS) is a computer-based educational system that provides individualized instruction like a human tutor. A traditional Intelligent Tutoring System decides how and what to teach based on the learner pedagogical state. However, it has been demonstrated that an experienced human tutor manages the emotional state (besides the pedagogical state) of the learner to motivate him and to improve the learning process. Therefore, the learner model structure needs to be augmented to include knowledge about the affective state. The ITS needs the ability of reasoning about the affective state to provide learners with an adequate response from a pedagogical and more precisely affective point of view; that's why we require the affective e-learning system that it has two main functions: i) to infer the affective learner state; and ii) to establish the optimal tutorial action considering the learner affective state. In this way, our proposed framework improve learning within our virtual learning environment by means of a more personalized environment through recognizing the learners' affective state with the aim of reacting appropriately from a pedagogical and affective point of view. The affective system considers the learner affective state and the tutorial situation to establish the affective action (via the EECA). The affective action helps the tutor to establish the next pedagogical action

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 665–667, 2008.

based in the knowledge base (KB), and it also helps to the curriculum agent to establish the physical realization of the pedagogical action based in the DB1 and DB1. So the learner receives a tutorial action with an affective component and a pedagogical component, which is our main contribution in this paper. The other novelty of our paper is the use of the multi-agent methodology that can certainly bring several advantages to the development of e-learning systems since it deals well with applications where such crucial issues (distance, cooperation among different entities and integration of different components of software) are found. As a result, multi-agent systems, combined with technologies of networking and telecommunications, bring powerful resources to develop the affective e-learning systems. So, in this research work, we propose affective framework for an intelligent affective system.[This framework: EMASPEL (Emotional Multi-Agents System for Peer-to-peer E-Learning) [1] (fig.1), where we have integrated five kinds of agent (Interface agent, emotional agents, EEC agents, curriculum agent, and tutoring agent) in order to promote a more dynamic and flexible affective communication between the learner and the affective system.

1.1 The Emotional Agents

Facial expression is a fundamental carrier of emotional information and is used widely in all cultures and civilizations to express as well as perceive emotion. In addition, to make effective communication between an EECA [2] and a learner, they need to be able to identify the other's emotion state through the other's expression and we call this task emotion identification established by the emotional agents. Extracting and validating emotional cues through analyzing the learner's facial expressions is of high importance for improving the level of interaction in man-machine communication systems. Extraction of appropriate facial features and consequent recognition of the learner's emotional state that can be robust to facial expression variations among different learners is the topic of these emotional agents.

The emotional agents have been successfully integrated in a learning environment and aims at capturing and managing the emotions expressed by the learner during a learning session. They recognized the learner emotional state by capturing currently emotions that he or she expressed during learning activities and sent it to the EECA [3] Nevertheless, to develop a system that interprets facial expressions is difficult. Two kinds of problems have to be solved: facial expression feature extraction and facial expression classification. Facial features' extraction uses a standard webcam and requires no specific illumination or background conditions. Emotional classification is based on the variation of certain distances from the neutral face and manages the six basic universal emotions of Ekman [4].

1.2 Emotional Embodied Conversational Agent

The affective virtual character (EECA) was integrated to a learning environment, communicating with the learners in verbal and in non verbal language such as facial expression, suggested readings according to the activities being performed. When a learner needs assistance to learn a given topic, the EECA is capable of finding other learners that may play the role of a tutor after recognizing and processing his affective

state via the emotional agents. In the case that the EECA has not found the appropriate learner, he can address this request to the tutor for giving any explanations and/or remarks.

2 Results

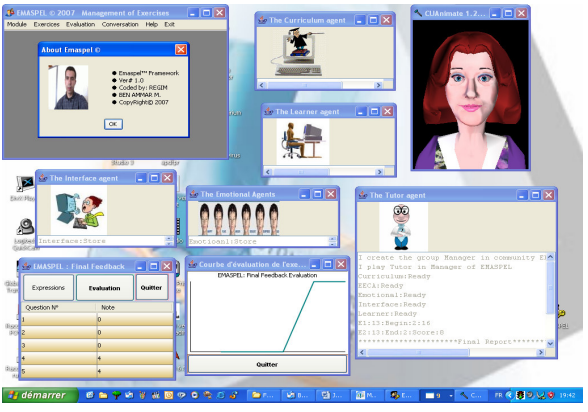


Fig. 1. EMASPEL Framework

References

1. Neji, M., Ben Ammar, M.: Agent-based Collaborative Affective e-Learning Framework. The Electronic Journal of e-Learning 5(2), 123–134 (2007)
2. Ben Ammar, M., Neji, M., Gouardères, G.: Conversational Embodied Peer agents in affective e-learning. In: Rebolledo-Mendez, G., Martinez-Miron, E. (eds.) Workshop on Motivational and Affective Issues in ITS. 8th International Conference on ITS 2006, pp. 29–37 (2006)
3. Nkambou, R.: Towards Affective Intelligent Tutoring System, Workshop on Motivational and Affective Issues in ITS. In: 8th International Conference on ITS 2006, pp. 5–12 (2006)
4. Ekman, P., Friesen, W.V.: Unmasking the face: a guide to recognizing emotions facial cues. Prentice-Hall (1975)

Measuring the Perceived Difficulty of a Lecture Using Automatic Facial Expression Recognition

Jacob Whitehill, Marian Bartlett, and Javier Movellan

Machine Perception Laboratory
University of California, San Diego
{jake,movellan}@mplab.ucsd.edu, marni@salk.edu

Abstract. This project explores the idea of facial expression for automated feedback in teaching. We show how automatic real-time facial expression recognition can be effectively used to estimate the difficulty level, as perceived by an individual student, of a delivered lecture. We also show that facial expression is predictive of an individual student's preferred rate of curriculum presentation at each moment in time. On a video lecture viewing task, training on less than two minutes of recorded facial expression data and testing on a separate validation set, our system predicted the subjects' self-reported difficulty scores with mean accuracy of 0.42 (Pearson r) and their preferred viewing speeds with mean accuracy of 0.29. Our techniques are fully automatic and have potential applications for both intelligent tutoring systems (ITS) and standard classroom environments. [1]

1 Introduction

This paper explores the utility of *facial expression* as a feedback signal from student to teacher. Previous work [2,3] has investigated using facial expression for predicting the student's affective state. This paper investigates the usefulness of automatic expression recognition for two tasks: (1) measuring the student's perception of difficulty of a delivered lecture at each moment in time, and (2) determining a student's preferred viewing speed of lesson material.

2 Experiment

We conducted a pilot experiment in which subjects viewed a video lecture at an adjustable speed while their facial expressions were recognized automatically and recorded. Using the "difficulty" scores that the subjects report, the correlations between facial expression and difficulty, and between expression and preferred viewing speed, were assessed. Each of 8 human subjects watched a recorded video lecture 200 seconds (3 min 20 sec) in length. The video consisted of 7 short video clips concatenated together about a disparate range of topics, including physics, philosophy, math, and teenage gossip. The experiment proceeded as follows:

[1] The full version of this paper is available at [1].

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 668–670, 2008.

1. **Watch the video lecture.** The playback speed could be adjusted by the subject using the keyboard. Facial expression data of each human subject were recorded while they watched the video.
2. **Take the quiz.** The quiz consisted of 6 specific questions about the lecture.
3. **Self-report on the difficulty.** The lecture was re-played at a speed of 1.0. Subjects adjusted their rating of the video's difficulty on a scale of 1 to 10 on a frame-by-frame basis using arrow keys.

Facial expressions were analyzed and recorded automatically using the CERT system [4]. CERT outputs estimated intensities of facial "action units" (AUs) of face videos in real time. AUs are the component muscle movements that comprise facial expressions, analagous to phonemes for words, and are defined in the Facial Action Coding System [5], which is the standard framework for objectively coding human facial expression. CERT recognizes AUs 1, 2, 4, 5, 9, 10, 12, 14, 15, 17, 20, and 45 as well as Smile.

None of the subjects was aware of the purpose of the experiment or that expression data were measured. Subjects were encouraged to view the video efficiently (by adjusting its speed) by a conspicuous automatic "shut-off" timer and a quiz. In actuality, the timer had no effect, and the quiz was not graded.

3 Analysis

Our experiment yielded three data series which we time-aligned:

1. **Difficulty:** The subjects' self-reported scores of how difficult they perceived the lecture at each moment in time.
2. **Speed**: The lecture speed at each moment as controlled by the user.
3. **Expression:** A set of real-valued expression channels output by CERT which estimate the intensity of 13 expression channels (12 AUs + Smile). The channels were smoothed using local quadratic regression.

We employed linear regression over the Expression channels to predict both Difficulty and Speed (see Figure 1 for an example). Separate regression models were trained for each subject. The average correlation (over all 8 subjects) of the Difficulty scores predicted by the regression model (training over all data) with the self-reported Difficulty scores was 0.75, suggesting that facial expression contains a large amount of useful signal.

Subjects exhibited a wide variability of which facial muscle movements were correlated with Difficulty and Speed. The facial movement most consistently correlated was AU 45 ("blink"). For six out of eight subjects, the correlation was negative (subjects blinked less during more difficult lecture segments), which is consistent with evidence from experimental psychology that blink rate decreases when interest or mental load is high [6].

Generalization Performance: To assess the effectiveness of using facial expression to predict Difficulty and Speed values in a real intelligent tutoring system, we split the data series into disjoint "bands" ($\sim$ 12 sec long) of training

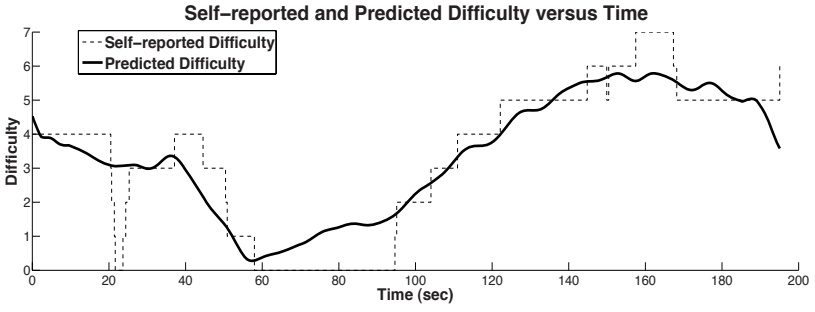


Fig. 1. Comparison of the self-reported Difficulty scores with the predicted Difficulty scores using linear regression over the facial expression channels for Subject #6. For this figure, all data were used for training the regression model.

and validation sets (see [1] for details). Linear regression models over the Expression data series were trained to predict Difficulty and Speed for each human subject. Generalization performance was assessed by correlating predicted to self-reported Difficulty and Speed values over the validation data.

The average correlation across all subjects of predicted to self-reported Difficulty was 0.42; the average correlation for Speed was 0.29. Individual correlations were statistically significant for all but one subject. While these results show room for improvement, they already demonstrate that automatic facial expression recognition can extract relevant information for ITS.

4 Conclusions

Our results show that automatic facial expression recognition is already a useful feedback signal for intelligent tutoring systems for two concrete tasks: perceived difficulty estimation, and preferred speed prediction. As expression recognition technology improves, its usefulness in ITS will continue to grow.

References

1. Whitehill, J., Bartlett, M., Movellan, J.: Measuring the perceived difficulty of a lecture using automatic facial expression recognition. Technical Report 2008.01, Machine Perception Lab (2008), http://mplab.ucsd.edu/~jake/its08.pdf
2. Rio, H., Soli, A.L., Aguirr, E., Guerrer, L., Santa, J.P.A.: Facial expression recognition and modeling for virtual intelligent tutoring systems. In: Proceedings of the Mexican International Conference on Artificial Intelligence (2000)
3. D'Mello, S., Picard, R., Graesser, A.: Towards an affect-sensitive autotutor. IEEE Intelligent Systems, Special issue on Intelligent Educational Systems 22(4) (2007)
4. Bartlett, M., Littlewort, G., Frank, M., Lainscsek, C., Fasel, I., Movellan, J.: Auto. recog. of facial actions in spontaneous expressions. Jour. of Multimedia (2006)
5. Ekman, P., Friesen, W.: The Facial Action Coding System: A Technique For The Measurement of Facial Movement. Consulting Psychologists Press (1978)
6. Holland, M., Tarlow, G.: Blinking and mental load. Psych. Reports 31(1) (1972)

Minimal Feedback During Tutorial Dialogue*

Pamela Jordan and Diane Litman

Learning Research and Development Center
University of Pittsburgh, Pittsburgh PA 15260

Abstract. We analyzed unexpected student responses to a natural language (NL) ITS to determine if improved feedback could be beneficial. Our analysis of a corpus of ITS-student dialogues suggests that unexpected responses represent learning opportunities. We outline our plans for testing feedback appropriate to subclassifications of these responses.

1 Introduction

Some patterns found in human-human and human-computer interactions predict similar outcomes while some do not [1,2,3]. So it is valuable to look at both types of interactions when building ITS. One study of human-human interactions suggests that increased tutor feedback predicts a bad learning outcome and that student errors do not account for these negative correlations [1]. Our research focuses on student responses that an ITS categorizes as unrecognizable (or *default*). In such cases ITS typically provide only minimal feedback that indicates whether a student action is correct or not [4]. The variety of ways in which ITS handle the default category (e.g. always treat it as either correct or incorrect) [4] suggests that choosing one fixed way of handling these responses may not suffice in all situations.

Default student responses range from those that are not fully correct to those that are simply failures to respond [4]. We are exploring whether some default responses arise because the student does not recognize what the tutor wants (i.e. tutor's communicative intentions) since this could lead students to respond in a way that ITS builders did not anticipate. Generally during human-human dialogue a hearer is expected to show to what degree he has understood a speaker [5] so that repairs are better informed. If an ITS provided this type of feedback, the human dialogue partner would have more information with which to make a repair and could improve the efficiency of the interaction. For example, if the ITS says "You're close but you need to be more careful. Try it again." then the student knows he should focus on adding details. Otherwise the student doesn't know what sort of repair to attempt.

2 The NL ITS Testbed and Corpus

The corpus we are using to analyze default student responses comprises 448 typed dialogues between 64 students and a NL ITS that tutors seven qualitative physics

* This work was funded by NSF grant 0325054.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 671–673, 2008.

Table 1. Corpus excerpts

	Example of a *no attempt* response
system:	Which variable quantity does the '24.0 N' represent?
student:	I don't know. *(choices: applied force, attempted,* **no attempt***)*
system:	The '24.0 N' represents the man's applied force on the crate.

	Example of an attempted but wrong response
system:	Which variable quantity does the '24.0 N' represent?
student:	Force of friction *(choices: applied force,* **attempted***, no attempt)*
system:	No. The '24.0 N' represents the man's applied force on the crate.

	Example of a vague response
system:	What quantities influence the kinetic energy of the truck?
student:	The velocity of travel and the mass of the truck
	(choices: mass and magnitude of velocity, **attempted***, no attempt)*
system:	I disagree. From $KE = 1/2 * m * v^2$, we can infer that kinetic
	energy is influenced by the truck's magnitude of velocity and its mass

problems in the work-energy domain [6]. Domain experts identified 30 knowledge components (KCs) that were needed to solve the problems and developed a 33 item test for pre-/post-testing. Every item tests one or more KCs. Learning gains were significant for a majority of the KCs, as were composite learning gains (e.g. the lowest composite gain was for KCs about net work $F(3,61)=3.2$, $p<.03$).

The NL ITS guided students through problem solving and asked for justifications for key KCs. Because the corpus was collected for the purpose of deriving dialogue strategies on when to elicit responses and when to request justifications, its fully automated NL understanding module was replaced with a human interpreter, called the *wizard*, to reduce the confounds of misrecognizing student explanations. The wizard's interface showed the dialogue history, the student's last response and a list of choices for classifying the student's response. Dialogue excerpts from the corpus are illustrated in Table 1, where the choices available are shown after the student response and the wizard's selection is in bold.

3 An Analysis of Students' Default Responses

Only two *default* subclasses were available for wizards to select; *attempted* and *no attempt*. For *no attempt* responses such as "I don't know." the system gave no minimal feedback before it followed up and for *attempted* responses, it gave negative feedback since the response did not answer the intended question. We found that 21% of all NL responses from students were classified as *attempted* and 12% as *no attempt*. But only 2% of responses turned out to be *no attempt* because *attempted* responses were sometimes misclassified by wizards as *no attempt*. Possibly this was done to circumvent students receiving negative feedback. But even with these misclassifications, there were still significant correlations between classifications of responses and learning; there was a significant moderate positive correlation between post-test scores (after removing the effects of pre-test scores) and the

percentage of students' non-default responses (R=.47,p=0) and there were significant weak negative correlations between post-test scores and responses that were classified as *no attempt* (R=-.30,p=.017) or *attempted* (R=-.32,p=.011). The non-default responses are step specific and are either correct or non-correct ones that warrant a specific follow-up. As in other studies substantive responses from students were predictive of learning regardless of correctness. Negative correlations with learning suggest that default responses are possible learning opportunities and thus warrant further analysis.

4 Discussion and On-Going Work

We identified the following alternative subclasses for *default* responses in the corpus; *no attempt*, *wrong*, *vague* and *overly specific*. Intuition suggests that *vague* and *overly specific* responses indicate the student is close to having learned a KC while the remaining *default* responses indicate the opposite. These subclasses could reflect the student's progress on a KC. When the student receives a pointer on what kind of error to look for and fix, it is reasonable to expect that if he attempts a repair he will move closer to a fully correct response. Furthermore, the student may be more motivated to pay attention to any specific feedback that follows a retry. Thus students may be able to achieve *correct* contributions sooner during their tutoring when they receive generic feedback that indicates the type of error and have a chance to retry. To test this hypothesis, we will conduct a controlled experiment with two conditions; the treatment condition will receive appropriate minimal feedback for each subclass of a *default* response and the control condition will receive no minimal feedback for any *default* response. We will compare the learning gains, learning curves and correctness during dialogues for the two conditions.

References

1. Chi, M., Roy, M., Hausmann, R.: Learning from observing tutoring collaboratively: Insights about tutoring effectiveness from vicarious learning. Cognitive Science (in press)
2. Jackson, G., Person, N., Graesser, A.: Adaptive tutorial dialogue in autotutor. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, Springer, Heidelberg (2004)
3. Litman, D., Forbes-Riley, K.: Correlations between dialogue acts and learning in spoken tutoring dialogues. Natural Language Engineering (2006)
4. VanLehn, K.: The behavior of tutoring systems. International Journal of Artificial Intelligence and Education 16 (2006)
5. Clark, H.H.: Using Language. Cambridge University Press (1996)
6. VanLehn, K., Jordan, P., Litman, D.: Developing pedagogically effective tutorial dialogue tactics: Experiments and a testbed. In: Proceedings of SLaTE Workshop on Speech and Language Technology in Education (2007)

Can Students Edit Their Learner Model Appropriately?

Susan Bull[1], Xiaoxi Dong[2], Mark Britland[1], and Yu Guo[3]

[1] University of Birmingham, U.K.
[2] McGill University, Canada
[3] Worcester Polytechnic Institute, U.S.A.
s.bull@bham.ac.uk

Abstract. We investigate whether students can edit their learner model appropriately considering: (i) evidence about the model contents; (ii) accuracy of self-assessment; (iii) type of information in the model; (iv) level of knowledge.

1 Introduction

Some systems open the learner model to the user, and some also allow users to provide data directly to their model (e.g. [1],[2]), giving learners greater responsibility and control over their learning (see [1]). However, little work has investigated users' ability to edit their model accurately. We address some of the initial questions using three simple learner modelling systems: DataEvidence (data structures), BusinessAssess (accounting & finance), NLPInfo (natural language processing).

1. Does evidence for model contents affect ability to edit the model appropriately?
2. Does accuracy of self-assessment affect ability to edit the model appropriately?
3. Does type of information displayed affect ability to edit the model appropriately?
4. Does level of knowledge affect ability to edit the model appropriately?

2 Do Users Accurately Edit Their Learner Model?

Users may not notice if adaptation is incorrect based on preferences [3]. If users also do not notice errors in their learner model, they will not be able to correct them. To explore whether users notice and correct errors, errors were automatically inserted in the models (e.g. knowledge→misconception), with a balance of positive/negative changes (4-6 errors for most users). We investigate how learners edited their models.

Multiple-choice questions are used to elicit learner beliefs. The learner models are presented textually, describing concepts/misconceptions, e.g. in DataEvidence: 'You may believe the queue adds items at one end and removes items at the other' for a belief modelled with uncertainty (uncertainty shown by 'you *may* believe'). Model editing is by selection from a belief list - see Fig. 1 (BusinessAssess). NLPInfo has two model presentations: skill meters showing knowledge level (Fig. 1); and the concept model (as above, but without misconceptions). Editing skill meters has to allow students to change knowledge level rather than select a belief. Radio buttons give 3 levels of knowledge: good, moderate, unsatisfactory. For consistency, to edit a concept students select whether they believe, may believe, or do not believe it. Users of each system were volunteers taking relevant degrees. Sessions lasted 30 minutes. Logs, learner models and questionnaires were analysed.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 674–676, 2008.

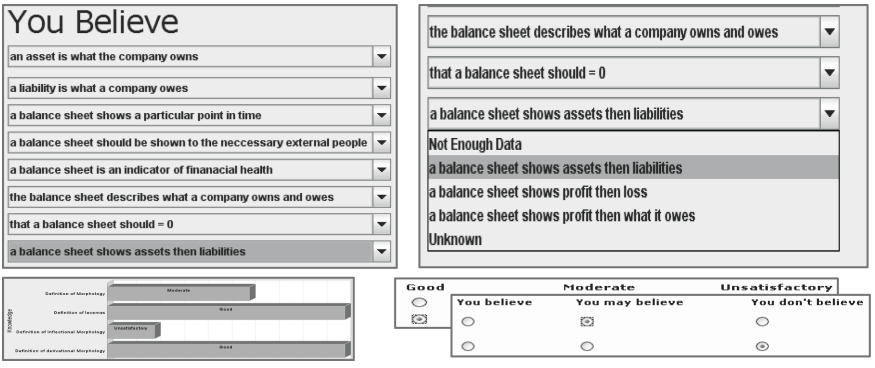


Fig. 1. Editing the learner model

Table 1. Editing the learner model

DataEvidence (n=24)	*Errors Noticed*	*Correct Edits*		*Incorrect Edits*		*Evidence Useful*
		Concept	*Miscon*	*Concept*	*Miscon*	
(i) no evidence (n=8)	6 users	5 users	1 user	2 users	3 users	-
(ii) evidence (n=8)	6 users	6 users	0 users	2 users	1 user	7 users
(iii) ev. optional (n=8)	5 users	5 users	1 user	0 users	3 users	3 users

BusinessAssess (n=16)	*Errors Noticed*	*Correct Edits*		*Incorrect Edits*		*Would Edit*
		Concept	*Miscon*	*Concept*	*Miscon*	
(i) accurate (n=8)	7 users	0 users	7 users	0 users	0 users	7 users
(ii) overestimate (n=5)	5 users	0 users	3 users	0 users	1 user	5 users
(iii) underestim. (n=3)	3 users	0 users	2 users	0 users	0 users	2 users

NLPInfo (n=20)	*Errors Noticed*	*Correct Edits*	*Near Correct Edits*	*Incorrect Edits*	*Confident to Edit*
(i) concept (n=20)	18 users	8 users	10 users	2 users	13 users
(ii) skill m (n=20)	14 users	5 users	7 users	5 users	3 users

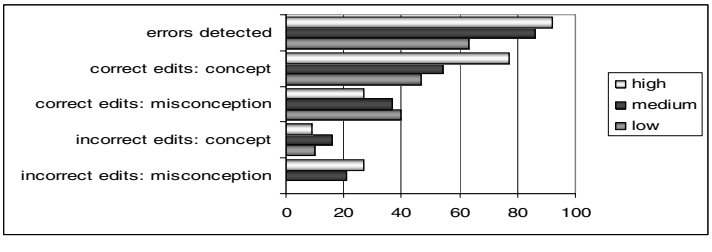


Fig. 2. Editing the learner model according to knowledge level - percent (n=60)

1. DataEvidence explains its inferences, e.g. 'You chose to output the last inserted element first in questions about the output sequence to a certain input'. There were 24 users in 3 groups of 8: no access to evidence; requirement to read evidence; optional access to evidence (3 of the 8 read the evidence). Results are given in Table 1.

2. Viewing the learner model can have positive effects on self-assessment [4]. In contrast, BusinessAssess investigates whether students who assess their knowledge accurately also edit their model accurately. Of 16 users, 3 had high knowledge; 7 medium; 6 low. Although often suggested that competent learners may more likely underestimate, and less able learners overestimate themselves [5], no relationship was found. Table 1 gives results for model editing in relation to self-assessment accuracy.

3. NLPInfo shows skill meters and beliefs on 3 levels. Errors were inserted on the extremes (e.g. good→unsatisfactory; believe→do not believe). 20 users took part: half viewing belief statements, and half skill meters first. There were no differences relating to sequence. ('Near correct edits' are *towards* correct edits.) See Table 1.

4. To consider knowledge level we use all data: 60 users. 13 had a high knowledge level; 28 medium; 19 low. NLPInfo does not model misconceptions, so the maximum number who could have edited misconceptions is 40. Of these, 11 had high knowledge; 19 medium; 10 low. As 'near correct edits' indicate an edit that moves closer to a correct representation, we include them in 'correct edits'. Results are in Table 1.

3 Conclusions

Learner involvement in learner modelling can assist the modelling process while giving learners greater responsibility for decisions in their learning. Not all errors inserted into the models were identified, but most users noticed at least one inserted error, and most errors detected were edited appropriately; with fewer new errors introduced. There was no clear link between some of the features investigated and the accuracy of model editing (provision of evidence, self-assessment skills). But: weaker learners tended to have more difficulty detecting errors and editing inserted known concepts (but not misconceptions); and editing was more successful when the model showed belief statements rather than knowledge levels. We hypothesise also that the nature of the domain may influence model editing, as there were differences between editing concepts and misconceptions in DataEvidence and BusinessAssess. Given that some of the inserted errors were detected and correctly edited, we recommend further research into the potential for editable learner models to increase model accuracy and give learners more responsibility for their learning.

References

1. Kay, J.: Learner Know Thyself: Student Models to Give Learner Control and Responsibility. In: International Conference on Computers in Education, AACE, pp. 17–24 (1997)
2. Mabbott, A., Bull, S.: Student Preferences for Editing, Persuading and Negotiating the Open Learner Model. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 481–490. Springer, Heidelberg (2006)
3. Czarkowski, M., Kay, J., Potts, S.: Web Framework for Scrutable Adaptation. In: Learner Modelling for Reflection Workshop, Artificial Intelligence in Education 2005, pp. 11–18 (2005)
4. Mitrovic, A., Martin, B.: Evaluating the Effect of Open Student Models on Self-Assessment. International Journal of Artificial Intelligence in Education 17(2), 121–144 (2007)
5. Boud, D., Falchikov, N.: Quantitative Studies of Student Self-Assessment in Higher Education: A Critical Analysis of Findings. Higher Education 18, 529–549 (1989)

When Is Assistance Helpful to Learning? Results in Combining Worked Examples and Intelligent Tutoring

Bruce M. McLaren, Sung-Joo Lim, and Kenneth R. Koedinger

Human-Computer Interaction Institute, 5000 Forbes Avenue, Carnegie Mellon University, Pittsburgh, PA, 15213-3891 United States
bmclaren@cs.cmu.edu, sungjol@andrew.cmu.edu, koedinger@cs.cmu.edu

Abstract. When should instruction provide or withhold assistance? In three empirical studies, we have investigated whether worked examples, a high-assistance approach, studied in conjunction with tutored problems to be solved, a mid-level assistance approach, can lead to better learning. Contrary to prior results with *untutored* problem solving, a low-assistance approach, we found that worked examples alternating with isomorphic tutored problems did not produce more learning gains than tutored problems alone. However, the examples group across the three studies learned more efficiently than the tutored-alone group. Our studies, in conjunction with past studies, suggest that mid-level assistance leads to better learning than either lower or higher level assistance. However, while our results are illuminating, more work is needed to develop predictive theory for what combinations of assistance yield the most effective and efficient learning.

1 Introduction

The *Assistance Dilemma* [1] characterizes a long-standing unsolved problem in the learning sciences: when should instruction provide students with assistance and when should it be withheld? Some researchers have argued for providing maximal assistance [e.g., 2], while others, argue for minimal assistance [e.g., 3]. In three studies in the domain of chemistry we have explored the assistance dilemma [4, 5].

In this paper we discuss our results in experimenting with an intelligent tutor supplemented with worked examples, a combination that has only recently been investigated. The worked example principle, as stated in [6], is: "Replace some practice problems with worked examples", i.e., provide students with an alternating combination of worked examples and problems. The theory behind the principle is that human working memory, which has a limited capacity, is taxed by strictly solving problems, which requires thinking, such as the setting of subgoals. The rationale, then, is that worked examples free mental resources for learning processes.

But then why *mix* worked examples and problem solving, as suggested by the worked example principle? The theory seems to suggest that worked examples alone, a high-assistance approach, would be best for learning. Past empirical results have been mixed on this issue. For instance, Trafton and Reiser compared problem solving

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 677–680, 2008.

with no tutoring to interleaved worked examples and problem solving with no tutoring and found statistically significant learning gains and learning efficiency for the alternating condition [7]. Lovett performed a study that showed that *both* a low and high assistant approach could be beneficial [8], while Kalyuga *et al*'s results suggest that assistance should decline over time, as subjects gain expertise [9]. These (and other) mixed results indicate that there is room for continued studies to explore the issue of when assistance is appropriate in instruction.

Furthermore, until recently there had been little study of the comparative learning benefits of *intelligent tutored* problem solving and other forms of assistance. Tutored problem solving is a mid-level assistance approach that provides more assistance than untutored problem solving but somewhat less than worked examples. Only Schwonke *et al* [10], besides ourselves, have explored the combination of tutored problems and worked examples (without one approach being used by and subservient to the other). Our hypothesis is that *the interleaving of worked examples with tutored problems will further improve learning beyond the benefits of the tutor itself.* We explored this hypothesis using the Stoichiometry Tutor, a web-based intelligent tutor that provides support for a basic subarea of high school chemistry [4, 5].

2 Study Design and Results

In all three studies a 2x2 between-subjects design was employed. The independent variable of primary interest to this paper is *Worked Examples*, with one level being *Tutored Alone* and the other *Worked Examples + Tutored*. In the former condition, which will be referred to as the "Problems Condition", subjects solely solved problems with the Stoichiometry Tutor. In the latter condition, which will be referred to as the "Examples Condition", subjects alternated between studying a worked example (followed by self-explanation questions) and solving an isomorphic problem with the aid of the tutor. (The second independent variable of the studies, "personalization," will not be discussed in this brief paper, as it is not our focus here.)

All instructional materials were provided via the Internet. All subjects worked on 10 study problems (15 in Study 1), presented according to the two conditions described above. All subjects were also given pre- and post-questionnaires, and pre and posttests, with all pre and posttest problems involving the same type of problems as the study problems. Instructional videos on stoichiometry were interspersed, as appropriate, with the study problems in both conditions. The N values of the 3 studies were, respectively, 63, 60, and 81, for a total of 204 participants across all three studies.

Table 1. Average learning efficiency, calculated, per subject, as z-score (learning gain) - z-score (instructional time) with z-score = (value – average) / stddev. The P-value was calculated using a one-way ANOVA between the Examples and Problems Conditions' learning efficiency.

	Examples Condition Learning Efficiency	**Problems Condition Learning Efficiency**	**P-value**	**Effect Size (Cohen's *d*)**
Study 1	0.47	-0.45	**0.005***	0.75
Study 2	0.24	-0.26	**0.146**	0.39
Study 3	0.40	-0.41	**0.015***	0.56

* - Significant result.

The results were as follows. The students exhibited significant learning between the pre and posttests in all conditions of all studies. On the other hand, the students in the Examples Conditions did not learn more than those in the Problems Conditions, contrary to previous findings such as [7]. However, subjects in all of the Examples Conditions spent significantly less time with the study problems. That is, the subjects in the Examples Condition, while they did not learn *more*, they learned *more efficiently* than those in the Problems Condition, as shown in Table 1.

3 Conclusion

The students in the Examples Condition used 21% less time to complete the same problems. If these results were to scale across a 20-week course, students could save 4 weeks of time – yet learn just as much. What explains these findings? Students in the Examples Conditions worked significantly faster on the *first* of the isomorphic example-problem pairs (i.e., the example) than the second (i.e., the problem). The extra time the students in the Problems Condition took on the first problems – which often seemed to be used to turn problems into examples by clicking to bottom-out hints – did not benefit them. This may be because clicking through hints is a less efficient way to see an example compared to seeing that example immediately.

Unlike most prior studies, ours involved *tutored* instead of *untutored* problem solving. Like [10], our results are consistent with the hypothesis that mid-level assistance provides the greatest learning advantages. We plan to test this more explicitly in a new study, which will compare three levels of assistance: all tutored problems (lower assistance), alternating examples and tutored problems (mid-level), and all worked examples (higher assistance). Of course, to more generally test the hypothesis it is important to do similar studies in other domains with different tutors.

Acknowledgements. The PSLC, NSF Grant #0354420, supported this research.

References

1. Koedinger, K.R., Aleven, V.: Exploring the Assistance Dilemma in Experiments with Cognitive Tutors. Educational Psychology Review 19(3), 239–264 (2007)
2. Kirschner, P.A., Sweller, J., Clark, R.E.: Why Minimal Guidance During Instruction does not Work: An Analysis of the Failure of Constructivist, Discovery, Problem-Based, Experiential, and Inquiry-Based Teaching. Educational Psychologist 41(2), 75–86 (2006)
3. Steffe, L., Gale., J. (eds.): Constructivism in Education. Lawrence Erlbaum Assoc. (1995)
4. McLaren, B.M., Lim, S., Gagnon, F., Yaron, D., Koedinger, K.R.: Studying the Effects of Personalized Language and Worked Examples in the Context of a Web-Based Intelligent Tutor. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 318–328. Springer, Heidelberg (2006)
5. McLaren, B.M., Lim, S., Yaron, D., Koedinger, K.R.: Can a Polite Intelligent Tutoring System Lead to Improved Learning Outside of the Lab? In: 13th Int'l Conf. on Artificial Intelligence in Education (AIED 2007), pp. 433–440 (2007)
6. Clark, R.C., Mayer, R.E.: e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning. Jossey-Bass/Pfeiffer (2003)

7. Trafton, J.G., Reiser, B.J.: The Contributions of Studying Examples and Solving Problems to Skill Acquisition. In: 15th Annual Conf. of the Cog. Science Society, pp. 1017–1022 (1993)
8. Lovett, M.C.: Learning by Problem Solving Versus by Examples: The Benefits of Generating and Receiving Information. In: 14th Annual Conference of the Cognitive Science Society, pp. 956–961. Erlbaum, Hillsdale (1992)
9. Kalyuga, S., Chandler, P., Tuovinen, J., Sweller, J.: When Problem Solving is Superior to Studying Worked Examples. Journal of Educational Psychology 93, 579–588 (2001)
10. Schwonke, R., Wittwer, J., Aleven, V., Salden, R.J.C.M., Krieg, C., Renkl, A.: Can Tutored Problem Solving Benefit from Faded Worked-Out Examples? In: 2nd European Cognitive Science Conference, pp. 59–64 (2007)

Enabling Reputation-Based Trust in Privacy-Enhanced Learning Systems

Mohd Anwar and Jim Greer

ARIES Lab, Department of Computer Science
University of Saskatchewan, Saskatoon, Canada
{mohd.anwar,jim.greer}@usask.ca

Abstract. Privacy and trust are the cornerstones of safe and engaging learning environments. Trust is associated with the reputation of an actor. Pseudonymous identities can accrue trust while preserving the actor's privacy. But activities such as aggregating reputations from an actor's multiple pseudo-identities or transferring reputation to a new pseudonym pose problems with privacy preservation. In this vein, we propose privacy-preserving identity management systems for learners that support reputation aggregation and transfer across multiple pseudonyms.

1 Introduction

We [1], along with many others [3], view that identity management is an effective solution to privacy in the learning domains. In an identity management scheme, each user participates in a context by assuming a context-specific partial identity and potentially many identifiers or pseudonyms. However, since the pseudo-identities and pseudonyms offered by the identity management solutions are not linkable, reputation earned over a pseudonym is untransferable with the cancellation or switching of that pseudonym.

In this paper, we present an implementation of a secure reputation transfer protocol [2] to allow reputation transfer among multiple pseudo-identities (e.g. pseudonyms) without letting anyone draw associations among these pseudo-identities. We have built a prototypical system that manages reputation for three different sorts of roles present in an e-learning community: helper, peer, and lurker. We have also done a preliminary evaluation to show that our system helps facilitate trust while effectively preserving privacy.

2 Reputation Evaluation and Transfer

Conceptually, the reputation transfer protocol proposed earlier [2] is able to securely transfer reputation across pseudonyms ensuring no leakage of identities of participants through the process of transferring reputation. We have employed a reputation generating technique and implemented the reputation transfer protocol in a prototypical system to manage reputation for learners. Additionally,

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 681–683, 2008.

our system makes a reputation transfer unobservable and restricts foul play for taking undue advantage of reputation transfer.

To provide a solid and parsimonious foundation for the empirical study of trust for another party, Mayer et al. [4] observe three characteristics of a trustee appearing often in the literature: ability, benevolence, and integrity. To evaluate reputation of learners, we map each of these characteristic to a feature, specific to roles a learner plays in various learning contexts. For example, a learner in a helper role is evaluated by three features: insightfulness, helpfulness, and availability representing ability, benevolence, and integrity respectively. In our system, reputation is measured by averaging a set of feedbacks (ratings on a 0 to 5 scale) by other actors on role specific features. In case of merge/transfer, one actor's aggregate rating is incremented one-by-one by for each rating transaction of the other actor and vice versa. There is a random time delay between each of the increments to make reputation transfer indistinguishable from reputation update by a new rating, making reputation transfer unobservable.

3 Implementation

Our prototypical system is implemented through a client (for actors) and a multi-threaded server (for guarantor) suite written in Java language. The system manages reputation for 3 different roles that are present in an e-learning community: helper, peer, and lurker. Our system allows a user to perform any of the following 4 tasks: "register (i.e., register a pseudonym with a guarantor)", "evaluate (i.e., rate an actor)", "transfer"(e.g. transfer/merge reputation across pseudonyms), and "query"(e.g. query reputation of a pseudonymous actor).

As a pseudonym, say TomTheHelper requests a reputation transfer, it presents the registration number and the reputation digest (originally provided to it by the guarantor) to the guarantor, so the guarantor could authenticate its identity and retrieve its reputation. The guarantor then awaits consent (of reputation transfer) from another pseudonym, Say JerryTheSage in the form of presenting its (i.e. JerryTheSage's) registration number and reputation digest. The guarantor transfers/merges TomTheHelper's reputation to JerryTheSage only when they appear to be a registered user making a simultaneous request. Empirical tests successfully show that the transferring aspect of our system works.

4 Evaluation and Results

Our study was designed to see whether our system facilitates reputation based trust while preserving privacy by making secure reputation transfer/merge across multiple pseudonyms. For the above purpose, we have initialized our system to generate multiple instances of the four types of events (reputation evaluation request, reputation transfer request, reputation merge request, and nothing happened) in some random order for four pseudonyms representing two actors. At multiple time steps during the simulation, we queried the guarantor for the latest reputation of each of the registered pseudonyms and logged them accordingly.

These logs were then provided to a security attack-defense expert to attempt to deduce what events might have occurred based on an analysis of the reputation score patterns over various time steps. We also asked our expert to see whether he could distinguish among reputation transfer, reputation merge, and normal updates of ratings.

In generating reputation logs, the simulation performed 3 transfers and 7 merges of reputations across four pseudonyms of two actors. Although the data set was relatively small, the expert could not make any definitive conclusions that would identify which pseudonyms corresponded to the same actor. Our expert suspected that four mergers or transfers of reputation occurred. The one merger hypothesis in which he was most confident was totally incorrect. Two of our expert's suspected mergers or transfers actually did correspond to real mergers or transfers, but he entirely missed eight of the events. Our expert correctly had a suspicion that one transfer and one merger (of the ten) had occurred, but he could not be sure. Out of these 2 correct hypotheses, the expert could not confirm conclusively about any of the mergers or transfers. We could say that these correct guesses are no more than random luck. With an increase in the number of actors, it becomes even harder to guess about any reputation transfer or merge. Therefore, we could say that our system supports reputation transfer with privacy preservation.

5 Conclusion

A learner uses trust as a scale to find a suitable recommender, peer, helper, and mentor. On the other hand, a naively constructed privacy-enhanced learning environment offers isolated personal learning spaces, which allow learners to be sometimes frustrated, overwhelmed, or dissatisfied with learning objects or instructors. In this paper, an approach to address privacy protection and trust facilitation was explored. A mechanism to evaluate and attach reputation to a pseudonymous identity can help measure trust without the loss of privacy.

References

1. Anwar, M., Greer, J., Brooks, C.: Privacy Enhanced Personlization in E-learning. In: Proceedings of the 2006 International Conference on Privacy, Security, and Trust. Markham, Ontario, Canada (2006)
2. Anwar, M., Greer, J.: Reputation Management in Privacy-Enhanced E-learning. In: The proceedings of the 3rd Annual Scientific Conference of the LORNET Research Network (I2LOR 2006), Montreal, Canada (2006)
3. Borcea, K., Donker, H., Franz, E., Pfitzmann, A., Wahrig, H.: Towards Privacy-Aware Elearning. In: Danezis, G., Martin, D. (eds.) PET 2005. LNCS, vol. 3856, pp. 167–178. Springer, Heidelberg (2006)
4. Mayer, R.C., Davis, J.H., Schoorman, F.D.: An integrative model of organizational trust. Academy of Management Review 20, 709–734 (1995)

Authoring Educational Games with Greenmind

Brent Martin

Department of Computer Science and Software Engineering
University of Canterbury, Private Bag 4800 Ilam, Christchurch NZ
brent.martin@canterbury.ac.nz

Abstract. We present Greenmind, an authoring tool for developing intelligent educational games. Greenmind separates game development from ITS delivery, allowing specialist game developers or teachers to create their own game front-end clients for the WETAS ITS server, and making it possible for a game interface to be added to existing tutoring systems.

1 Introduction

Intelligent Tutoring Systems are emerging from the lab and beginning to expand the horizons of education from traditional students and learning institutions to anyone with an Internet connection. Acting as an enhancement to traditional distance learning offerings, they promise to augment laboratories and tutorials by allowing students to practice the skills they are learning from home. In recent years tutors such as the Geometry and Algebra tutors [1], and the Addison-Wesley database place suite (SQL-Tutor, ER-Tutor and NORMIT) [2] have made it out of the lab and into the classroom.

Another emerging trend in education is the use of computer games as a teaching medium. For example, at the University of Canterbury courses are now being offered on the development of educational games, and topics such as Machine Learning and Artificial Intelligence are being taught using games, because this is an approach most students are familiar with and can relate to. The ITS community has also become interested, with several education game systems recently appearing, including Tactical Iraqi [3], Language Builder [4], ExpertCop [5] and My-Pet-Our-Pet [6]. Such systems typically report gains attributed to the game environment, and thus provide a strong motivation for building games into our ITS.

We are interested in whether adding a game front-end increases the effectiveness of ITS in general. Unfortunately educational games are not easy to build. In particular, it is not readily apparent how a game interface could be easily built for an existing ITS to measure the difference in student learning, an essential step to verify the efficacy of the approach. To overcome this we have developed Greenmind, an educational game authoring system that allows ITS developers to add educational game interfaces to existing and new ITS developed using the WETAS [4] ITS authoring shell.

2 Greenmind

Greenmind is an educational game-authoring tool that works with Greenfoot [7], a JAVA IDE for 2D graphical programming that is particularly suited to game development, by

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 684–686, 2008.

extending its class library to include ITS functionality. Game authors develop their game scenario as they normally would in Greenfoot, however they also implement additional methods that provide the educational content. Once one or more game scenarios have been created they can be bundled into a Greenmind game and exported as a standalone Java application. Greenmind uses the WETAS intelligent tutoring shell via an RPC interface to provide the ITS functionality.

Central to a Greenmind scenario is the "Tutor" class, which is an extension of Greenfoot's "Actor". This class contains low-level methods for interacting with the tutoring backend (WETAS). These methods expose an API very similar to that provided by the WETAS RPC interface, including functions such as get problem, get feedback, set the sub domain, load/save the current solution and get help. The game developer extends this class to create the individual characters in the game. The author uses the tutoring methods to add ITS content by calling them as part of the "act()" method for the Tutor class. This method determines the character's overall behavior. The typical situation is as follows: when the player's actor encounters another actor in the game the latter will provide an educational experience by retrieving a problem from the ITS. This problem is presented to the player, who is required to answer it. The other actor then submits the solution on the player's behalf and receives feedback, which it uses to help the player (e.g. via further dialogue).

Each actor has individual control over the ITS interaction and may tailor this interaction in several ways. First, each actor can specify what questions it may ask by restricting possible questions to those containing a certain keyword or keywords. For example, if the domain is French and the actor is a chef, she might ask questions about food, while a "taxi driver" actor might ask about places in Paris. Second, each actor can request questions from a different sub domain, allowing, for example, certain actors to ask hard questions (from an advanced sub domain) and others to ask easier questions, allowing the player to have some choice over problem difficulty (by seeking out "easy" actors). Note that sub domains do not even need to necessarily be related, thus an author could potentially develop a single ITS game that spans several (related) subjects.

Actors also have control over the structure of interactions, including what happens when the user gets an answer right or wrong. In the latter case WETAS returns a severity (from 0 to 1) as well as feedback about errors. The severity is computed based on the likelihood the student would have made that error: the less likely (i.e. the more certain the system is that they knew the underlying concepts) the higher the severity. The actor can use this to moderate their response: a high severity error may cause an actor to become less co-operative, for example.

The source code for the Greenmind classes is made available to the developer, allowing them to extend or tailor Greenmind itself (via the standard Greenfoot development environment), by modifying its base classes or developing new ones. For example, Greenmind provides a standard interface for retrieving data from the student via a free text field. However, developers can create alternate classes that interact in other ways, such as presenting radio buttons for multiple-choice questions.

3 Conclusions and Further Work

With the advent of authoring systems such as WETAS, Intelligent Tutoring Systems are maturing to the point where they could soon become mainstream. One of the challenges

is to serve ITS content to students in a way that is engaging as well as educational. ITS experts are not necessarily the best people to provide such interfaces, so the need exists to separate the user experience from the reasoning systems. Greenmind attempts to do this by building on Greenfoot, a free educational tool for developing Java programs with a 2D visual element. Greenmind takes Greenfoot's excellence as a game development environment and adds ITS capability by allowing it to communicate with the WETAS intelligent tutoring shell. Educational games are an interesting variation on classic ITS, and may prove to be an essential approach for widening the acceptance of ITS.

One of the barriers to mass use of ITS is the need for specialists to develop the system, either leading to "off-the-shelf" systems that do not meet the teachers' needs or requiring massive investment in the development of individual, customized tutors. Greenmind helps this dilemma by separating the student experience from the ITS, allowing teachers (or their developers) to create their own student experience for a given tutoring system, but re-using existing ITS development. We believe tools like Greenmind are an important step towards greater uptake of ITS in the classroom.

Acknowledgements. The author would like to thank the "Actions Per Minute" software engineering team at the University of Canterbury (Haruki Nishikawa, Johannes Pagwiwoko, James Oh, Jack Fang, Joshua Trotter and Geoffrey Clark) who built Greenmind. Their efforts and talents are much appreciated.

References

1. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A.: Intelligent Tutoring Goes To School in the Big City. International Journal of Artificial Intelligence in Education 8, 30–43 (1997)
2. Mitrovic, A.: Large-Scale Deployment of three intelligent web-based database tutors. In: Proceedings of ITI, Cavtat, Croatia, pp. 135–140 (2006)
3. Johnson, W.L., Beal, C.R., Fowkes-Winkler, A., Lauper, U., Marsella, S., Narayanan, S., Papachristou, D., Vilhjalmsson, H.H.: Tactical Language Training System: An Interim Report. In: Seventh International Conference on Intelligent Tutoring Systems, Maceio, Brazil, pp. 336–345. Springer (2004)
4. Martin, B., Mitrovic, A.: Authoring web-based tutoring systems with WETAS. In: International conference on computers in education, Auckland, pp. 183–187 (2002)
5. Furtado, V., Vasconcelos, E.: Geosimulation in Education: A System for Teaching Police Resource Allocation. Artificial Intelligence in Education 17, 57–81 (2007)
6. Chen, A.H., Chou, C.Y., Deng, Y.C., Chan, T.W.: Active Open Learner Models as Animal Companions: Motivating Children to Learn through Interacting with My-Pet and Our-Pet. Artificial Intelligence in Education 17, 145–167 (2007)
7. Henriksen, P., Kolling, M.: Greenfoot: Combining object visualisation with interaction. In: OOPSLA 2004, Vancouver, BC, CANADA, pp. 73–82 (2004)

An Experimental Use of Learning Environment for Problem-Posing as Sentence-Integration in Arithmetical Word Problems

Tsukasa Hirashima[1], Takuro Yokoyama[2], Masahiko Okamoto[3], and Akira Takeuchi[2]

[1] Information Engineering, Hiroshima University,
1-4-1, Kagamiyama, Higashi-Hiroshima, Hiroshima, Japan
tsukasa @isl.hiroshima-u.ac.jp
[2] Information Engineering, Kyushu Institute of Technology,
680-4, Kawazu, Iizuka, Fukuoka, Japan
{yokoyama,takeuchi}@ai.kyutech.ac.jp
[3] Humanities and Social Sciences, Osaka Prefecture University,
1-1, Gakuen-cho, Nakaku, Sakai, Osaka, Japan
okamoto@hs.osakafu-u.ac.jp

Abstract. We have developed a computer-based environment for learning by problem-posing as sentence-integration. The system was evaluated through two months usage in classroom by the ninety-nine students belonging to three classes of an elementary school second grade. As the results, we found that (1) the second grade students of elementary school had posed problems continuously with the system, and (2) both students and teachers answered questionnaires that the problem-posing activity using this system was useful for learning. Moreover, we confirmed that (3) our system improved the problem solving ability of low performance students.

1 Introduction

Learning by problem posing is well recognized as an important way to learn mathematics. In the USA, documents promoting curricular and pedagogical innovation in mathematics education have called for an increased emphasis on problem posing activities in the mathematics classroom. Several investigations have also suggested that mathematical problem posing had a positive influence on the learners' problem-solving abilities or their attitude toward mathematics.

However, despite the importance of problem posing, it is not popular as a learning method in reality. This is due to a few factors. First, learners can make various kinds of problems, but some of the problems may be wrong. In addition, some of the learners might repeatedly make similar problems, or make problems that are too simple to be useful for learning. In such cases, adequate feedback for each problem is required. However, because learners can make a large range of problems, it is difficult to prepare adequate feedback for every problem that learners might make. In problem posing, assessment of each posed problem and assistance based on the assessment is necessary. Because the above task puts a heavy burden on teachers, it is very difficult

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 687–689, 2008.

for teachers to use problem posing as a learning method. From this point of view, this "learning by problem-posing" is highly required to realize individual and intelligent interaction. Therefore, we have been investigating a computer-based learning environment for interactive problem-posing in arithmetical word problems [1]. We call the learning environment MOSAKUN.

In this paper, a use case of the environment for two months in elementary school second grade is reported. In this use, several computers installed the learning environment were set in several classrooms and allowed students to pose problems freely with the computers out of class time. Through this practical use, we evaluated the learning effect of the learning environment by long-term use and confirmed that whether it can be used by students by their own initiative.

2 Experimental Use of MOSAKUN in Classroom

The interface of problem-posing (Figure 1) offers, on the left side, an image blackboard as a problem-composition area. At the top of the area, a calculation expression is presented. A learner is required to pose a problem that is able to solve by the calculation expression. Here, the expression is the solution-method. Although the expression itself is easy, the learner has to consider the combination of not only a number but also a subject, object and predicate in each sentence. The three blanks in the area are the ones to put sentence cards. Sentence cards are presented at right side of the interface. A learner can move the card by drag&drop method freely in the interface. When a learner pushes "diagnosis button" under the problem-composition area, the system diagnoses the combination of sentences. The results of the diagnosis and message to help the learner's problem-posing is presented by another window.

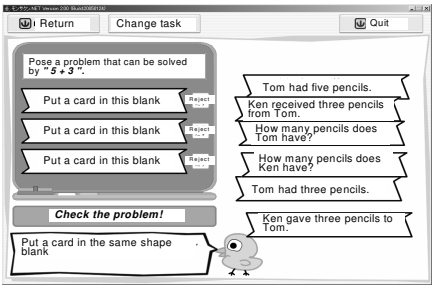


Fig. 1. Problem-Posing Interface of MONSAKUN (Currently, it can deal with Japanese only. All words were translated into English for this paper).

2.1 Situation of the Experimental Use

Purposes of this experiment were (1) to examine the learning effect of long-term use of problem-posing with MONSAKUN, and (2) to confirm whether students use MONSAKUN of their own free will. For the first purpose, we used “extraneous problem test” which includes extraneous information that is not necessary to solve the

word problem. For the second purpose, we examine the numbers of problems posed by individual students and the results of questionnaire.

Before this experimental use, two class times (ninety minutes in total) were taken as introductory use of MONSAKUN in an elementary school where each student was able to use a computer. Then, two computers with MONSAKUN are placed in each class (six computers were used in the school in total). So, about fifteen students were assigned to one computer. Teachers of the classes didn't instruct students how to use the system but made rules to share the systems. The period of the experimental use was nine weeks including 46 school days. We carried out pre-test just after the introductory use, and post-test at the end of the period.

2.2 Results of System Use

The total number of problems posed by the six systems is 8,386. In a day, 30.4 problems were posed with a system. The average used days of each student is 8.5 days. In summary, three students were used a system a day and each of them posed ten problems. As the results of questionnaires, most of the students thought that the use of MONSAKUN made arithmetic enjoyable, and they hope to use it more. Teachers who had taken charge of the classes also agreed to these considerations. Because more than fifteen students shared one system, students could not always use the system when they liked to use. Besides, the available time of the systems was only out of class time. Considering these restrictions, the above results suggest that the second grade students were continuously able to pose problems with MONSAKUN by their own will, and accepted the use as learning and enjoyable activity.

In the analysis of the test of extraneous problems, based on the average score (= 8.32) of extraneous problem test, the students were divided into two groups: a high-score group and a low-score group. Then, the students were also divided into a high-posed group and a low-posed group based the median (= 77) of the number of posed problems by each student. As the results, the number of high-score and high-posed group is thirty-two, high-score and low-posed group is twenty, low-score and high-posed groups is twelve, and low-score and low-posed group is twenty-one. As the results, only low-score/high-posed students had significantly improved their scores.

3 Conclusion

Through two months use of MONSAKUN, we found that (1) the second grade students of elementary school had posed problems continuously with the system, and (2) both students and teachers answered questionnaires that the problem-posing activity using this system was useful for learning. Moreover, we confirmed that (3) our system improved the problem solving ability of low performance students.

Reference

1. Hirashima, T., Yokoyama, T., Okamoto, M., Takeuchi, A.: Learning by Problem-Posing as Sentence-Integration and Experimental Use. In: Proc. of AIED 2007, pp. 254–261 (2007)

Automatic Analyses of Cohesion and Coherence in Human Tutorial Dialogues During Hypermedia: A Comparison among Mental Model Jumpers

Moongee Jeon and Roger Azevedo

Department of Psychology & Institute for Intelligent Systems, University of Memphis
400 Innovation Drive, Memphis, TN, 38152-6400, USA
{mjeon1,razevedo}@memphis.edu

Abstract. We analyzed cohesion and coherence in tutorial dialogues from 66 think-aloud transcripts collected from a human tutorial dialogue study which investigated the effect of tutoring on middle and high school students' learning about the circulatory system with hypermedia [1]. Our findings showed that there were significant differences in the tutorial dialogues of Jumpers (i.e., those who showed significant pretest-posttest mental model shifts about the science topic) versus No-jumpers (i.e., those who showed no significant shifts) in the semantic/conceptual similarity, readability scores, incidence scores of causal verbs and causal connectives, and turn length. We argue that the semantic/conceptual similarity of the discourse, causal verbs/causal connectives, and longer turns primarily facilitated the improvement in Jumpers' mental models and deep learning.

Keywords: Cohesion, Coherence, Human Tutorial Dialogue, Learning, Hypermedia, Human Tutoring.

1 Introduction: Cohesion and Coherence in Text and Discourse

People normally attempt to construct a coherent mental model during learning and comprehension. Cohesion and coherence in text and discourse are important factors that influence text and discourse processing. Cohesion differs from coherence in that coherence refers to characteristics of mental (or situation) models that people establish during comprehension, whereas cohesion indicates linguistic characteristics of text and discourse such as argument overlap, anaphora, discourse markers, and connectives [2].

Recent technology-based computational linguistics have motivated our research group to develop Coh-Metrix, an automated software tool that analyzes text and discourse on over 600 measures of cohesion, coherence, readability, and language [3]. Coh-Metrix was used in this human tutoring study to analyze cohesion, coherence, and readability of human tutorial dialogues.

2 Method

The original study conducted by Azevedo and colleague [1] examined how self-regulated learning (SRL) and externally-facilitated self-regulated learning (ERL) differentially

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 690–692, 2008.

affected adolescents' learning about the circulatory system while using a hypermedia environment for 40 minutes. A total of 128 middle and high school students with little prior knowledge of the topic were randomly assigned to either the SRL or ERL condition. Learners in the SRL condition regulated their own learning, while learners in the ERL condition had access to a human tutor who facilitated their self-regulated learning.

The goal of this paper is to report the analyses that were conducted on the 66 think-aloud protocols only from the ERL tutoring condition by focusing on cohesion and coherence measures extracted from the 1,250 pages of think-aloud protocols using Coh-Metrix. As such, the 66 adolescents were re-classified into 2 groups: (1) Mental model shift group (n = 33) which includes a pretest score between 1-6 and a posttest score between 7-12; and (2) No jump group (n = 33) which includes anyone whose pretest mental model score is identical to their posttest mental model score (e.g., 2 on pretest and 2 on the posttest).

2.1 Materials and Procedures for Analyzing Tutorial Dialogue Data

Based on the 66 think-aloud transcripts (i.e., tutor-student dialogues from 66 adolescents who participated in the ERL condition for this study), we split the tutor-student conversations into 'tutor-only' (i.e., those that contain only tutor turns) and 'student-only' (i.e., those that include only student turns) turns. Then, we entered only the student turns into Coh-Metrix because the primary goal of this paper is to compare cohesion, coherence, and readability of Jumpers' dialogues versus No-Jumpers' dialogues.

3 Results

We focused on a few specific measures of Coh-Metrix for this paper. Table 1 presents the means of those Coh-Metrix indices of the jump and no jump conditions. Our findings showed that (1) there was no significant difference between the Jumpers and No-Jumpers for the co-referential cohesion (i.e., the argument overlap proportions for

Table 1. Means (SDs) for the indices of Coh-Metrix by mental model classification (* $p < .05$)

Indices of Coh-Metrix	Mental Model Jumpers (n=33)	Mental Model No-Jumpers (n=33)
Argument Overlap for adjacent sentences	.05 (.02)	.05 (.03)
LSA (Latent Semantic Analysis) cosines for adjacent sentences *	.34 (.20)	.23 (.09)
LSA cosines for turn to turn *	.36 (.20)	.25 (.09)
Flesch Reading Ease score (0-100) *	91 (6.2)	87 (6.4)
Flesch-Kincaid Grade level (0-12) *	1.6 (1.1)	2.3 (1.1)
Incidence of causal verbs *	116 (138)	54 (30)
Incidence of causal connectives *	28 (22.7)	17 (10.4)
Total number of turns	161 (51)	157 (57)
Total number of sentences	212 (93)	180 (75)
Total number of words	860 (439)	847 (515)
Average sentences per turn *	1.3 (.4)	1.1 (.1)
Average words per sentence	4.2 (1.6)	4.6 (1.8)

neighboring sentence pairs); (2) the discourse of the Jumpers were more coherent locally (i.e., the higher LSA cosines for neighboring sentence pairs) and globally (i.e., the higher LSA cosines for all the turns in the tutorial discourse) than No-Jumpers; (3) No-Jumpers' discourse is more difficult to read than the Jumpers' discourse (based on the standard readability formulas); (4) the Jumpers tended to use more causal verbs and causal connectives interacting with the human tutor to gain a deep conceptual understanding for the circulatory system; (5) the Jumpers inclined to produce more sentences in a single turn (i.e., the longer turns) than No-Jumpers to assimilate critical concepts and information of the circulatory system.

4 Summary

In this paper, we analyzed cohesion and coherence in tutorial dialogues from 66 think-aloud transcripts collected from a human discourse study which investigated the effect of tutoring on adolescent students' learning about the circulatory system with hypermedia. We argue that the semantic/conceptual similarity (i.e., the local and global coherence) of the discourse, causal verbs, causal connectives, and longer turns primarily facilitated the improvement in Jumpers' conceptual mental model and deep learning for the circulatory system.

Acknowledgents. This research was supported by funding from the National Science Foundation (ROLE 0133346, ROLE 0731828, and REESE 0633918) awarded to the second author. The authors would like thank Jeffrey Greene, Daniel Moos, and Emily Siler for assistance with data collection and transcribing the audio data. We also acknowledge Dr. Graesser's Institute for Education Sciences (R3056020018-02) and National Science Foundation (REC 0106965 and ITR 0325428) grants supporting the development of Coh-Metrix.

References

1. Azevedo, R., Moos, D., Greene, J., Winters, F., Cromley, J.: Why Is Externally-regulated Learning More Effective Than Self-regulated Learning with Hypermedia? Educational Technology Research and Development 56(1), 45–72 (2008)
2. Graesser, A.C., Jeon, M., Yan, Y., Cai, Z.: Discourse Cohesion in Text and Tutorial Dialogue. Information Design Journal 15(3), 199–213 (2007)
3. Graesser, A.C., McNamara, D., Louwerse, M., Cai, Z.: Coh-Metrix: Analysis of Text on Cohesion and Language. Behavioral Research Methods, Instruments, and Computers 36, 193–202 (2004)

Interface Challenges for Mobile Tutoring Systems

Quincy Brown[1,*], Frank J. Lee[1], Dario D. Salvucci[1], and Vincent Aleven[2]

[1] Drexel University, Philadelphia, PA 19104, USA
[2] Carnegie Mellon University, Pittsburgh, PA 15213, USA

Abstract. Mobile devices used in education have the potential to provide learners with access to tutoring systems outside of the classroom or computer laboratory setting. To effectively deliver tutors on mobile devices, developers must consider the interface constraints imposed by the devices. The primary restriction is the small display available to users and the large amount of text and diagrams integral to desktop tutors. This paper describes two approaches to creating a mobile tutor interface and discusses the tradeoffs of each approach.

1 Introduction

The ubiquitous use of cell phones in society has led researchers to investigate methods to employ mobile devices in education. The integration of mobile devices and intelligent tutoring systems can deliver a low-cost one-to-one tutoring solution to a wide audience of learners. Tutors delivered on mobile devices have the potential to move tutor use beyond computer labs and traditional classrooms thereby providing learners with additional opportunities for use after school, at home, and in other locations outside of brick and mortar school buildings.

There have been tutoring systems implemented on mobile devices [1, 2]. One system allows instructors to tailor questions for students based on their previous performance [2]. The questions are then delivered to students via short messaging system (SMS). Another system investigates users learning procedural knowledge in fraction addition [1]. However, neither system addresses the issue of modifying the user interface for compatibility with mobile device displays.

One major challenge for mobile tutoring systems is that mobile device displays are small while the user interfaces for typical desktop ITS applications are large and complex. Tutors created for desktop-sized displays present users with text, diagrams, workspace, a problem, and status information enabling them to view the necessary components while using the tutor. Thus, we are faced with the challenge of reducing the complex interface to a small display in manner that does not impair user performance and student learning. In this paper we examine tutor interface components and address issues involved in representing the complex interfaces on small displays.

2 Mobile Intelligent Tutor Interface Challenges

One constraint in mobile device user interface design is the limited amount of information that can be clearly displayed. Devices with higher resolutions encounter this

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 693–695, 2008.

problem as well because font sizes as low as 10 points can be too small for users to comfortably read and applications requiring users to read or enter large amounts of text create undesirable user experiences [3]. Hierarchical design is a technique used to present mobile device users with high level information and allow them to select links to receive detailed information [4]. Another successful approach has been to use interfaces that are smaller versions of desktop sized applications.

An evaluation of tutor interfaces, such as, Carnegie Learning [5], Active Math [8], and Andes Physics Tutor [7], yields complex interfaces with multiple regions of user interaction. Each region is integral to the efficacy of the tutor thereby making each a necessary component in a mobile tutor interface. The importance of each region poses a dilemma because none can be removed without lessening the efficacy of the tutor.

The primary challenges in creating mobile tutor interfaces are the display of the complex interface and the small font size required by the smaller display. The designs presented in this paper integrate mobile device research and the limitations imposed by complex tutor interfaces. Each design introduces tradeoffs that can affect user interaction.

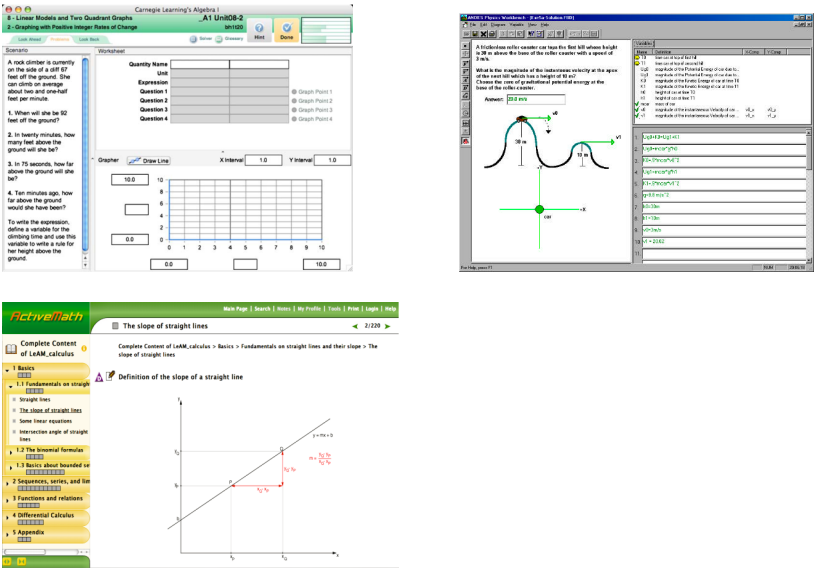


Fig. 1. Carnegie Learning [5], Andes Physics [7] and Active Math [6] Tutor interfaces

3 Preliminary Mobile Tutor Interface Designs

The designs presented in this paper integrate mobile device interface research and the constraints imposed by complex tutor interfaces. Each design presents tradeoffs that can effect user interaction. The first presents users with a smaller version of the desktop interface displaying all regions in one screen. This design was developed to minimize the navigation required to view all regions. The second design presents users

with a multi-tab interface where each tab corresponds to one region. This design improves the readability of the interface by placing each region in its own screen and utilizes larger font sizes that are easier to read. However, the larger font size is presented at the expense of increased user navigation.

4 Conclusion and Future Work

To explore the usability of the interface designs we will conduct iterations of user tests to arrive a design that balances the aforementioned tradeoffs. Upon completion of the interface design we intend to implement a mobile intelligent tutoring system. In addition to the interface issues described in this paper, our future research includes teaching and learning strategies as well as an architecture required to implement a mobile intelligent tutoring system.

Acknowledgements

**The author is supported by National Science Foundation grant #DGE-0538476, and the third author is supported by National Science Foundation grant #IIS-0426674.

References

[1] Kong, S.C., Lam, S.Y., Kwok, L.F.: A Cognitive Tool in Handheld Devices for Collaborative Learning: Comprehending Procedural Knowledge of the Addition of Common Fractions. In: Computer Support For Collaborative Learning, Taipei, Taiwan (2005)

[2] Virvou, M., Alepis, E.: Mobile educational features in authoring tools for personalised tutoring. Computers & Education 44, 53–68 (2005)

[3] Sharples, M., Taylor, J., Vavoula, G.: Towards a Theory of Mobile Learning. In: mLearn 2005, Cape Town, South Africa (2005)

[4] Hao, J., Zhang, K.: A Mobile Interface for Hierarchical Information Visualization and Navigation. In: International Symposium on Consumer Electronics, pp. 1–7. IEEE (2007)

[5] Koedinger, K.R., Aleven, V.: Exploring the Assistance Dilemma in Experiments with Cognitive Tutors. Educational Psychological Review 19, 239–264 (2007)

[6] ActiveMath, ActiveMath Main Page (2008). Retrieved April 1, 2008 from, http://commons.activemath.org/ActiveMath2/main/menu.cmd

[7] Schulze, K.G., Shelby, R.N., Treacy, D.J., Wintersgill, M.C.: Andes: A Coached Learning Environment for Classical Newtonian Physics. In: International Conference on College Teaching and Learning, Jacksonville (2000)

Agora UCS
Ubiquitous Collaborative Space

Pascal Dugénie, Stefano A. Cerri, Philippe Lemoisson, and Abdelkader Gouaich

LIRMM: Université Montpellier 2 & CNRS; 161 rue Ada, Montpellier, France

Abstract. Agora UCS is a new architecture designed for distributed learning as a side effect of communication and collaboration. This architecture aims to achieve (i) ubiquity (time and space independent access by community members); (ii) immanence (full internal control of the destiny of the community) and (iii) multi-modal communication (reinforcing the interactions between the members of the community). The theoretical model underlying Agora UCS is inspired by an integration of agents and Grid concepts (AGIL). Agora UCS has been experimented by a dozen of communities, which represent alltogether about seventy members. We achieved quite promising results in terms of motivation and collective performances[1].

1 Requirements

Conversation and collaboration is the foundation of human learning. We base our requirements on a learning theory consisting of conversational cycles [9]. This theoretical background led us to draft the essential characteristics of a collaborative environment supporting mutual understanding and joint work: (i) immediate awareness of the life of the community; (ii) ability to maintain a clear and unambiguous internal model of the working environment and ongoing processes during the successive collaboration sessions. These general ideas have grounded the concept of Agora UCS in order to: (i) allow secure access to services in a terminal independent way; (ii) support ubiquity yet keeping the state of the collaboration independently from the user access location; (iii) dynamically allocate resources for any use of any service within a pool of mutualized and virtualized physical resources.

2 Architecture

The integration of Agents [1,2,3,7] and Grid concepts [5,6] has been extensively modeled in [8]. Figure 1 represents the Agora UCS architecture which is based on a ternary relation between three concepts. The *VO (Virtual Organisation)*, a concept borrowed from Grid, is a *community* of *agents* associated with dedicated

[1] As partners from the European ELeGI project (European Learning Grid Infrastructure, IST-002205): Joost Breuker and Marc Eisenstadt are gratefully acknowledged.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 696–698, 2008.

resources (the *service container*) and *rights* which specify the level of authorisations of a *VO* member over a given *service*. This conceptual model allows to develop a flexible structure satisfying the self organisation requirement of the communities. In the figure, the inspiring Agent and Grid models are highlighted (AGR: Agent-Group Role [5], OGSA: Open Grid Service Architecture [6]).

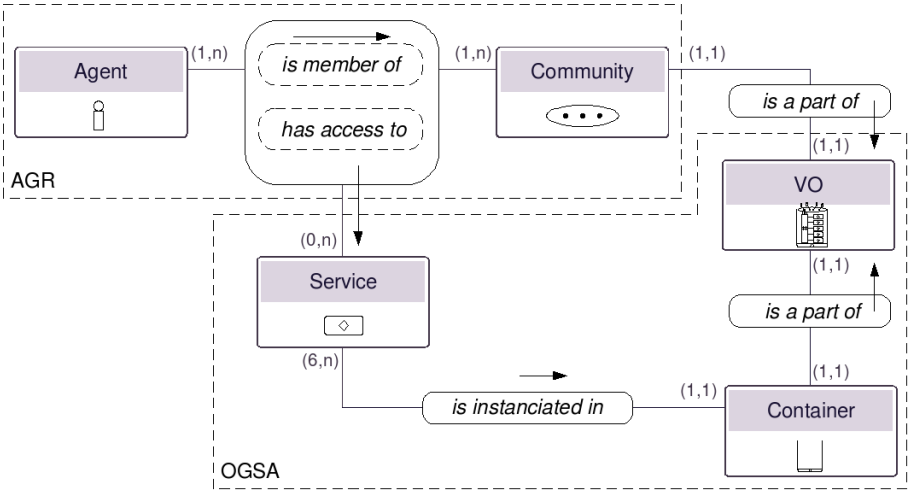


Fig. 1. The AGORA conceptual model

3 Experimentations and Results

The experimentations focus both on user behavior as well as system performance to compare subjective and objective results. Most users do not have any skill in computer management. The seamless access to AGORA UCS via a simple web browser as a *thin terminal*, was implied by the requirements. Once created, a *VO* is completely autonomous. Among others, the EnCOrE[2] scenario has provided the most interesting results. AGORA UCS allowed a quick mastering of complex computational tools: unskilled users were at ease in their operations including the delegation of rights [4]. AGORA UCS enabled, for instance, the visual representation of chemistry models at a distance so that attention was put by the users on the semantics of the domain. Since, the behavior could not be forseen in advance, the flexibility of the AGORA UCS model allowed the community to freely organise itself. Various situations of collaboration with reinforced modalities of interaction by using a synchronous communication interface has favored the emergence of collective intelligence. Discussions in real time, combined with

[2] EnCOrE: Encyclopédie de Chimie Organique Electronique.
Demo available at http://agora.lirmm.fr

visual representations on shared desktops, allowed the actors to increase the effectiveness of the collaboration process. As a conclusion, the acceptance of the technology was extremely high and promising, even if we cannot yet certify that *learning* has indeed occured as a side effect of enthousiastic collaboration at a distance.

References

1. Cerri, S.A.: Cognitive environments in the strobe model. In: Self, J. (ed.) EuroAIED: the European Conference in Artificial Intelligence and Education, Lisbon, Portugal, pp. 254–260 (1996)
2. Cerri, S.A.: Computational mathetics tool kit: architectures for dialogues. In: Gauthier, G., Frasson, C., Lesgold, A. (eds.) ITS 1996. LNCS, vol. 1086, pp. 343–352. Springer, Heidelberg (1996)
3. Cerri, S.A.: Shifting the focus from control to communication: the stream objects environments model of communicating agents, collaboration between human and artificial societies. In: Padget, J.A. (ed.) Collaboration between Human and Artificial Societies 1997. LNCS, vol. 1624. Springer, Heidelberg (1999)
4. Dugénie, P., Lemoisson, P., Jonquet, C., Crubézy, M., Laurenço, C.: The Grid Shared Desktop: a bootstrapping environment for collaboration. In: Advanced Technology for Learning (ATL), Special issue on Collaborative Learning, vol. 3, pp. 241–249 (2006)
5. Ferber, J., Gutknecht, O., Michel, F.: From agents to organizations: an organizational view of multi-agent systems. In: Giorgini, P., Müller, J.P., Odell, J.J. (eds.) AOSE 2003. LNCS, vol. 2935, pp. 214–230. Springer, Heidelberg (2004)
6. Foster, I., Jennings, N.R., Kesselman, C.: Brain meets brawn: why Grid and agents need each other. In: AAMAS 2004, New York, NY, USA, vol. 1, pp. 8–15 (2005)
7. Jonquet, C., Cerri, S.A.: The strobe model: dynamic service generation on the grid. Applied Artificial Intelligence Journal 19, 967–1013 (2005)
8. Jonquet, C., Dugénie, P., Cerri, S.A.: Service-Based Integration of Grid and Multi-Agent Systems Models. In: Kowalczyk, R., Huhns, M.N., Klusch, M., Maamar, Z., Vo, Q.B. (eds.) International Workshop on Service-Oriented Computing: Agents, Semantics, and Engineering, SOCASE 2008, Estoril, Portugal, May 2008. LNCS, vol. 5006, pp. 56–68. Springer, Heidelberg (2008)
9. Laurillard, D.: A conversational framework for individual learning applied to the learning organisation and the learning society. Systems Research and Behavioral Science 16, 113–122 (1999)

Adapte, a Tool for the Teacher to Personalize Activities

Marie Lefevre, Nathalie Guin, and Stéphanie Jean-Daubias

Université de Lyon - LIRIS, CNRS, UMR5205
43 bd du 11 novembre 1918, 69622 Villeurbanne Cedex, France
{Marie.Lefevre,Nathalie.Guin,Stephanie.Jean-Daubias}
@liris.univ-lyon1.fr

Abstract. In this paper, we want to present our tool for personalization of learning activities: the Adapte module. Our approach consists in helping teachers to provide activities suited to the abilities highlighted in learners' profiles. We will present the context of the utilization of Adapte, then we will explain its principles and its mechanism. We will finish with the research prospects of this module and the PERLEA project in witch it is integrated.

Keywords: Personalization of learning, learner's profile, generation of activities, teacher's tool.

1 Introduction

In the context of the personalization of learning, we want to provide teachers with generic tools enabling them to personalize activities they offer learners. The PERLEA project [1] aims at conceiving a system enabling teachers to manage existing learners' profiles. In this system, the Adapte module proposes to children activities suited to the abilities highlighted in their profiles. These activities are either paper and pencil worksheets or activities on an ILE.

2 Context of Utilization

A teacher uses in his classroom an ILE on geography. At the end of the learning session, this ILE generates a profile for each learner. In addition, the teacher organized for all his students the national assessments due in the beginning of year. These assessments have generated a diagnostic on the achievements, mistakes and difficulties of each pupil in mathematics and French. Thus, the teacher has for each pupil several profiles from different sources, ILE and pencil and paper, and for several disciplines. He wants to use these profiles globally so as to provide, for each pupil, personalized exercise sheets. These sheets should enable learners to be self-reliant when working and this in several disciplines. The teacher also wants to define parameters of the ILE on geography so that it proposes sessions suited to each learner's profile.

3 The Adapte Module: What Help for the Teacher?

The role of the Adapte module is to provide learners with activities suited to their profiles. These activities can be paper and pencil exercises or computerized activities managed by an external ILE.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 699–701, 2008.

In the case of paper and pencil activities, Adapte generates a worksheet matching the profile of each learner. To do so, it creates tailor-made exercises to be included in the sheet and determines the size and/or duration of the worksheet. It also provides the teacher with answers to the exercises contained in the sheet.

In the case of computerized activities, there are three scenarios depending on how the ILE is customizable by Adapte: definition of the parameters of ILE when possible; definition of an instruction sheet for the teacher to set ILE through an administrator interface; definition of a list of exercises that the student will have to do on the ILE, if this ILE is not customizable. In all cases, Adapte sets personalized sessions on the ILE according to the learner's profile. To do so, it uses ILE exercise generators or chooses exercises in the ILE database. It also determines the order in which the exercises appear, their number and the duration of the session.

4 The Mechanism of the Adapte Module

Adapte, with the help of the teacher, proposes activities adapted to the learners' profiles. In this purpose, the teacher defines a set of assignment *criteria*. These assignment *criteria* use the activity frames contained in the system or defined by the teacher. Then the system creates personalized activities. We will now go back on each step of Adapte enabling this mechanism to operate (cf. Fig. 1).

Integration of ILEs. This step is performed by an expert of the ILE to personalize. It enables to integrate the necessary technical and didactical knowledge for each ILE. The didactic knowledge contains everything related to what is taught in the ILE (discipline, practiced competences…). The technical knowledge specifies how to act on the ILE to personalize it (position of files, generators, exercise bases…). This step is compulsory so that Adapte can personalize an ILE but it is only done once.

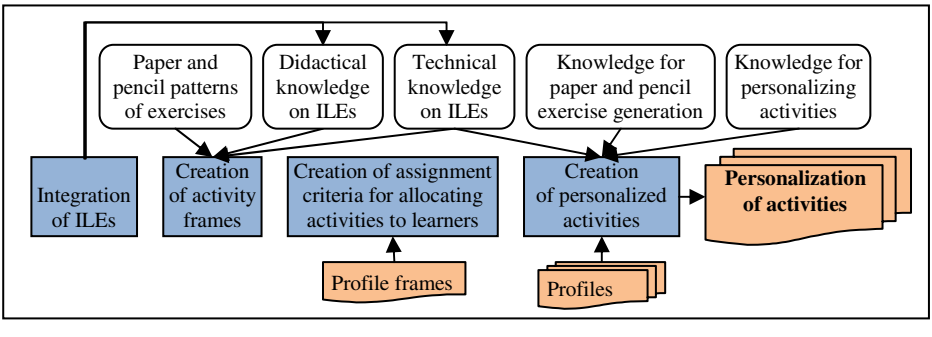


Fig. 1. Mechanism of the Adapte module

Creation of activity frames. This step is performed by the teacher to define exercises corresponding to his working habits. For the paper and pencil part of Adapte, the teacher chooses an exercise pattern (i.e. a type of exercise) and defines the constraints he wishes so that the system can generate exercises that satisfy him. For the personalization of ILE, the teacher defines the constraints of exercise generation when

the ILE to personalize contains a generator, or defines the constraints to select an exercise in the system databases. All these constraints are saved in an activity frame.

Creation of assignment *criteria* for allocating activities to learners. This step is performed by the teacher and enables him to link parts of learners' profiles to activity frames. The parts of profile frame are selected and are constrained in order to choose students with a particular difficulty or competence. For example, the teacher will choose in the learner's profile the "Command of the punctuation rules" competence and will provide a type of exercise for students with a success rate between 0 and 25%, another type of exercise for those with a rate between 25 and 75%, and nothing for students with a rate higher than 75%.

Creation of personalized activities. This step is performed by the system from the learners' profiles, the assignment *criteria* defined by the teacher and the knowledge related to either the creation of a paper and pencil worksheet, or to the creation of a session on an ILE. After Adapte has proposed its selection of personalized activities, the teacher can validate or modify the choices of the system.

5 Prospects

The Adapte module offers learners activities suited to their profiles. A first prototype implements the results defined for the generation and assignment of paper and pencil activities. This prototype enables teachers to achieve the full approach proposed by Adapte. Now, we would like to work with experts from education sciences to validate our typology of exercises and therefore all the exercises generators. We are currently focusing on the part of Adapte offering sessions adapted to the competences of the learner on an external ILE. Then, we will set up more rigorous evaluations of our results. This will be done through experiments with many teachers unrelated to the conception of the module. These experiments will involve all related modules of the PERLEA project environment, and will range from the definition of a profile frame by the teacher to the effective use of personalized activities by learners.

Reference

1. Jean-Daubias, S., Eyssautier-Bavay, C.: An environment helping teachers to track students' competencies. In: Workshop LEMORE, AIED 2005, Pays-Bas (2005)

A long version of this paper is available at this address: https://liris.cnrs.fr/publis/pdf/LIRIS-RR-2008-008.pdf?id=3411.

Framework for a Competency-Driven, Multi-viewpoint, and Evolving Learner Model

Lucie Moulet[1,2], Olga Marino[1], Richard Hotte[1], and Jean-Marc Labat[3]

[1] LICEF Research Center, Télé-université / Université du Québec À Montréal
100, rue Sherbrooke Ouest, Montréal, QC, H2X 3P2, Canada
{lucie.moulet,olga.marino,richard.hotte}@licef.ca
[2] CRIP5, Université Paris Descartes – Paris 5
45, rue des Saints Pères, 75270 Paris Cedex 06, France
[3] LIP6, Université Pierre et Marie Curie – Paris 6
104 avenue du Président Kennedy, 75016 Paris, France
jean-marc.labat@upmc.fr

Abstract. Considering learning as a dynamic, evolving, social, and lifelong process which occurs in a wide variety of contexts, we aim to improve online learning by offering learning adaptation possibilities, lifelong learning follow-up, learning evolution follow-up, openness to the learner's contexts (personal, professional, and academic), and openness to actors involved in the learning process. To achieve this goal, we propose the development of a meta-model for learner models based on competencies, integrating learner production and personal and professional information, IPP, evolving in time and taking into account different viewpoints. This meta-model may be used for creating different kind of learner models which serve different purposes: adaptation of learning to the learner's cognitive state, learning evaluation, self-reflection, team work… From these kinds of models, models representing individuals will be instantiated.

Keywords: Online learning, learner model, personalization, competency.

1 Introduction

Online learning fills the need for lifelong learning and is supported by the development of information technologies. The issues around this theme are multiple. We are especially interested in work done around actors (learner, peers, professor…) and their knowledge and competencies. In this project, we will consider learning as a dynamic, evolving, social, and lifelong process that occurs in a variety of contexts.

Our research issue is how to create a learner model that offers learning adaptation possibilities, lifelong learning and learning evolution follow-up, openness to the learner's contexts (personal, professional, and academic), and openness to the actors involved in the learning process. The learner model needs to be semantically rich, should evolve in time for learning never stops, must take into account the social dimension of learning, and should be formal, and interoperable. In this paper, we will present the conceptual model of a meta-model for learner models. The meta-model allows the design of different kinds of learner model depending on the purposes of

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 702–705, 2008.

these ones (for evaluation, collaboration, self-reflection…). And from those kinds of learner models, we can instantiate learner models modeling individuals. We start by presenting our cognitive learner model. In the following sections, we describe its multi-viewpoint and evolving characteristics as well as the mechanism on which the model interactions are based. We end with the main conclusions and the future work.

2 A Cognitive Learner Model

We propose a cognitive learner model based on competencies which integrates personal and professional information as well as learner's productions. The model is owned by the learner and connects, as needed and with the learner agreement, to different online learning systems. It is semantically rich (based on competencies), formal, and interoperable. And it may be used for different purposes: adapting learning, facilitating team work, evaluation of learning, self-reflection…

Competencies are the heart of our model. We will work with Paquette's definition [1] of a competency: a relation between actor, skill, knowledge and context. We choose this competency approach because it offers a strong semantic referencing to link the learner model with the learning resources. We assume that semantic referencing on competencies is stronger than semantic referencing on knowledge for they consider skills and context. We believe that this semantic referencing is essential for a pertinent personalization according to learner expectations and needs.

The second component of our model is the *ePortfolio*. It contains the learner's productions. Our learner model will have core competencies as well as domain competencies for each domain the learner is involved in.

The third component is the *personal and professional information (PPI)*, the learner's link to his broad world. Learning is a continuous process which occurs in different contexts, including the personal and professional ones. PPI and learner's productions are linked to the competencies they illustrate.

3 Multi-viewpoint Learner Model

We have already stated that learning takes place in the learner's different learning situations and through social interactions [2]. Different units of learning as well as actors interact with the learner and should be allowed to register the result of these interactions in the learner model. The learner model is thus seen from different viewpoints. A viewpoint is "A mental position from which things are viewed" [3].

Our learner model is a multi-viewpoint model composed of the *core learner model* and the related *learner model viewpoints*. The core learner model includes objective information such as PPI and productions, as well as the learner information that has the consensus of the different human and machine actors. Linked to this core learner model, there are as many viewpoints as there are differentiated observers allowed to modify the learner model (tutors, professors, peers, learning units, etc.) Those viewpoints are not pre-established but settled by the learning context of the learner (their can be a view point by individual or a viewpoint by roles (learner, professor…)). A viewpoint contains competencies along with links between those competencies and the learner's PPI and productions.

When a modification is proposed by one of the actors in the system, the viewpoint corresponding to this actor is updated with the modification and a modification proposal mechanism is launched. This mechanism is defined outside of the model and takes into account organizational hierarchies and conflict solving strategies. If agreement is obtained, the modification is reflected in the core model.

4 Evolving Learner Model

For us, learning is a knowledge building process proceeding from the interaction between a learner and his/her environment [4]. From this environment the learner pulls off his/her learning conditions. We can then consider learning as a social act which evolves taking into account interaction with the learner's environment, making it a dynamic learning process. A learner model has to reflect this dynamic characteristic and must be evolving too. An evolving learner model updates itself according to the learning progress. Thus, during learning, different versions of the model will be created. Those versions need to be managed.

To deal with this evolutionary dimension, the learner model proposed in the preceding section will be extended to include different learner model versions. For this, we will integrate a version control system into our learner model. The model can then be searched by viewpoint or by version, and the evolution of a competency can be tracked in time.

5 Learner Model Interaction

We state that the learner is the owner of his/her model. Thus, any manipulation of the model should have the agreement of the learner. As the model will be in interaction with actors and online learning systems, the learner should agree on those actors and systems interactions. For this, we propose to use contracts. Contracts are used in computer science to specify the relations between different actors (or components or systems or models…) [5]. For example, if a learner wishes to take a course at a specific university, this latter may require accessing all information contained in the learner model. This access allowance could be specified in a contract. More precisely in our project, contracts will allow the specification of which kinds of actors can interact with the model and which action each of these actor kinds can take. Contracts can also specify when modifications can take place (evolution characteristic of the model).

Interactions can be of two kinds. The first one consists of extending the model with new information. The second one consists of consulting the information contained in the model (to adapt learning for example).

6 Conclusion and Further Work

This research project aims to improve online learning by offering learning adaptation possibilities, lifelong learning follow-up, learning evolution follow-up, openness to the learner's contexts (personal, professional, and academic), and openness to actors

involved in the learning process. To achieve this goal, we propose the development of a meta-model for the design of learner models based on competencies that integrate learner production and IPP, that evolve in time as well as learning progress, and that take into account different viewpoints while remaining interoperable. Those models may be of different kinds depending on their purposes: adapting learning to the learner's cognitive state, learning evaluation, self-reflection, teamwork… The strength of this model is its ability to integrate different elements in the same learner model: a learner cognitive model (knowledge and competencies), a learner production model, a multi-viewpoint model, and an evolving model.

The next step of this project is to complete a prototype development. This development has begun and is based on Java, using Castor Project to map Java objects and XML files, and integrating the Subversion version control system to manage the model's evolution in time.

References

1. Paquette, G., Rosca, I.: An Ontology-based Referencing of Actors, Operations and Resources in eLearning Systems. In: SW-EL 2004 Workshop, Einhoven (2004)
2. Lasnier, F.: Réussir la formation par compétences. Guérin, Montréal (2000)
3. WordNet, lexical database for the English language, `http://wordnet.princeton.edu/`
4. De Vries, E., Baillé, J.: Apprentissage: référents théoriques pour les EIAH. In: Grand-Bastien, M., Labat, J.-M. (eds.) Environnements informatiques pour l'apprentissage humain, Hermès/Lavoisier, Paris, pp. 27–46 (2006)
5. Bachman, F., Bass, L., Buhman, C., Comella-Dorda, S., Long, F., Robert, J., Seacord, R., Wallnau, K.: Technical Concepts of Component-Based Software Engineering. Technical Report, vol. II. Carnegie Mellon University, Pittsburgh (2000)

Use Chatbot CSIEC to Facilitate the Individual Learning in English Instruction: A Case Study

Jiyou Jia[1] and Meixian Ruan[2]

[1] Department of Educational Technology, School of Education
Peking University. Beijing, 100871, China
jjy@pku.edu.cn
[2] Jingxian School, Jiangmen, Guangdong, China
ruanmeixian@sina.com.cn

Abstract. CSIEC (Computer Simulation in Educational Communication), is not only an intelligent web-based human-computer dialogue system with natural language for English instruction, but also a learning assessment system for learners and teachers. Its multiple functions including grammar gap filling exercises, talk show and chatting on a given topic, can satisfy the various needs from the students with different backgrounds and learning abilities. In this paper we present a case study of the integration of CSIEC's multiple functions into English syllabus design in a middle school and its pedagogical effectiveness. The comparison of two examination results before and after the system integration shows great improvement of students' performance, and the survey data also indicates the students' favor to this system.

1 Brief Introduction of CSIEC System

Brennan defined a chatbot as "an artificial construct that is designed to converse with human beings using natural language as input and output" [1]. Since 1990s with the development of natural language processing, the usage of chatbot systems in education is drawing more and more attention from researchers in related fields.

CSIEC (Computer Simulation in Educational Communication), firstly introduced in [2], is one of the most early chatbots applied in Education. It originally aimed at providing the English learners a virtual talking partner which can be accessed anytime anywhere via Internet. After continuous development its current pedagogical functions include automatic scoring of gap-filling exercises without defined answers, listening training, talk show of two robots, multimodal user interface and selectable chatting pattern, free chatting adaptive to user preference and topic, and guided chatting in given scenarios, and scoring mechanism, etc. So it is not only an intelligent web-based human-computer dialogue system with natural language for English instruction, but also a learning assessment system for learners and teachers. Therefore it is on one side freely accessible by the Internet users, most of which are students at different levels, on the other side integrated in graduate students English instruction, and middle school English class. In this paper we explore its application in middle school English classes to facilitate the individual study.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 706–708, 2008.

2 Syllabus Design: Integration of CSIEC into Individual Learning

The experiment class is in Grade One of a middle school located in Guangdong Province. The school is equipped with modern computer rooms with Internet access and projectors. The 50 students in the class graduated from primary schools with different teaching levels, and their entrance English examination scores varied greatly. The mean was 64.39 of 100, the minimal score was 0, maximal was 94, and the standard deviation was 20.129. The average score was ranked number 16 in all 16 classes in Grade One, and was 15.3 less than the number one class.

How to treat the 50 different students is a great challenge to the sole female English teacher of the class. Purely the checking of students' exercise outcome is a great burden for her. The teaching content is the same. If the teaching methods are also the same, the students' difference will be ignored, and the fair teaching can't be guaranteed. If the teacher separates the students into several groups explicitly, the students with lower scores will feel disregarded and frustrated.

The CSIEC system with its multiple functions supplies a good opportunity to apply individual instruction for the different students. We illustrate how to use it to facilitate the individual learning through the example of a teaching unit syllabus design and implementation: "*How much is it?*". The students were classified into A, B, C three groups according to their entrance examination score: A→[80, 100], B→[60, 80), C→[0,60). The students in group A, B, C made up 18%, 56%, and 26% of the total students, respectively. We just wrote this classification into the students table of CSIEC system, but didn't inform the students of it. In this sense it was an implicit classification, and could protect the self-respect of the students in lower groups. The students in group C were required to do more exercises and drills than those in B and A, and the ones in B were required to do more than those in A.

We used gap-filling exercises without defined answers to drill the students in the mastery of grammar knowledge and useful expressions, and transformed a dialogue example in the textbook into a sentence filling exercise so that the students should learn the expressions by heart as exactly as the ones in the textbook. For them in Grade 1 the reciting of textbook content is necessary for the language learning.

We designed 6 pieces of talk shows for the given topics in this unit, as well as 6 pieces of human-computer interactive dialogues for the topics in this unit. The guided chatting on one given topic requires that the user is familiar with the chatting content, for example after watching the talk shows, and should try not to escape from this topic. So the marking mechanism in the chatting will praise the user with a high extra mark if he/she keeps the topic, otherwise punish him/her with a low extra mark. All the students must take part in the six dialogues, and must get certain points of extra mark in every dialogue according to their group.

If one student has finished the required exercises, he/she will win a star from the system as congratulation to his/her achievement. The teacher is also authorized to check the status of students. She can send emails to any student to praise him/her for the good performance or remind him/her to finish the required exercises on time.

Every week one hour's English class was held in the computer room with Internet access, so that the students could do the exercises together. Additionally as most student families had computers at home, they could also do the exercises after school.

3 Students' Perception about CSIEC Application and Examination Performance Improvement

At the end of the school term an online survey was conducted to investigate the students' attitude toward CSIEC application. The collected questionnaire data shows the students agreed the design of the system functions adaptive to the textbook can benefit their English learning. 41% of them used the user log to review their exercise history, and 56.4% thought the user log function can supervise their learning activities. All the students were willing to recommend this system to others. 48.7% of them used the system in the time span from 1 hour to 5 hours, 10.3% more than 10 hours, 17.9% between 5 hours and 10 hours. 76.9% of them hoped to use the CSIEC in the whole English instruction very much, 23.1% hoped to use it in the whole English class.

In the midterm exam the students in the experiment class achieved very good scores. The mean was 90.81, the median 93.25, and the standard deviation was only 9.572. The average score was ranked number 2 in all 16 classes in Grade One, and only 0.2 less than the number one. Compared with the entrance exam scores, all students improved their scores. Those with lower entrance scores got much more increase.

4 Conclusion and Discussion

Rooted in the constructivist learning theory and situated learning theory, the CSIEC system attempts to create situated practice for English learners and suitable situation to motivate the learning activities. The comparison of the pretest and posttest performance shows the students in the experiment class greatly improved their English skills. Moreover, compared with other 15 classes in the same grade without using the CSIEC, the collective performance improvement of this class was also remarkable. Surely many factors caused the score improvement. But because only this experiment class used the CSIEC system just between the two tests, the great score improvement can't be irrelevant with the CSIEC application. We should keep on integrating the CSIEC into English syllabus in this experiment class to watch the long-term students' performance and to compare it with other classes.

Acknowledgments. We thank KAS Germany, Ministry of Education China, Peking University, and Korea Foundation for Advanced Studies for supporting our projects.

References

1. Brennan, K.: The managed teacher: emotional labour, education, and technology. Educational Insights 10(2), 55–65 (2006)
2. Jia, J.: CSIEC (Computer Simulator in Educational Communication): A virtual context-adaptive chatting partner for foreign language learners. In: Kinshuk,, et al. (eds.) Proceedings of ICALT 2004, pp. 690–692. IEEE Press, NY (2004)

Using an Adaptive Collaboration Script to Promote Conceptual Chemistry Learning

Dimitra Tsovaltzi[1], Bruce M. McLaren[2,1], Nikol Rummel[3], Oliver Scheuer[1], Andreas Harrer[4], Niels Pinkwart[5], and Isabel Braun[3]

[1] Deutsches Forschungszentrum Für Künstliche Intelligenz (DFKI), Germany
[2] Carnegie Mellon University, U.S.A.
[3] Albert-Ludwigs-Universität Freiburg, Germany
[4] Katholische Universität Eichstätt-Ingolstadt, Germany
[5] Technische Universität Clausthal, Germany
Dimitra.Tsovaltzi@dfki.de

Abstract. Chemistry students often learn to solve problems algorithmically, applying well-practiced procedures to problems. Such an approach may hinder development of conceptual understanding. We propose to promote conceptual learning by having pairs of students collaborate on problems in a virtual laboratory (VLab), assisted by a computer-mediated collaboration script that guides the students through the stages of scientific experimentation by adapting to a particular student's (or dyad's) skills. In this paper, we report on our early steps toward this goal, including technology development and an initial wizard-of-oz study.

1 Research Motivation

Chemistry educators face the challenge of teaching students to solve problems conceptually rather than simply apply mathematical equations. Students struggle in solving problems similar to ones in textbooks or in the classroom, because they do not grasp the similar underlying concepts [1]. Research in chemistry education has suggested that collaborative activities can improve conceptual learning. While there have been very few controlled experiments which have investigated the benefits of collaborative learning in chemistry, evidence that collaboration is beneficial exists in other disciplines, such as physics [2]. This evidence has led us to investigate the potential advantages of collaborative activities for conceptual learning in chemistry.

When learning collaboratively, learners do, however, often not benefit as much as they could, because they fail to engage in productive forms of interaction. This observation suggests supporting students with *collaboration scripts*. By scripting collaboration we mean providing prompts and questions that guide students through collaborative work (e.g., [3]). However, it is possible to *over-script*, i.e., provide too many scaffolds, or overwhelm students with the concurrent demands of collaborating, following script instructions, and trying to learn [4]. To avoid these bottlenecks of collaboration we propose to use *adaptive scripting*: altering and/ or fading scripts depending on the collaborators' need for support. Adaptive scripts can be considered a form of intelligent tutoring [5] as feedback is provided based on the individual student (or group) performance.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 709–711, 2008.

We hypothesize that computer-mediated collaboration within an experimental framework - and guided by a script - can promote conceptual chemistry knowledge. We also believe that *adaptive* scripting will promote conceptual chemistry knowledge even more. This paper outlines our experiences with a wizard-of-oz study that tested the adaptive scripting approach.

2 Technology Development

The implementation of the adaptive scripts into a collaborative setting involved developing collaborative extensions to the VLab chemistry experimentation tool [6], through integration with FreeStyler, an existing collaborative software environment [7]. Our script consisted of experimentation steps inspired by [8] and was implemented in FreeStyler with tabs representing the experimentation steps and additional ordering restrictions.

3 Wizard-of-Oz Study

We conducted a small wizard-of-oz study that compared an adaptive and a non-adaptive version of our system. There were 3 dyads per condition, and the experimental procedure was standard pre-test/intervention/post-test. The students in both conditions collaborated by using a number of tools that scripted and supported their collaboration within FreeStyler tabs.

In the adaptive condition, a human wizard who observed the students as they collaborated provided adaptive support via prompts sent to the students, to promote explanations, reflection, and help giving/receiving. The wizard used a flowchart to observe and recognize situations requiring a prompt, and to choose and give the appropriate prompt. The flowchart was developed based on the data analysis of the first study and on a literature review of collaborative learning research (e.g., [2]). Examples of adaptive prompts are *"Don't forget to explain your statements and actions to each other."* which was provided by the wizard when a student neglected to explain a statement or action despite a request by his partner, and *"Remember to talk about and reach a consensus on your next activity before moving on."* which was provided when a student started an activity alone before agreeing on previous activities with his partner.

The study results were encouraging. With a possible highest score of 6 points on the conceptual post-test, the adaptive condition mean was M=4.6 (SD 1.63) compared to M=3.5 (SD 2.81) for the non-adaptive condition, showing a tendency toward better conceptual understanding due to adaptive support. A process analysis of screen recordings taken during the experiments showed that the non-adaptive dyads were less likely to correct flaws in their collaborative and script practice. On the other hand, the prompts given to the adaptive dyads, although not always appreciated by the students, had a clear positive effect on collaboration, motivation, and to some extent on the way the collaborating students followed scripts.

4 Conclusion

We will use the knowledge gained from the wizard-of-oz study presented here to extend the system towards a full collaborative intelligent tutoring system by automating our adaptive feedback. To this end, one of the approaches we will explore is the use of machine learning to create "adaptive detectors", similar to the work of Baker in developing gaming detectors [9]. That is, we will annotate student actions in the VLab for aberrant behavior and apply machine-learning algorithms to identify situations in which prompts are necessary. We also plan to use the collaboration expertise captured in the wizard flowchart as guidance for feedback in particular situations.

Acknowledgements. The Pittsburgh Science of Learning Center (PSLC), NSF Grant # 0354420, provided support for this research.

References

1. Gabel, D.L., Sherwood, R.D., Enochs, L.: Problem-Solving Skills of High School Chemistry Students. Journal of Research in Science Teaching 21(2), 221–233 (1984)
2. Hausmann, R.G., Chi, M.T.H., Roy, M.: Learning from Collaborative Problem Solving: An Analysis of Three Hypothesized Mechanisms. In: Forbus, K.D., Gentner, D., Regier, T. (eds.) 26th annual Conference of the Cognitive Science Society, pp. 547–552. Lawrence Erlbaum, Mahwah (2004)
3. Kollar, I., Fischer, F., Hesse, F.W.: Collaboration scripts - a conceptual analysis. Educational Psychology Review 18(2), 159–185 (2006)
4. Rummel, N., Spada, H.: Learning to Collaborate: An Instructional Approach to Promoting Collaborative Problem Solving in Computer Mediated Settings. Journal of the Learning Sciences 14(2) (2005)
5. VanLehn, K.: The Behavior of Tutoring Systems. International Journal of Artificial Intelligence in Education 16(3), 227–265 (2006)
6. Yaron, D., Evans, K., Karabinos, M.: Scenes and Labs Supporting Online Chemistry. In: The 83rd Annual AERA National Conference (2003)
7. Harrer, A., Pinkwart, N., McLaren, B.M., Scheuer, O.: How Do We Get the Pieces to Talk? A New Software Architecture to Support Interoperability Between Educational Software Tools. In: The 9th International Conference on Intelligent Tutoring Systems (ITS 2008) (2008)
8. De Jong, T.: Scaffolds for Computer Simulation Based Scientific Discovery Learning. In: Elen, J., Clark, R.E. (eds.) Dealing with Complexity in Learning Environments, pp. 107–128. Elsevier Science Publishers, London (2006)
9. Baker, R.S.J.d., Corbett, A.T., Koedinger, K.R., Evenson, S., Roll, I., Wagner, A.Z., Naim, M., Raspat, J., Baker, D.J., Beck, J.: Adapting to When Students Game an Intelligent Tutoring System. In: Proceedings of the 8th International Conference on Intelligent Tutoring Systems, pp. 392–401 (2006)

Towards an Intelligent Emotional Detection in an E-Learning Environment

Iness Nedji Milat[1], Hassina Seridi[2], and Mokhtar Sellami[1]

[1] Research Laboratory in Computer Science (LRI)
[2] LabGed Laboratory
University of Badji Mokhtar at Annaba, Algeria
BP 12, 23000 DZ
{iness,seridi,sellami}@lri-annaba.net

Abstract. Several research in psycho-pedagogy showed that the relevance of the training implies as many intellectual aspects as socio-emotional and the emotional state of the learner influence directly his performance in a positive or negative way. For that, the e-learning systems must take into account the emotional state of learners in order to favor their training. However, it is important to underline the implementation problem which is posed and summarized in the detection and the interpretation of the emotions which are not directly observable by the machine and which are generally expressed by a whole of behaviours whose indicators are either the words used, or the voice ton, the gestures and body attitudes or facial expressions. In this paper, we propose emotionally intelligent system architecture dedicated to the learning activities. We focus ourselves more particularly on the process of emotional recognition, ensured by agent EMOTIO, from a bimodal analysis of the speech and text used by a speaker's learner, in a training session, in order to improve the recognition precision (precision in acoustical analysis is 63.44 % but precision in bimodal analysis is 71.2%). This analysis is based on the indices extraction on two linguistic levels: prosodic and lexical.

Keywords: Emotional agent, Recognition, speech, prosodic cues, textual cues, e-learning, learner.

1 Introduction

Recent research in neurosciences and psychology [1] showed that the emotions have influences on several behavioural and cognitive processes, such as the attention and the memorization in the long term, and have a preponderant role in the communication process where the quality is an important aspect in training tasks. Consequently, in the learning activities, the emotional state of the learner influences directly his performance in a positive or negative way [2].

For these reasons, it is important that the systems dedicated to the training are endowed with a certain intelligence called emotional which allows recognizing the current learner emotional state, in order to induce emotions favoring his training [3]. However, this intelligence carry some implementation problems, because learner's

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 712–714, 2008.

emotional expressions expressed generally by a verbal or not verbal behaviours [4], are not directly observable by the machine and research works in human machine interaction was mainly focused on the set up of automatic characterization models and the vocal expression of the emotion by acoustic analysis [5,6]. Also, the facial expression by analysis of several indices of a physiognomic order [7], of the textual expression by analysis of several indices of the lexical, morphological, syntactic, semantic and dialogical type [8], and finally, the physiological expression by analysis of certain indices of physiological and expressive orders [6]. In our study, we plan to combine the two methods, vocal and textual, that the speech integrates to improve the recognition precision of the learner emotional state.

2 Contextual Framework: The EMOTORAT System

We are interested here in the emotional treatment in learning context where an emotionally intelligent system is conceived, dedicated to the emotional recognition and management of learners baptized EMOTORAT (see Fig 1).

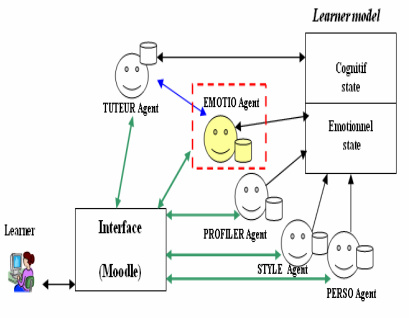


Fig. 1. Architecture of EMOTORAT

The EMOTORAT system integrates several agents, designed for the learner emotions management, and a learner model containing a cognitive state (knowledge, competences, historic, pedagogical traverses …) and an emotional state (moods, emotions, psychological profile, preferences, styles of training,…). EMOTORAT also integrates an interface module accessible by the learner and which contains the whole of the interactions tools necessary for the training.

3 EMOTIO: An Emotional Recognition and Management Agent

In its emotional recognition method, EMOTIO is based on a bimodal perception of the emotional expressions. It detects emotions from two types of information: voice and text, and classify them according to five basic types, joy, sadness, anger, fear and neutral. This bimodal analysis will give more precision for the recognition of the leaner current emotional state. The final emotional state is given by combining the results of the prosodic and textual analyses with the history of the learner emotional states. The architecture of the EMOTIO agent is based on: the learner emotional recognition (ER), which is the purpose of this article, and the emotional management (EM) to guarantee an effective training. Emotional recognition (ER) module's architecture, as schematized in Fig 2, is based on two sub modules for recognition: an emotional recognition by vocal analysis (ERVA) and an emotional recognition by textual analysis (ERTA).

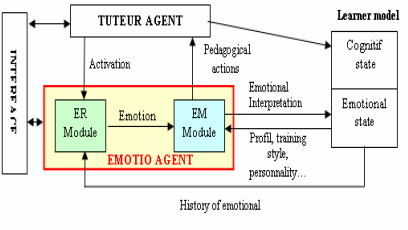


Fig. 2. Architecture of EMOTIO

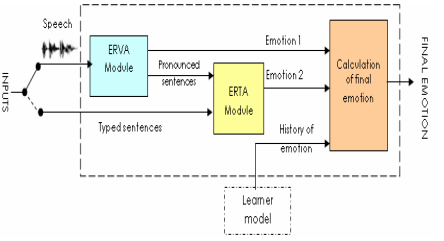


Fig. 3. Architecture of ER module

The ERVA sub module consists in identifying the emotion expressed in the speaker speech by the voice analysis which consists to a signal pre-treatment that allows preparing the data received sensor to the following analysis phase devoted to parameters extraction which the goal is to extract the emotion characteristic property. **The sub Module ERTA** goal is the emotional detection from lexical information and classify them according to five basic studies types. For that, we suppose that the emotional orientation of a sentence in entry is primarily represented by the appearance of two types of words: the keywords with emotional reference, which provide a fundamental emotion description of a sentence, and the words of emotional modification, which can intensify or eliminate the emotional state.

4 Conclusion and Perspectives

The emotional detection is certainly not perfect; we are still far from being able to concretely determine the demonstrations subjacent with a given type of emotion. Owing to a part of signals complexity and on the other extreme variability intra and inter individual. Nevertheless, the idea of this bimodal recognition based speech and text gives us a satisfaction as for the improvement of the emotional perception precision.

References

1. Salovey, P., Mayer, J.: Emotional Intelligence. Imagination, cognition and personality (9), 185–211 (1990)
2. Chaffar, S.: l'utilisation d'un agent émotionnellement intelligent dans les systèmes de e-learning (2004)
3. Schröder, M., Cowie, R.: Towards emotion-sensitive multimodal interfaces, HUMAINE, Réseau d'excellence, http://emotionresearch.net/aboutHUMAINE/
4. Plantin, C.: Structures verbales de l'émotion parlée et de la parole émue, CNRS 5
5. Devillers, L., Vasilescu, I., Mathon, C.: Prosodic cues for perceptual emotion detection in task-oriented Human-Human corpus, ICPHS, Barcelona (2003)
6. Nkambou, R., Heritier, V.: Reconnaissance émotionnelle par l'analyse des expressions faciales dans un tuteur intelligent affectif, TICE (2004)
7. Cowie, R., Cowie, D.: Emotion recognition in human-computer interaction. Signal Processing Magazine 18, 32–80 (2001)
8. Bisognin, L., Pesty, S.: Agents, Langage et Emotions: un prototype d'agent émotionnel, AGENTAL, Journée ATALA, Paris (2004)

How Do We Get the Pieces to Talk? An Architecture to Support Interoperability between Educational Tools

Andreas Harrer[1], Niels Pinkwart[2], Bruce M. McLaren[3,4], and Oliver Scheuer[3]

[1] Catholic Univ. Eichstätt-Ingolstadt
[2] Clausthal Univ. of Technology
[3] DFKI (Germany)
[4] Carnegie Mellon University (USA)

Abstract. For many practical learning scenarios, the integrated use of more than one learning tool is educationally beneficial. In these cases, interoperability between learning tools – getting the pieces to talk – is a crucial requirement that is often hard to achieve. This paper describes an architecture that aims at the integration of independent learning tools into one collaborative learning scenario.

1 Introduction

In the field of educational technology, there have been numerous attempts in recent years to connect differently targeted learning tools to one another. For example, a teacher may want to start her course with students' individual intelligent tutoring sessions, followed by plenum discussions about the topic, then some small-group work with simulation tools, and finally followed by individual essay writing. For each of these sessions, there is likely to be a different tool that is suitable to support the activity, such as Cognitive Tutors, discussion support tools, and educational simulations, yet these individual tools do not usually interoperate or exchange data, resulting in scattered artifacts and functionality that is hard to integrate.

Especially in the field of collaborative learning, tool interoperability and data exchange between heterogeneous learning tools is a crucial requirement. This is so because the data flows in group learning tend to be more complex and require data exchange between a greater variety of tools (such as discussion and graphical mapping tools) and more instances of such tools (e.g., one per student) than in individual learning scenarios. Many collaborative software tools have the advantage that they need to externalize their data anyhow to allow users to exchange data with peers (and thus transfer this data between applications). This characteristic should facilitate inter-tool data exchange – yet, reality has shown that in the field of educational technology, not many collaborative learning tools are interoperable.

In our efforts to implement longer-term learning flows, we have frequently encountered the need to make our tools – such as Cool Modes [1] or FreeStyler – interoperable with other tools. A recurring approach we have adopted is to employ a generic data storage and exchange mechanism, and use this to achieve seamless communication with learning components through the use of *adapters*. This solution principle, which we call the "Scalable Adapter," has proven successful in a number of different scenarios and projects. Thus, we propose this principle as a general software design pattern [2],

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 715–718, 2008.

i.e. a re-usable solution to the named problem,and discuss it together with its implementation in the remainder of the paper.

2 The "Scalable Adapter" Design Pattern

This section describes the "Scalable Adapter" design pattern which constitutes a software architecture that can be used to create interoperability between different educational tools, in particular between collaborative learning tools and intelligent tutors.

The *problem context* is that there exist learning tools (e.g. discussion tools, simulation tools, or ITS systems) that each provide specific functionality and data. Part of this data can be used to inform other tools within an integrated learning scenario. The recurring *problem* to solve is that the different preexisting learning tools need to interoperate with each other through data exchange. Potentially, each environment is interested in only specific portions of the data of other applications. Since it cannot be foreseen which applications need which parts of the data, a flexible design solution is required. Another *force* to be considered is that the existing learning tools should not need to be altered (or at least not much) in order to facilitate their use in their original context. Yet, the interoperation and data exchange between the systems must be supported in a flexible way to allow for arbitrary learning data to be exchanged.

The *solution* we propose is to extend existing learning tools with specific adapter components [2] that provide the interoperability with other components. The granularity of the information to be exchanged between components is tailorable in a scale-free way through the use of a composite data structure. The **Adapters** leave the original learning tools unaltered for the most part. They provide the interface for interaction between the learning tool that the adapter is attached to and the other components used within an integrated learning scenario. The **Composite Data Structure** provides access to arbitrary parts of the data to be shared between the learning applications. Additionally, a subscription mechanism for parts of this data structure is provided. This mechanism uses notifications to inform registered learning tools about changes in shared data (parts), thus avoiding inefficient communication via active polling processes. The **learning tools** use the functionality of the adapter to gain access to the data elements of interest. The processing of the data (i.e., the interpretation of the exchanged learning data) is fully encapsulated within this component.

The learning tools and the composite data structure are completely decoupled (i.e., the shared data is separated from the specific tools), with the adapter assuming a mediating function between these two. The composite data structure provides access to arbitrary data elements using a tree structure (de-)composition. Note that there is a 1-to-n relation between the data structure and the adapters and a 1-to-1 relation between an adapter and a learning tool. This implementation requires both minimal changes to the learning tools (only the communication with the adapter has to be developed) and allows multiple learning tools to access the shared data. The composite data structure allows different learning tools to use different (or the same) parts of the shared data.

The typical component interaction in this micro-architecture is that a learning tool lt_1 sends out data changes (e.g. learner actions) via an adapter a_1 to the composite data structure. This notifies all registered adapters about changes to the respective components of

the structure, enabling the adapters to process relevant information only and thus to communicate efficiently. Each adapter a_i updates its learning tool lt_i. This way, application lt_i can be informed about the relevant changes caused by students or system actions.

3 Using the "Scalable Adapter" in the CoChemEx Project

The Scalable Adapter pattern was used within the CoChemEx project [3], where the virtual chemistry experimentation laboratory "VLab" [4] is used within different collaborative inquiry learning scenarios. Here, the features for collaboration and communication of the shared workspace systems FreeStyler / Cool Modes environments [1], such as chat and graphical argumentation, are used in conjunction with the sophisticated experiments students can conduct within the VLab.

In the CoChemEx project, the educational scenario was defined by researchers and practitioners from educational psychology and chemistry education. The scenario is implemented by a collaboration script [3] consisting of several phases of activities. These are represented by separate tabs in the FreeStyler environment.

In this learning scenario, interoperability and data exchange between the different pre-existing tools – in particular, between VLab and FreeStyler – was a crucial requirement. Following the Scalable Adapter design pattern principle, this interoperability – and thus the integration of the VLab into a collaborative context – was achieved via a newly developed *VLab adapter* that creates a communication channel to FreeStyler through the jointly-used data stored in a *composite data structure* (MatchMaker). The use of the Scalable Adapter pattern to connect the applications has two immediate advantages. First, it enables the VLab to interact with other VLab instances of the collaborators, thus supporting collaborative experimentation, and, second, it enables data exchange between the experimentation functions in VLab and the FreeStyler tools which are valuable for experimentation, such as hypothesis generation and documentation of experiments. Both features are important contributions towards richer learning experiences that only integrated solutions combining various learning tools can achieve.

One conceptual challenge we still must tackle is that for every learning tool, a specific adapter must be defined. Also, the shared composite data structure is currently scenario-specific. Mechanisms for a more flexible configuration of the data structure by explicit specification of structure levels seem feasible and preferrable to hard-coding all dependencies.

Acknowledgements. The Pittsburgh Science of Learning Center, NSF Grant 0354420, supported this research. We are grateful to Nikol Rummel, Dimitra Tsovaltzi and all our student workers for their valuable contributions to the CoChemEx project.

References

1. Pinkwart, N.: Collaborative Modeling in Graph Based Environments. PhD thesis, Universität Duisburg-Essen (2005)
2. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns. Elements of reusable Object-Oriented Software. Addison-Wesley, Boston (1995)

3. McLaren, B.M., Rummel, N., Tsovaltzi, D., Braun, I., Scheuer, O., Harrer, A., Pinkwart, N.: The CoChemEx project: Conceptual chemistry learning through experimentation and adaptive collaboration. In: Proceedings of the Workshop on Emerging Technologies for Inquiry Based Learning in Science at AIED 2007, Los Angeles, CA (2007)
4. Yaron, D., Evans, K., Karabinos, M.: Scenes and labs supporting online chemistry. In: Proc. of 83rd Annual AERA National Conference (2003)

Cognitive Load Estimation for Optimizing Learning within Intelligent Tutoring Systems

François Courtemanche[1], Mehdi Najjar[2], and André Mayers[1]

[1] Department of Computer Science, University of Sherbrooke
[2] Interdisciplinary Research Center on Emerging Technologies, University of Montreal
{f.courtemanche,mehdi.najjar,andre.mayers}@usherbrooke.ca

Abstract. This paper presents a guided learning strategy model for dynamic pedagogical tailoring within intelligent tutoring systems (ITS). The proposed model is based on the integration of the cognitive load theory within an ITS architecture. Our approach takes into account the cognitive limitations of the student in order to offers personalised learning via instructional guidance.

1 Introduction

Most intelligent tutoring systems (ITS) use pedagogical objectives or performance measures in order to adapt tutoring strategies [1,3,11]. A number of researches in educational psychology suggest that highly effective instruction can only be attained by taking into account the learner cognitive constraints [4,10]. This paper introduces a novel approach for guided learning within ITS which aims to dynamically adapt instruction in order to respect the student cognitive limitations. The approach is based on the cognitive load theory (CLT) framework and uses a working memory simulator. The remainder of the paper is organised as follow. Section 2 exposes the CLT framework. Section 3 expounds the integration of the CLT concepts in our ITS and concluding remarks are given in section 4.

2 The Cognitive Load Theory

The cognitive load theory is a framework representing characteristics of the mental effort that results from the performance of complex cognitive tasks during learning [2,8,9]. CLT is based on the interaction between the human cognitive architecture and knowledge structures in order to identify optimal methods of instruction.

Cognitive load is defined by three components: intrinsic cognitive load (ICL), extraneous cognitive load (ECL) and germane cognitive load (GCL). ICL represents the interaction between the knowledge to be learned and the expertise level of the learner (current knowledge). This load is imposed by the number of elements to be addressed simultaneously in working memory during problem solving. ECL represents all form of load which is not directly devoted to the execution of the current task. This load is not effective for learning and can be reduced by a better instructional design. GCL represents the load resulting from learning processes and is a form of ECL that actively participates in learning. These three types of cognitive load are additive. Their

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 719–721, 2008.

sum may not exceed the working memory capacity without causing a failure of the ongoing task or impairing learning.

3 Using the CLT Framework in ITS

Similarly to most intelligent tutoring systems [12], our ITS architecture includes four main modules: expert agent, interface agent, learner agent and pedagogical agent. Our learner agent extends the standard capabilities by modeling the mental effort (cognitive load) of the learner during problem solving. Using a working memory simulator, the learner agent provides cognitive load patterns representing the load composition (intrinsic, extraneous and germane) resulting from each resolution step. The originality of our pedagogical agent is its ability to take into account dynamically the cognitive load imposed by the various resolution steps of a learning task in order to implement a guided learning strategy.

Several researches in educational sciences [5,6] argue that guided learning is superior to other forms of learning that allow more freedom to the student (e.g. discovery learning [7]). Our pedagogical agent implements a form of guided learning based on the cognitive load theory (CLT) which aims to guide the learner during problem solving according to the cognitive load imposed by each resolution step. The role of the pedagogical agent is to help the student choosing resolution steps that maximise learning. To do so, our pedagogical agent chooses the optimal step sequence regarding learning efficiency via a CLT-based guidance heuristic.

3.1 The CLT-Based Guidance Heuristic

The first stage of the guidance heuristic is to quote the possible next actions with a cognitive load pattern provided by the learner agent. At each resolution step, or following a help request by the student, the pedagogical agent suggests a resolution step offering an optimal cognitive load pattern. The latter is defined as an equilibrate amount of germane cognitive load (GCL) and intrinsic cognitive load (ICL) [10]. Low ICL indicates a step containing no learning opportunities (knowledge to be automated) [9]. High ICL and GCL indicate that the interconnected handled knowledge exceeds the learner working memory capacity [10]. During an instructional session, cognitive patterns for the same resolution steps will change depending on the student expertise progress. More precisely, the intrinsic cognitive load component will decrease in function of the progressive automation of knowledge due to their utilisation during learning tasks. The pedagogical agent will then suggest to the learner a resolution step whose component knowledge is not automated.

The learner will progress from a novice level (low knowledge automation) to an expert level (high knowledge automation). Because the student is guided in a progressive manner which is consistent with his/her expertise level and his/her cognitive limitations, the number of resolution steps will be greatly reduced. This results in an optimal learning strategy in terms of instructional time and mental effort.

4 Conclusion and Further Work

We have proposed a model for dynamic pedagogical tailoring within intelligent tutoring systems. The model uses the cognitive load theory framework and a working memory simulator. An in-depth validation of our cognitive load pattern estimation mechanism is actually in progress.

References

1. Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive Tutors: Lessons learned. Learning Sciences 4(2), 167–207 (1995)
2. Clark, C., Nguyen, F., Sweller, J.: Efficiency in learning: evidence-based guidelines to manage cognitive load. Pfeiffer, San Francisco (2006)
3. Crowley, R.S., Medvedeva, O.: An intelligent tutoring system for visual classification problem solving. Artificial Intelligence in Medicine 36(1), 85–117 (2006)
4. Kalyuga, S., Ayres, P., Chandler, P., Sweller, J.: The expertise reversal effect. Educational Psychologist 38(1), 23–31 (2003)
5. Kirschner, P.A., Sweller, J., Clark, R.E.: Why Minimal Guidance During Instruction Does Not Work: An Analysis of the Failure of Constructivist, Discovery, Problem-Based, Experiential, and Inquiry-Based Teaching. Educational Psychologist 41(2), 75–86 (2006)
6. Klahr, D., Nigam, M.: The equivalence of learning paths in early science instruction: Effects of direct instruction and discovery learning. Psychological Science 15(10), 661–667 (2004)
7. Shulman, L.S., Keisler, E.R.: Learning by discovery. Rand McNally, Chicago (1966)
8. Sweller, J., van MerrienBoer, J.J.G., Paas, F.G.W.C.: Cognitive architecture and instructional design. Educational Psychology Review 10(3), 251–296 (1998)
9. van Merrienboer, J.J.G., Sweller, J.: Cognitive load theory and complex learning: recent developments and future directions. Educational Psychology Review 17(2), 147–177 (2005)
10. van Merrienboer, J.J.G., Kester, S., Paas, F.: Teaching complex rather than simple tasks: balancing intrinsic and germane load to enhance transfer of learning. Applied Cognitive Psychology 20(3), 343–352 (2006)
11. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weistein, A., Wintersgill, M.: The Andes Physics Tutoring System: Lessons Learned. International Journal of Artificial Intelligence in Education 15(3), 147–204 (2005)
12. Wenger, E.: Artificial intelligence and tutoring systems: computational and cognitive approaches to the communication of knowledge. Morgan Kaufmann Publishers, Los Altos (1987)

Investigating Learner Trust in Open Learner Models Using a 'Wizard of Oz' Approach

Alice Kerly, Norasnita Ahmad, and Susan Bull

Electronic, Electrical and Computer Engineering
University of Birmingham, Edgbaston, Birmingham, B15 2TT, UK
{alk584,nxa707,s.bull}@bham.ac.uk

Abstract. Open learner models (OLM) are learner models which are accessible to the learner, allowing them to view, and sometimes modify, their model. This openness may raise questions of learner trust in their learner model: if users do not agree with, or trust the information they can see about themselves, their trust in the interaction will likely be reduced. Using a Wizard-of-Oz approach, we consider learner trust and possibilities for developing trust in OLMs.

1 Introduction

Open learner models (OLM) externalise the learner model contents to the user. Thus OLMs assist learners in tracking their knowledge, and promote independent learning by offering information about their knowledge that the learner would not usually see (e.g. a breakdown of concept understanding or descriptions of misconceptions held) which may allow learners to identify areas to target their study. Opening the model to the learner raises issues of learner trust in their learner model, and hence trust in the system. Lack of trust in the inferences of a learner model may discourage system use, and is therefore an important issue to address.

Gaining students' trust in their learner models has been explored using a multi-agent system with agents cooperating on behalf of learners in a collaborative context [1]. In psychology, trust may relate to personal traits that deal with belief and expectation or feelings [2]. In decision aid systems trust may be defined as the extent to which a user is confident in, and willing to act on the basis of, the recommendations of the system [3]. These definitions may also apply to OLM.

Learners may have more or less control over their learner model contents [4], although there may be risks when control is given to learners [4], [5]. Research suggests students may be uncomfortable with directly editing the model, and prefer an OLM where they have less direct control, such as one they can negotiate [6]. Negotiated OLMs facilitate maintenance of the model through collaborative student-system negotiation of the represented beliefs. The fact that students are willing to give some control to the system suggests that they may trust an OLM or, at least, may have greater confidence in the system assessing their knowledge than in their self-assessments. Similarly, allowing learners some control over the model (such as in negotiated OLMs) may help to increase learners' trust as they are able to influence the model contents if they disagree with it. Given the user preferences for some level of system

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 722–724, 2008.

control, we adopt a negotiated OLM approach to explore trust issues. We present a Wizard-of-Oz study (where part of the system behaviour is simulated by a human 'wizard' – the experimenter) of negotiating the learner model, to identify some of the trust issues in OLM systems, as perceived by learners.

2 Investigating Student Trust in Open Learner Models

The participants were 40 students in the Electronic, Electrical and Computer Engineering Department at the University of Birmingham, UK. Participants interacted with Flexi-OLM [6], modified to provide (by the 'wizard') a simulated chatbot for negotiation of the learner model [7], for a minimum of 20 minutes (mean 28.79 minutes). Participants completed a questionnaire with a five-point Likert scale, and further free response elements, designed to investigate issues relating to trust in OLM.

Table 1. Summary of questionnaire responses

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
I was frequently convinced by the Chatbot's arguments	4	19	14	2	1
The negotiation changed my view of my understanding	7	17	11	5	0
I always challenged the Chatbot when I disagreed (or I would)*	15	15	6	2	0
I was happy to accept the Chatbot's opinions when I disagreed	2	18	12	6	2
I liked the Chatbot when it disagreed with me	4	21	10	3	2
I liked the Chatbot when it agreed with me *	7	22	10	0	0

* 38 responses.

As shown in Table 1, 23 of the 40 users agreed they were frequently convinced by the chatbot's (wizard's) arguments in negotiation; 14 remained neutral. 34 said the negotiation changed their view of their understanding. These are key issues for negotiated learner modelling as if learner and system are to be equal in the negotiation then each must be prepared to consider the other's arguments. The level of trust in the system will influence the user in whether they follow the system's advice. However, different OLM designs suit different users, and we would not expect any single system to suit (or engender trust from) all users.

A common theme in responses was the ability of the system to provide reasoning, e.g. "It gives me objective reasoning", "I could see why it was disagreeing", and "when it disagreed it was justified". If transparency of the OLM can help to support trust, then these comments are consistent with the dimension of interpretive transparency in [5] and suggest the building of some level of trust in this system.

30 users agreed they would always challenge the system if they disagreed with it. Users will gain most benefit from a negotiated OLM if they are willing to initiate discussion where their own and the system's assessments of their knowledge differ. A

user must feel that challenging the system can be effectual or they may not attempt it. This suggests that the users who would challenge the system believe their inputs will be considered, and trust their ability to affect the system's behaviour.

Mayer et al.'s definition [2] does not imply reciprocity of trust; it does not require both parties be vulnerable to the other. However, if the user may influence the system, then the system's willingness to vulnerability may develop users' trust in the system.

It might be expected that users will like the chatbot (wizard) when it agrees with them; it is perhaps surprising that 28 users agreed they liked the chatbot when it disagreed with them. Users appeared to demonstrate some level of trust in the system's beliefs, stating "it explained its reasoning", and "seemed to understand exact misconception and how to get rid of this". Without trusting the system's assessments and arguments of the system it is unlikely that users would claim to like the chatbot.

While it is difficult to measure trust directly, user comments about their interaction suggest trust in the OLM. Given these findings using a Wizard-of-Oz simulation, there are now further issues to consider relating to trust in open learner modelling. It will be interesting to investigate how trust develops in an OLM over time, in a real learning setting. It is also likely that different users have different demands and expectations of what the OLM will offer them, and so it may be that different facilities are required by different users to enhance trust in the learner model.

3 Summary

This paper explored some of the trust issues in learner modelling, including trust in the relationship with the system, in the OLM evidence, in the user's ability to influence the system, and in the purpose of negotiated learner modelling. The users appeared to demonstrate trust in all of these areas. Further investigation will consider issues affecting trust in wider OLM contexts, and will seek to develop strategies to support the development and enhancement of user trust in open learner modelling.

References

1. Vassileva, J., Greer, J., McCalla: Openness and Disclosure in Multi-Agent Learner Models. In: Morales, R., Pain, H., Bull, S., Kay, J. (eds.) Proceedings of the Workshop on Open, Interactive and Other Overt Approaches to Learner Modelling, AIED 1999 (1999)
2. Mayer, R.C., Davis, J.H., Schoorman, F.D.: An integrative model of organizational trust. The Academy of Management Review 20, 709–734 (1995)
3. Madsen, M., Gregor, S.: Measuring human-computer trust. In: Gable, G., Viatle, M. (eds.) 11th Australasian Conference on Information Systems, p. 53 (2000)
4. Kay, J.: Learner Control. User Modeling and User-Adapted Interaction 11, 111–127 (2001)
5. Tanimoto, S.: Dimensions of Transparency in Open Learner Models. In: 12th International Conference on Artificial Intelligence in Education 2005, pp. 100–106 (2005)
6. Mabbott, A., Bull, S.: Student Preferences for Editing, Persuading and Negotiating the Open Learner Model. In: Ikeda, M., Ashley, K., Chan, T.W. (eds.) ITS: 8th International Conference, pp. 481–490. Springer, Heidelberg (2006)
7. Kerly, A., Hall, P., Bull, S.: Bringing Chatbots into Education: Towards Natural Language Negotiation of Open Learner Models. Knowledge-Based Systems 20, 177–185 (2007)

Personalized Learning Path Delivery: Models and Example of Application

Hend Madhour and Maia Wentland Forte

Busines School of Lausanne, 1015 Lausanne, Switzerland
hend.madhour@unil.ch, mwf@unil.ch
http://www.hec.unil.ch/

Abstract. In this article we present the Lausanne Model: a learning object based reference model that: (i) considers learning issues such as granularity level, description formalism,(ii) organizes learning objects in a network where links are explicated, (iii) enhances user mobility from one environment to another and (iv) considers both individual and social adaptation.

1 Introduction

In order to deliver personalized learning paths composed with adequate sets of inter- connected learning resources extracted from Learning Object Repositories [1], we tried to apply some adaptive hypermedia reference models [2] [3].

Because, none of these models addresses specific learning issues, we felt the need to come up with a learning object based reference model.

In this paper we depict our Lausanne Reference Model paying special attention first to the Domain Model and in particular to the granularity issue. We then concentrate on the User Model focusing on the information that should be retained about the learner to provide the best-possible personalized learning path. To conclude, we describe the adaptation algorithm based on the user model and other users' experience.

2 Domain Model: How to Model What?

In the context of a learning environment, the Domain Model focuses on pedagogical material content and aims at describing it by representing its entities and their relationships in a standardized manner. Learning issues raised in this model are: granularity degree and description formalism. If we consider a level of granularity to be adequate when the element is small enough to allow for flexible and integrative reuse and big enough to make sense by itself" then a learning object, as defined here above, could be considered as a potential good candidate. Two methods are available to describe educational resources: indexation and annotation. Both provides useful but different information about the object. In the following example, we show how to combine each of those methods.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 725–727, 2008.

Example. Let us consider the context of segmentation (an annotation formalism) [4] in which it can be claimed that the highest level of granularity of a document is the document itself, while the smallest level of granularity is any of its identified presentation chain seen as a learning object per se. We have adapted MLR (Metadata for Learning Resources) (an indexation formalism) [5] to describe the relationships between learning objects capitalizing on the semantic information obtained when segmenting the document. We distinguish three types of metadata:

- Navigational metadata that aims at personalizing the navigation in a given domain taking into account the environmental constraints (MLR: Contextualization), the security issues (MLR:Access) and the relationships between presentation chains (MLR:Description:Relation). Navigation is done through a graph of resource anchors, the navigational metadata being associated to the anchors.
- The conceptual metadata that comprises all the attributes of a concept other than those belonging to navigational metadata. It is stored, when it exists, in the Description category (MLR:Description :Description).
- Descriptive metadata that gives a global description of the resource. It is associated directly with the resources in one and same file which is uploaded in a Learning Object Repository (LOR) [1].

3 User Model: Adapt to What?

Bearing in mind that we aim at being able to deliver a suitable personalized path to a learner, we need to gather as much information as possible about the learner to derive and determine a number of specific useful characteristics: a User Model. We have studied two standards of a user model: IEEE Public And Private Information (PAPI) [6] and IMS Learner Information Package (LIP) [7].

Much more detailed than PAPI, LIP nevertheless entitles the learner to modify the attributes of his user model (like PAPI). Because we believe that the responsibility of the learning process should not be entirely delegated to the learner, we think that this possibility should be shared and restricted. We propose therefore to split the learner's attributes into four categories depending on the modification rights: machine driven, learner driven, system driven and tutor driven. For any specific learner to be able to easily retrieve adequate learning objects, and therefore for the system to provide a personalized learning path, we propose to map XUM with some MLR attributes (as a metadata example).

4 Adaptation Model: How to Adapt?

We are now in a position to describe how we use the information stored in the XUM to retrieve suitable learning objects: we first get a set of possible candidates from which we choose the elements to be ultimately retained to be included in the learning path to be proposed to the learner. Based on an active user

model (individual adaptation) and on other users' profiles (social adaptation), the Adaptation model describes how the adaptation is performed.

The individual adaptation process filters items based on the user's needs as they are mainly recorded in the fields of the machine driven modification category. It is used when calculating the next unit to be visited (Choice function). The Social adaptation process aims at providing a personalized learning path based on the experience of other learners provided they share a similar knowledge level (proficiencies) and the same interests. Our algorithm includes both processes and aims at building a suitable learning path for a specific learner. To generate this personalized learning path, we propose to use the Ant Colony Optimization algorithm (ACO) [8].

5 Conclusion

In this article, we have described the Lausanne Reference Model, designed for learning object systems. The Domain Model is described as a set of indexed learning objects. The User Model, baptized XUM, is based on the user profile two main standards, the IEEE/PAPI and IMS/LIP. The Adaptation Model is based on the ACO algorithm which has the advantage of benefiting from the social dimension and provides a suitable learning path. Future work will consist in integrating the Lausanne Model based system in a learning environment. This system, applied in a lifelong learning process, could bring a real added-value if coupled to a knowledge portfolio.

References

1. Rehak, D., Mason, R.: Keeping the Learning in Learning Objects. Carnegie: Learning Systems Architecture Lab, Mellon University (2003), http://www.lsal.cmu.edu/lsal/expertise/papers/
2. De Bra, P., Houben, G.J., Wu, H.: AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. In: Proc. ACM Hypertext 1999, Darmstadt, pp. 147–156 (1999)
3. Koch, N., Wirsing, M.: The Munich reference model for adaptive hypermedia applications. In: Proc. 2nd International conference on Adaptive Hypermedia and Adaptive Web-based systems, Malaga, Spain, pp. 213–222 (2002)
4. Wentland, M.: Modelisation d'un domaine de connaissance et orientation conceptuelle dans un hypertexte pedagogique, PhD Thesis from University of Lausanne (1994)
5. MLR Draft for a standard (2005), http://mdlet.jtc1sc36.org/doc/SC36WG4N0145.pdf
6. IEEE Public And Private Information (PAPI) (2002), jtc1sc36.org/doc/36N0186.pdf
7. IMS Learner Information Package (LIP) (2001), www.imsglobal.org/profiles/index.cfm
8. Dorigo, M., Di Caro, G.: Ant algorithms for discrete optimization. In: Artificial Life, pp. 137–172. MIT Press, Cambridge (1999)

Semi Automatic Generation of Didactic Resources from Existing Documents

Mikel Larrañaga, Jon A. Elorriaga, and Ana Arruarte

Department of Languages and Information Systems,
University of the Basque Country. P.K. 649
20080 Donostia, Basque Country
{mikel.larranaga,jon.elorriaga,a.arruarte}@ehu.es

Abstract. Tools for generating learning material in automatic or semiautomatic way are needed in order to lighten the development of Computer Supported Learning Systems. This paper describes an approach to semi automatic generation of didactic resources from electronic documents using ontologies and Natural Language Processing techniques.

Keywords: Computer Supported Learning Systems, Semi Automatic Domain Acquisition, Didactic Resources, Ontologies

1 Introduction

Content authoring is a time and effort-consuming task so the idea of knowledge reusing existing learning material in Computer Supported Learning Systems should be promoted. Traditionally, written documents have been used as a resource to transmit knowledge from one generation to the next one. Textbooks and books in general collect years and years of knowledge about many different domains. To be able to reuse that knowledge in computers would be really a big success.

The work here presented is part of a project which aims to semi automatically develop the Domain Module for Computer Supported Learning Systems from electronic documents by using a combination of different NLP tools, ontologies and heuristics reasoning [1]. This paper deals with the identification of the Didactic Resources (DRs) related to the topics of a learning domain. After describing the identification and creation of DRs from documents, some conclusions and future work are pointed out.

2 Semi Automatic Domain Module Generation. DR Generation

In the approach here presented, the process of semi automatically acquiring the Domain Module from electronic documents is divided into three phases:

- ***Domain Module Structure Acquisition*:** the domain ontology containing the domain topics and the pedagogical relationships among topics is gathered.
- ***Generation of Didactic Material*:** ontology-driven analysis of the whole document in which fragments related to the domain topics are detected and categorized.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 728–730, 2008.

- ***Domain Maintenance*:** this maintenance implies analysing new documents repeating the first two phases of the process in order to keep the domain module up to date. Ontology integration techniques will be needed to tackle this phase.

This paper focuses on the Generation of the Didactic Material. A domain ontology, which has been gathered in the Domain Module Structure Acquisition phase, and a didactic ontology [2, 3] are used to look for the fragments of the document that correspond to didactic resources. First, a linguistic analysis is performed on the document obtaining the *part-of-speech* information of the document. This information, the original document and the domain ontology are used in the next step, DR identification, in order to find fragments of the document that correspond to DRs. Document authors use similar structures to provide topic definitions, examples, exercises and so on. A grammar that defines these patterns has been developed. This grammar facilitates the identification of the fragments of text in the documents that contain simple DRs. The identified DRs are usually quite simple, so consecutive DRs are combined in order to get more accurate DRs. Two DRs are combined if they are considered similar. The similarity of two DR is determined by two aspects: the similarity of their content, i.e. the domain topics they reference, and the resemblance of their DR types. The methods that determine these similarities return a value in the [0, 1] range. Two DRs are considered similar if the obtained topic similarity and the DR type similarity are beyond the corresponding threshold values. The DR composition process is repeated until no changes are done in the input DR set.

In order to measure the similarity of the contents of two DRs, not only the referred topics but the semantic relationships between them are considered. The content similarity measuring method is based on the work of Hughes and Ramage [4], which computes the semantic relationships among topics in order to get the stationary probabilistic distribution for every topic. The relatedness of the two documents is computed as the similarity of their stationary distributions. In this case, the vectors containing the stationary distributions of the two DRs are computed using the formula (1) to get their similarity where *d* and *d'* are the vectors used to model each text. In this work, each vector element contains how many times the corresponding domain topic is referenced in the DR.

$$\cos(d, d') = \frac{d \bullet d'}{|d||d'|} \tag{1}$$

The method to measure the DR type similarity uses the same approach but using a Didactic Ontology [2, 3] to represent the relationships among the types of DRs and compute their similarity. In this case, *d* and *d'* contain how many times the corresponding DR type appears in the DR.

When developing the application, all the found patterns were considered and included in the grammar in order to get a 100% recall, i.e. percentage of real DRs detected, even if that may affect the precision (percentage of correctly identified DRs). It might be easier to discard invalid DRs than building undetected ones. The obtained results have been compared to the DRs identified by human instructors. 92.86% of the gathered DRs were correct. The referred topics were correctly identified for 92.22% of the obtained DRs. Just 7.59% of the DRs had to be enhanced while just 4.12% of the DRs had to be split to meet human instructors' classification.

3 Conclusions and Future Work

In this paper a domain independent method for semi automatically generating didactic resources (DRs) from documents has been described. The method relies on the use of ontologies and NLP techniques. A grammar defining the patterns or syntactic structures that may identify DRs has been developed after the analysis of several textbooks in Basque language. This grammar is applied on electronic documents and the obtained atomic DRs are combined in order to get more accurate ones, i.e, closer to the DRs the human instructors identify. Several similarity measure methods that determine which DRs must be composed have been tested so as to get results near to human instructors' outcomes.

Future work includes the development of a graphical user friendly application that will allow the supervision of the results to any human instructor and the integration of the work here presented in Elkar-DOM [5] in order to facilitate the whole process of generation of DRs.

Acknowedgments. This work is supported by the Univ. of the Basque Country (UE06/19), the MEC (TIN2006-14968-C02-01) and the Gipuzkoa Council in an EU Program. Also, we want to thank the collaboration of the Gipuzkoako Ikastolen Elkartea.

References

1. Larrañaga, M.: Enhancing ITS building process with semi-automatic domain acquisition using ontologies and NLP techniques. In: Young Researches track of the Intelligent Tutoring Systems (ITS 2002). Biarritz, France, pp. 5–8 (2002)
2. Meder, N.: Didaktische ontologien. In: Globalisierung und Wissensorganisation: Neue Aspekte für Wissen, Wissenschaft und Informationssysteme, pp. 401–416 (2000)
3. Leidig, T.: L3–Towards an Open Learning Environment. ACM Journal of Educational Resources in Computing 1(1), 5–11 (2001)
4. Hughes, T., Ramage, D.: Lexical Semantic Relatedness With Random Graph Walks. In: Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, pp. 581–589 (2007)
5. Larrañaga, M., Niebla, I., Rueda, U., Elorriaga, J.A., Arruarte, A.: Towards Collaborative Domain Module Authoring. In: IEEE International Conference on Advanced Learning Technologies 2007 (ICALT 2007), pp. 814–818 (2007)

An Evaluation of Intelligent Reading Tutors

Sowmya Ramachandran[1] and Robert Atkinson[2]

[1] Stottler Henke Associates, Inc
San Mateo CA
sowmya@stottlerhenke.com
[2] Arizona State University
Mesa, AZ
Robert.Atkinson@asu.edu

Abstract. Reading comprehension is a very important life skill, yet millions of Americans are functionally illiterate. Technology can help, but most computer-based training programs for reading skills fall short in their ability to provide self-paced, individualized learning. The Navy funded three innovative intelligent tutors that were developed based on well-recognized cognitive models of reading and learning. In addition, these tutors provide tools for customization of content. This paper reports a large-scale study of these tutors that demonstrates their teaching effectiveness when used individually and in combination. Of particular note is an ordering of the tutors that led to a skill gain of 1.1 grade levels.

1 The Problem of Literacy and Its Significance

While a high school diploma is required for enlisting in the Navy, a significant percentage of the enlisted crew have weak reading skills. Twenty-five percent of the Navy's enlisted population, for example, scores below the eighth grade level in reading, writing, and arithmetic. Beyond the Navy, literacy is one of the most fundamental requirements for succeeding in today's world. Yet, fifty million Americans are not functionally literate. A few computer-based training tools that have been developed for reading [www.ncslearn.com, www.plato.com] are not suitable for the Navy. First, they require instructor intervention for assessment and lesson assignment. Second, they have fixed content that cannot be tailored to an organization's environment and needs. Finally, many training programs for reading target skills at the phonetic level. To enlist in the Navy requires a basic ability to read. Navy personnel have mastered the skill of decoding, at least at its most basic level. To address these problems, the Navy funded the development of three intelligent tutors with customizable content for teaching reading comprehension skills. This paper briefly describes the tutors, and focuses on the results of a large-scale evaluation of their teaching effectiveness.

2 The Three Reading Tutors

ReadOn aims to improve the reading skills of students through targeted assessment, reading practice, and remediation. Using the reading skills described in [2] as the domain

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 731–733, 2008.

model, ReadOn provides coached reading practice by prompting learners to read passages and answer multiple-choice questions on the passage that cover a wide range of skills. The student model is a Bayesian Network overlaid on the domain model and is used to select optimal challenge level, present targeted remediation, for dynamic selection of questions with exercises. The domain model, the student model and the tutoring approach are described in detail in [4;5].

GradAtions [http://www.i-a-i.com/view.asp?aid=28] and STAR ([1] are the other two reading tutors developed with funding from the Navy. The three systems have complementary approaches. Whereas ReadOn is geared towards practice and assessment of a broad range of skills, GradAtions focuses mainly on summarization, a high-level cognitive skill that contributes significantly to reading comprehension. GradAtions provides several scaffolding mechanisms that help learners identify the key information in a given passage and generate a good summary. It includes automated assessment of the summaries provided by the learners. STAR, on the other hand, implements the reciprocal teaching method [3] where a teacher models critical thinking skills through active questioning and the learners attempt to follow the model with active questioning of their own. STAR implements this policy where the tutor models questions and learners respond by specifying questions in turn.

3 Evaluation

A large-scale evaluation of the three reading tutors was conducted at Arizona State University where the objective was to study the effectiveness of each of the three tutors individually and in combination. Of the 368 participants initially registered, 159 completed the entire study. The participants were assigned in approximately equal proportion to one of seven conditions. The first six conditions represent the six ways in which the three ONR-funded tutoring programs could be ordered. The seventh condition was *Lifetime Library's Reading Program* which is a popular computer-based training software for reading that is currently used by the Army. Each testing procedure for one of the experimental conditions and the control condition is shown below[1] (Here, A = *Read on!*, B = *STAR*, C = *Gradations*, and D = *Lifetime Library's Reading Program)*. The other five experimental conditions tested the other possible orderings of the three tutors.

Table 1.

	(Pretest) Assessment 1	Treat. Period 1	Assessment 2	Treat. Period 2	Assessment 3	Treat. Period 3	(Posttest) Assessment 4
Cond. 6	ACCUPLACER + TABE + ASVAB	C	ACCUPLACER + affective	B	ACCUPLACER + affective	A	ACCUPLACER + ASVAB + TABE + affective
Cond. 7	ACCUPLACER + TABE + ASVAB	D	ACCUPLACER + affective	D	ACCUPLACER + affective	D	ACCUPLACER + ASVAB + TABE + affective

[1] ACCUPLACER, TABE and ASVAB are standardized assessment instruments for reading comprehension skills.

The mean instructional time for the six experimental conditions—excluding the Lifetime Learning control condition—was 35.64 hours (SD = 4.82). The group with Lifetime Learning got about 40 hours of instruction.

Participants that initially worked exclusively on GradAtions and ReadOn achieved significantly higher scores on the ACCUPLACER after working with these tutors for approximately 15 and 5 hours, respectively ($p=<.001$ for GradAtions, and $p<0.05$ for ReadOn). The experiment also revealed that four combinations of the three reading programs produced significantly higher scores on the final ACCUPLACER test than the first. Of these, however, the GradAtions → STAR→ ReadOn combination produced the largest learning gains. The participants that worked with this sequence had a gain of 15.5 points, which corresponds with a reading comprehension grade level gain of 1.1 ($p < 0.01$). Thus, the most optimal way of ordering the three tutors is Gradations first, STAR second, and ReadOn third. The affective questionnaires indicate that learners rated ReadOn significantly higher than the other tutors in terms of ease of use and engagement.

In conclusion, we described an evaluation that shows the effectiveness of three intelligent tutoring systems for reading comprehension both alone and in combination. Results show that these tutors lead to significant gains in reading comprehension skills after only about thirty five hours of instruction. The tutors are undergoing further evaluations in the Navy.

References

1. Anastasi, D., Entin, E.B., Miller, D.L.: Skills Training in Advanced Reading (STAR): A Computer-Based Tutor to Improve Adult Reading Comprehension. AP-R- 1326. Aptima, Inc., Woburn (2006)
2. Just, M.A., Carpenter, P.A.: The Psychology of Reading and Comprehension. Allyn and Bacon, Inc. (1987)
3. Palinscar, A.S., Brown, A.L.: Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. Cognition and Instruction 1, 117–175 (1984)
4. Ramachandran, S., Stottler, R.: An Intelligent Tutoring System for Adult Literacy Enhancement. In: Proceedings of the fifth International Conference on Intelligent Tutoring Systems, Montreal (2000)
5. Ramachandran, S., Stottler, R.: An Intelligent Tutoring System for Adult Literacy Enhancement. In: Proceedings of the Industry/Interservice, Training, Simulation & Education Conference (I/ITSEC 2003) (2003)

An Intelligent Web-Based Learning System for Group Collaboration Using Contracts

Henri Eberspächer[1,2] and Michelle Joab[1]

[1] LIRMM - Université Montpellier II/CNRS 161 rue Ada 34392 Montpellier, France
[2] PUCPR – Pontifical Catholic University of Parana, Curitiba – PR, Brazil
Henri.Eberspacher@pucpr.br, Michelle.Joab@lirmm.fr

Abstract. In this paper we introduce an Intelligent Web-Based Learning System Virtus which supports role-based collaboration using a group contract model (a charter) based on roles and rules. The originality of our work consists, on the one hand, in proposing a declarative language to express contracts using declarative rules, commitments and role responsibilities and, on the other hand, in automatically executing the contract thanks to a knowledge-based system.

Keywords: Intelligent Web-based learning, Role-based collaboration, Contract.

1 Introduction

Web-based platforms devoted to e-learning usually provide a virtual learning environment which supports learning activities. These platforms are used within a variety of methods ranging from classroom education to partially or completely on-line distance education and from individual to collaborative work. They offer a wide range of tools to support one-to-one or group learning: communication tools, course management tools, content management tools and collaborative work tools.

In this paper we present a software component named VirtusCharte as a part of an e-learning platform that will support group work using a contract by automating the execution of the rules of the contract. Once the contract is defined for a group, it is translated into an executable rule language and periodically, the Virtus platform [1], resulting from our work, extracts information from the learning environment and activates a knowledge-based system for each active group in the virtual learning environment in order to ensure that the conditions described in the contract are followed.

2 Virtus Platform

The Virtus platform consists of two modules: VirtusWeb which manages the services of the learning environment (like usual Learning Management Systems) and VirtusCharte which implements an Intelligent Support System to support groups using the rules of the contract during collaborative work. This means that the Virtus platform supplies a regulation and monitoring mechanism [2] for group activities based on concepts of contract and role [3]. This mechanism is fully configurable through the rules expressed in the proposed contract language using freely defined roles. VirtusCharte is based primarily on an

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 734–736, 2008.

intelligent tool that will automatically support collaborative work using VirtusWeb, the LMS part of the Virtus platform. So, VirtusCharte contributes to the field of Intelligent Web-based learning.

Our proposal is based on the relation between rules and roles. *Rules* mean a guideline, a formula that indicates what needs to be done within a certain context. *Roles* mean that each person is identified with a role, so that he/she can quickly identify his/her participation, prerogatives and commitments. We use the term "role" in accordance with [4]: a role is a set of requirements that define how a member should behave in a group. The rules are based on the concepts of the learning environment: resources, activities, events and roles.

The contract declarative language is able to describe how to monitor the status of an activity and the actions carried out on an activity (such as postponing an activity deadline). Using the rules, the author of the contract expresses, for example, how to react in cases where there is a procedure that is non-compliant with the behavior expected for the role. The assigned user's roles indicate the responsibilities and expected actions of each group member. The rules could question the work carried out by other roles or those that are performed outside the specified time constraints.

Using the contract, the users will benefit from an automatic group management support mechanism. From the point of view of a human user, VirtusCharte may act as a virtual user on VirtusWeb working environment and could also recall the users' commitments by sending notifications. These actions and/or notifications are a consequence of the group contract execution.

A learning environment running on Virtus platform is made by a set of services in a given context. The context is structured on three levels: *(i)* the individual (private user space), *(ii)* the group (all group members space), and *(iii)* the community (whole community space). Everyone has his own private space and can participate in public spaces structured in groups within communities. An object always exists in a context and it is always handled in functional spaces categorized in services. Currently, Virtus provides three types of services: *(i)* activity management, *(ii)* event management, and *(iii)* resource management. All functionalities take into account the three contexts (a user could participate in several groups and communities playing different roles).

As a result, we have, on the one hand, VirtusWeb an environment where activities, events and resources are created, viewed and edited, and on the other hand, VirtusCharte, a system that interacts with these objects, applying the terms of the contract. VirtusCharte must be able to consult, edit and modify them using the contract in a specific context. By separating the two software modules of the platform, we aim to run the contract mechanism independently of a particular platform. This means that VirtusCharte could, with some adaptations, be plugged in another LMS. This will be effectively applied in the system's design and in the prototype implementation.

For the development of VirtusWeb, we used free software that is commonly used in open-source LMS systems, the LAMP platform: Linux, Apache, MySQL and PHP.VirtusCharte has been implemented in Java and uses the same classes as VirtusWeb with regard to the model of the system. We used JESS [5], a Java Expert System Shell to build the knowledge-based system used in VirtusCharte.

For each execution of the knowledge-based system in a private, community or group context, VirtusCharte built the initial facts recovering them with a selective loading algorithm importing them from the VirtusWeb database. Once it has loaded

the required information, it creates its own lists of objects and deals with this information to state the necessary facts to start the expert system.

At the end of the knowledge-based system inference cycle, VirtusCharte consults the working memory (which was changed by the rules) to find the facts which indicate actions to be undertaken on the Virtus platform and notifications to be sent. The actions undertaken by the methods of VirtusCharte are performed using web services in a session where VirtusCharte becomes a user of VirtusWeb.

3 Conclusion and Further Work

The originality of our work consists in proposing a declarative language to express contracts using declarative rules, commitments and role responsibilities and in automatically executing the contract using a tailored knowledge-based system.

In the medium term, the goal is to achieve an adaptive environment in the sense that VirtusCharte will run a contract originally adopted by a group and monitor how the effective behavior of groups complies with the terms of the contract when performing a learning activity. If a discrepancy is found between the contract prescribed and the actual conduct of the group, VirtusCharte could propose changes to the contract in order to be more in line with the group behavior [6]. The contract is then renegotiated and revised by the group members.

References

1. Eberspächer, H., Joab, M.: Virtus: group support using role-based collaboration. In: ICALT 2006 - The 6th IEEE International Conference on Advanced Learning Technologies, Kerkrade, The Netherlands, pp. 1079–1081 (2006)
2. Mezura-Godoy, C., Talbot, S.: Towards Social Regulation in Computer-Supported Collaborative Work. In: CRIWG 2001 - 7th International Workshop on Groupware, Darmstadt, pp. 84–89 (2001)
3. Ferraris, C., Brunier, P., Martel, C.: Constructing collaborative pedagogical situations in classrooms: a scenario and role based approach. In: CSCL 2002 - Computer Support for Collaborative Learning (2002)
4. Zhu, H.: Role Mechanisms in Collaborative Systems. International Journal of Production Research 44(1), 181–193 (2006)
5. Friedman-Hill, E.: JESS in action. Greenwich, Manning (2003)
6. Kildare, R., Williams, R.N., Hartnett, J.: An online tool for learning collaboration and learning while collaborating. In: 8th Australian Conference on Computing Education. ACM International Conference Proceeding Series, vol. 165, pp. 101–108 (2006)

An Adaptive and Customizable Feedback System for Intelligent Interactive Learning Systems

Maite Lopez-Garate, Alberto Lozano-Rodero, and Luis Matey

CEIT and Tecnun, University of Navarra.
Manuel de Lardizabal 15, 20018 San Sebastián, Spain
{mlgarate,alozano,lmatey}@ceit.es

Abstract. This paper describes a proposal to build an intelligent feedback selection system for Intelligent Interactive Learning Systems (IILS). The system is aimed at generating multimodal feedback in real-time as a response to student's actions. We examine both educational and human factors that have influence on the behavior and let the instructor decide the significance of each factor. The instructor will customize the system to refine its behavior in each training session, and while the system decides which the appropriate feedback is, the instructor can focus on other instructional tasks.

Keywords: Feedback system, simulators, adaptive, customizable, training, virtual reality, intelligent interactive learning system.

1 Introduction

Intelligent Tutoring techniques have also been used in well-known Virtual Reality systems like STEVE [1] showing some of the benefits that the combination of technologies can bring. However, as Virtual Reality evolves, increasingly complex systems can be developed and new challenges must be faced to build Intelligent Interactive Learning Systems (IILS), that is, systems that integrate both Intelligent Tutoring and Virtual Reality technologies.

This study is focused on advanced IILS. Specifically, a truck simulator is used as a demonstrator where the students feel like driving a real truck. Meanwhile, the instructor is in charge of monitoring and analyzing all the data to determine the performance of the students at the Instructor Position. The study presented in this paper is aimed at automating the process of giving feedback to the students during the training sessions while, the human instructor is in charge of other instructional tasks.

2 The Feedback System

Our objective is to build an adaptive and customizable feedback system for emulating desired behaviors. We focus our work on the real-time communication with the students in response to their actions. We deal with neither choosing the objectives of the task nor updating the student model. The instructor or the ITS (if it exists) must do this work. However, when deciding how to give feedback to the student, the objectives of the task

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 737–739, 2008.

and a simple student model are necessary to provide every student with feedback adapted to them. We use a diagnostic component based on DETECTive [2] [3].

As regards the design of a customizable feedback system, we mean that the instructors must be allowed to choose and refine the behavior of the system according to their needs, for example letting the students explore with freedom and learning from their mistakes [4].

Two modules cooperate to make decisions in the feedback selection process and a third one is the responsible for the presentation of the feedback:

Module 1: Discard insignificant messages.
Module 2: Decide which feedback is the optimal feedback.

We let the instructor decide the value of the main factor that characterizes the feedback system behavior, the intrusiveness, that is, how and how much the system can interrupt the student.

The system can have different behaviors when choosing which messages from the diagnostic component to discard. The significance of every received message is estimated, and negligible messages are discarded. To estimate the significance of a message we use four parameters. First, time since last feedback is important in order to avoid overloading the student [18]. Then, the seriousness of the mistake, that represents the importance that the instructor assigns to a mistake from an educational point of view. The difficulty of the action is the third parameter involved in the significance estimation. The fourth parameter is the difference of level between the instructional level of the action and the level of the student.

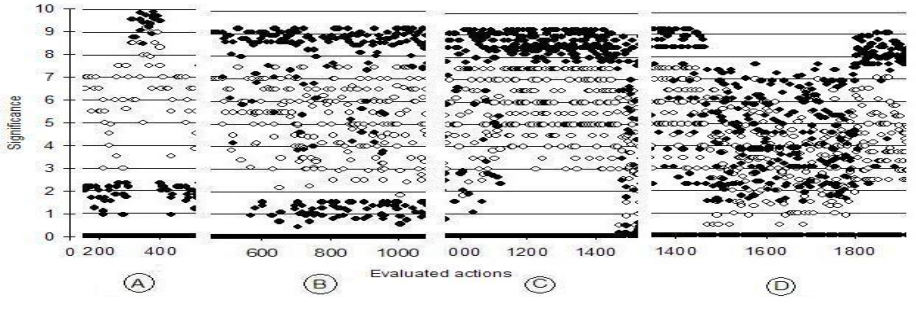


Fig. 1. Significance of errors with equi-weighted (white) and weighted (black) system

Once all the partial significances are estimated, the global significance of the message is estimated, then the instructor chooses weights for the parameters and the weighted average of all the partial results is calculated.

In our experiment the actions are distributed along the time to form different regions. These regions are defined by grouping actions that share a common feature related to the weighted parameters. Region A groups serious mistakes. Region B contains mostly easy actions and actions that the student worked in previous instructional levels (low level actions). Then we see region C where actions have low difficulty. Finally, the actions in the region D belong to low level.

The experiment is aimed at showing how the changes of weight modify the final decisions of the feedback system. Initially, the feedback system is configured with equi-weighted parameters. Next, we assign a high weight to each parameter individually and we compare the decisions made by the feedback system with the results produced by the equi-weighted configuration. In order not to overload the paper, we will only show the relevant parts of each simulation.

In Fig. 1 regions A and C show that significance is relevantly increased where the actions related to the modified parameter are present. In the center of the region D, there are actions with low difficulty, so, giving high weight to the difficulty parameter does not increase the significance. On the other hand, the difficulty of the actions is high in the borders, so the effect is similar to regions A and C. In B we see that the significance is either very high (when last feedback was long ago) or very low.

3 Conclusions and Future Work

We propose a customizable and adaptive feedback system which behaves as the instructor likes. We show how the decision of giving feedback change depending on how important is each parameter for the instructor. The system adapts itself to provide real-time multimodal feedback according to how the student performs every action during the whole learning process.

Our next steps involve researching about the different types of feedback that can be used. Then, we will conduct exhaustive validation experiments with real students to test the effectiveness of the system from the educational point of view.

Acknowledgments. This work has been supported in part by the Spanish Ministry of Education and Science grant TIN2006-14968-C02-01, Torres Quevedo Programme, European Social Fund and the Association of Friends of the University of Navarra.

References

1. Rickel, J.W., Lewis Johnson, W.: Animated Agents for Procedural Training in Virtual Reality: Perception, Cognition, and Motor Control. Applied Artificial Intelligence 13, 343–382 (1999)
2. Ferrero, B., Martín, M., Álvarez, A., Urretavizcaya, M., Fernández-Castro, I.: Authoring and diagnosis of learning activities with the KADDET environment. Universal Computer Science 11, 1530–1542 (2005)
3. Lozano, A., Urretavizcaya, M., Ferrero, B., d Castro, I.F., Ustarroz, A., Matey, L.: Integration of a Generic Diagnostic Tool in Virtual Environments for Procedural Training. In: Conejo, R., Urretavizcaya, M., Pérez-de-la-Cruz, J.-L. (eds.) CAEPIA/TTIA 2003. LNCS (LNAI), vol. 3040. Springer, Heidelberg (2004)
4. Siemon, J., Matey, L., Berasategui, M.I., y Klockmann, D.: A Failure is the Origin of a Success - Or How to Employ Errors for Effective Learning in Vocational Education. In: World Conference on Educational Multimedia, Hypermedia and Telecommunications, Honolulu, Hawai (2003)

Detection of Learning Styles from Learner's Browsing Behavior During E-Learning Activities

Nabila Bousbia[1,2], Jean-Marc Labat[2], and Amar Balla[1]

[1] Institut National d'Informatique (INI), BP 68M,
16270, Oued-Smar, Algiers, Algeria
{n_bousbia,a_balla}@ini.dz
[2] LIP6, Laboratoire de Paris 6, 104, Avenue du Président Kennedy
75016, Paris, France
{nabila.bousbia,jean-marc.labat}@lip6.fr

Abstract. One of the bases of adaptation and learning tracking is the learner's modeling. Research in this field, or more generally in the field of user modeling, was sustained mainly on the detection of features related to the user's knowledge, interests, goals, background, and individual traits [3]. We are interested in this last aspect, in particular the identification of the learning style. In this paper, we propose an approach for the learner's activity perception on an e-learning platform to identify the user's learning styles from observable indicators related to their learning path and interactions.

1 Introduction

Informing the teacher on the way in which the learner studies in distance learning is a fundamental element to organize the tracking in order to specify, at best, how and when the tutor intervenes. It also helps the learning or teaching contents designer to adapt them, to ensure an individualized learning, and gives the learners a reflexive glance on their learning methods so as to acquire a meta-cognitive knowledge. These characteristics, to locate, constitute the learner model. Different ways exist to model it [3]. Wenger distinguishes two levels [6]: a behavioral level, observed-reality organization and, in a more abstract way, an epistemic level which seeks to interpret this behavior and which often resides in a diagnostic function. It is on this level that we can identify the learner's working method, or his/her learning style. In this paper, we propose a methodology to identify learning styles by the interpretation of observable indicators in educational hypermedia systems (EHS).

2 Learning Style and EHS

Several definitions of learning style are proposed in literature. To clarify this concept, Chevrier and al. [4] organize them into three categories, according to whether they refer to: a specific way of behaving, predisposition or preferences related to learning and teaching contexts; information processing; personality characteristics. In this way,

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 740–742, 2008.

various learning styles models were proposed; Coffield and al. [5] count 71 models, of which some were implemented in educational hypermedia systems (WHURLE, CS383, ILASH, etc. [2]). The detection of these styles rests on questionnaires proposed for each model (ILS, FSQ, etc. [4]).

Our interest relates to the detection, in an e-learning framework, of learner's learning styles by the automatic analysis of behaviors through the collection and interpretation of information (called tracks or observables) on the learner's activities. To address this issue, we propose a methodology to identify learning styles by the interpretation of observable indicators. To determine these indicators, we relied, on the one hand, on state of the art studies on digital tracks resulting from learning situations [7], and on the other hand, on the analysis of feedback that teachers want to have on their students. This was done through an investigation carried out with teachers. Thus, we detected the most significant indicators at the different navigation levels we can observe (page, course, platform and session). They were also classified according to the level of interpretation they provide [7]: low level (having no meaning alone), intermediate, and high level (interpretable, inferred from other indicators). The interpretation is mainly based on the semantics of learner's browsing path, relied on studies undertaken for the analysis of Web browsing semantics [1].

To measure the learning style, we classify the studied learning style models proposed in literature by connecting them according to our needs, based on the three elements of the learning style definition: 1) Educational preferences: environment preference, representations and encoding methods, preferred learning time; 2) Learning process: learning strategy, comprehension and progression approach; 3) Cognitive abilities: motivation and concentration capacity. Each layer is defined by two or three attributes we both can and want to observe, each attribute having a value. The values of each layer's attribute are chosen from existing learning style models, by making their definitions closer. They are calculated from high level indicators as shown in the figure 1 for the comprehension attribute of the learning process layer. The proposed

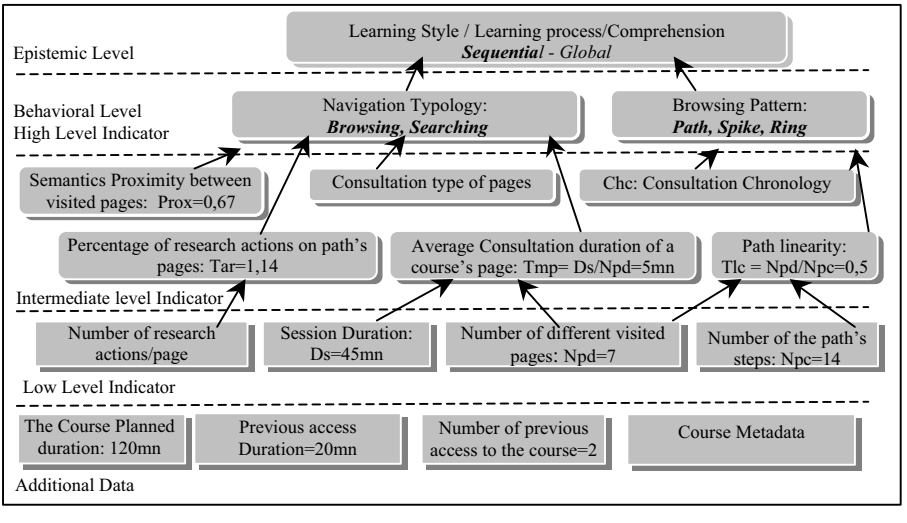


Fig. 1. Example of learning style calculation based on indicators

learning style model intervenes, therefore, on the epistemic level of the learner model. At the behavioral level, we find the already defined high level indicators.

3 Conclusion

In order to track user learning, we seek to understand the way the learner studies in an e-learning environment. Consequently, we developed a learning style model that takes its values from observable indicators, identified after the analysis of the feedback required by teachers. This approach allows the tutor to perceive learners individual characteristics on both epistemic and behavioral levels. It is also useful for adaptation and training individualization goals. The next step of this work is to observe the behavior of a group of learners in a context of distance learning course with no predefined scenario. For validation and relevance, the results from the learning style detection based on indicators will be compared with those of questionnaires.

References

1. Bousbia, N., Labat, J.M.: Perception de l'activité de l'apprenant dans un environnement de formation. In: INRP (eds.) Environnements Informatiques pour l'Apprentissage Humain, pp. 233–238. INRP (2007)
2. Brown, E., Cristea, A., Stewart, C., Brailsford, T.J.: Patterns in Authoring of Adaptive Educational Hypermedia: A Taxonomy of Learning Styles. Journal of Educational Technology and Society 8(3), 77–90 (2005)
3. Brusilovsky, P., Millán, E.: User models for adaptive hypermedia and adaptive educational systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 3–53. Springer, Heidelberg (2007)
4. Chevrier, J., Fortin, G., Théberge, M., Le Blanc, R.: Le style d'apprentissage: une perspective historique. In : Le style d'apprentissage, vol. XXVIII Numéro 1, printemps été 2000 ACELF (2000), http://www.acelf.ca/revue/XXVIII
5. Coffield, F., Moseley, D., Hall, E., Ecclestone, K.: Learning styles and pedagogy in post-16 learning. A systematic and critical review. Learning and Skills Research Centre, London (2004)
6. Croset, M.C.: Vers une modélisation épistémique de l'apprenant. Cas du développement et réduction d'expressions algébriques. In: SIPEMAT, 10-14 juillet 2006, UFPE Recife Brésil (2006)
7. Dimitracopoulou, et al.: State of the Art on Interaction Analysis: Interaction Analysis Indicators, Délivrable de la tâche 26.1 du projet JEIRP-ICALTS, Juillet 2004 Kaleidoscope, 148 pages (2004)

Analyzing Learners' Self-organization in Terms of Co-construction, Co-operation and Co-ordination

Patrice Moguel[1], Pierre Tchounikine[1], and André Tricot[2]

[1] Lium, Université du Maine, Avenue Laennec, 72085 Le Mans cedex 9, France
{Patrice.Moguel,Pierre.Tchounikine}@lium.univ-lemans.fr
[2] LTC – IUFM, 56 av. de l'URSS, 31078 Toulouse cedex, France
Andre.Tricot@toulouse.iufm.fr

1 Context: A Collective Mediated Challenge

We define a *pedagogic collective challenge* as a CSCL learning situation where: (1) the problem is designed to make learners practice some target domain-related and/or meta-cognitive competencies; (2) a group of learners is involved, as a team, in the solving of the problem; (3) the solving requires the learners to join their forces; (4) the problem and the setting are designed to create a positive tension that motivates learners. Such challenges aim at enhancing learners' motivation in involving themselves in the collective solving and, within this process, in knowledge generative interactions such as conflict resolution, explanation or mutual regulation [1].

We use as a case study a problem entitled "the race with no winner" (and the Flash simulation) defined by a community of practice dedicated to the use of simulations in mathematics and physics [5]. The simulation embeds 10 cars that can be put on a track. The cars have different behaviours (e.g., speed or dynamics). Learners must first test all the cars (with the simulation) in order to collect the data necessary to the establishment of a relation between the departure position of every single car and its arrival on the arrival line. This requires solving equations to determine speed, duration and distance. When the learners are ready, the tutor puts one of the 10 cars on the track and designates 2 others. The learners have to put these 2 cars on the track in order for the 3 cars to arrive on the arrival line at the same time. When done, the simulation is run to check their solution. There are thus 3 phases: (1) preparing the data (measuring and calculating the data related to the 10 cars); (2) calculating where to put the 2 cars after the tutor has selected his car and has put it on the track (this is to be done in a limited amount of time, on the basis of the behaviours of these cars as calculated at the previous step); (3) simulation to check if it is a success or a failure.

2 Issue: Supporting Learners' Self-organization

A collective learning situation such as a pedagogic collective challenge is made up of two overlapping systems of activities: the collective problem-solving and the organization of this collective problem-solving. It corresponds to a particular case of collective *work situation*, i.e., a situation where the learners are mutually dependent in their work. Actors engaged in such interdependent processes must address an overhead activity, that of articulating (dividing, allocating, coordinating, scheduling,

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 743–745, 2008.

meshing, interrelating) their respective activities [4]. This meta-level overhead activity aims at maintaining a more-or-less stable pattern of cooperative arrangement.

The notion of *learners' self-organization* denotes the meta-level activity that a group of learners engaged in a CSCL setting may engage in so as to maintain, within the reference frame that is externally defined by the setting, a more-or-less stable pattern of collective arrangement [2]. "Self" is meant to highlight that, in such a context, part of the organization is set by the setting (here, the challenge) and part is related to emergent features of learners' enactment of the challenge at run-time.

In our case study, learners only have limited amount to calculate where to put their cars (phase 2; in our experiments we gave them 20 minutes). Therefore, in phase 1, they must not only prepare all the useful data (i.e., calculate the different cars behaviours), but also organize themselves for the second phase: identify what are the different tasks to be achieved during phase 2 (acquire x, measure y, calculate z), and decide how to organize themselves (who will achieve each subtask, when and how). As we observed it in our experiments, they naturally engage (however, to different extents) in adopting a more or less explicit strategy and setting up a form of monitoring and regulation of the process: they self-organize themselves.

Our research aims at designing a computer-based system that can (1) support learners in self-organizing themselves in the context of on-line mediated challenges and (2) provide tutors with means to analyze this organization and engage regulation actions. This requires, as a preliminary step, identifying a theoretical basis that helps analyzing the challenge enactment in a way that denotes self-organization issues.

3 Model, Results, and Directions for Design

In order to analyze self-organization issues we propose to use J. Bardram's model (Fig 1). This model aims at perceiving breakdowns that may appear during collaboration, as a way to help in understanding the collaboration dynamics [3]. It stresses the dynamic transformations that may appear in a collective activity between the co-ordination, co-operation and co-construction levels. We present here below these different levels and their instances in our case study, as shown by the exploratory experiments we conducted.

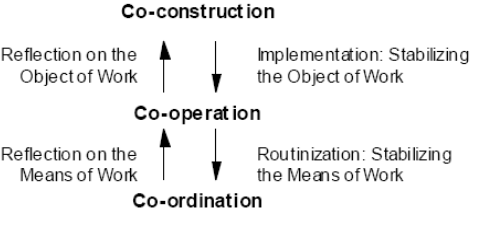


Fig. 1. Bardram's 3-levels model [3]

The co-construction is the level where actors focus on conceptualizing or re-conceptualizing their own organization and interaction in relation to their shared objects. In our case, the experiments corroborate the classical issue of common ground: it is of crucial importance that learners develop a common vocabulary to reflect both on organizational issues (e.g., subtasks) and domain-related issues (e.g., data to be acquired). A specific phase/place allowing the elaboration (and, in case of difficulty or breakdown, revision) of a common view and vocabulary, and of the general scheduling of subtasks, is required. Drawing learners' attention to this phase, proposing adapted means and allowing/facilitating tutor's monitoring and regulation

of this phase are critical to enhance the chances students involve in communication, argumentation, analysis or reflection related to both (1) the problem-solving strategy and division of labor and (2) the domain-level issues (here, mathematical issues).

The co-operation level is an intermediate level where actors are active at considering the shared objective. This enables them to relate to each other and make corrective adjustment to their own and others' actions according to the overall collective objective. In our case, this level relates with how to achieve what has been planned: the role of each member (task attribution, task decomposition if necessary), the means to achieve the tasks (e.g., a tool to help in editing and structuring the data), the sequencing, etc. Organization must be made visible and presented in a way that allows learners and tutors to understand it to its details.

The co-ordination level is the level where actors concentrate on the task they have been assigned. Their work is related to a common goal, but their individual actions are only externally related to each other: they realize the global task from the point of view of their individual activity. In our case, each learner is confronted with personal tasks: measuring distance or time, calculating speed, applying mathematical procedures, etc. Tasks, rules or roles have been fixed at the preceding level (and learners can come back to this upper-level by a bottom-up transition). Learners' work is both separated but coordinated with that of other learners.

Bottom-up transitions are related to an analysis of the object or the means of the work, which can occur in relation with a breakdown or an explicit shift of focus. Top-down transitions are related to the solving of problems and contradictions, and conduct to a stabilization of the object and means of the work. Transitions from one level to another can originate from two sources. Learners can spontaneously go from a level to another in relation with a difficulty they encounter, or by a voluntary shift of focus. Such transitions correspond to self-organization dimensions as defined previously. In our case another origin for transition appears: the learners' process is monitored by the tutor. He can launch regulation actions such as drawing learners' attention to the fact they should shift from a level to another (i.e., interact about a feature of another level than the current one) in relation with a problem encountered by a learner or by the group, an anticipation of a breakdown, a pedagogical opportunity, etc. This requires means to detect such issues.

References

1. Dillenbourg, P., Tchounikine P.: Flexibility in macro-scripts for CSCL. Journal of Computer Assisted Learning, 23(1), 1-13, 2007.
2. Tchounikine P., Operationalizing macro-scripts in CSCL technological settings. International Journal of Computer-Supported Collaborative Learning (in press; 2008).
3. Bardram, J.: Designing for the Dynamics of Cooperative Work Activities. In: Poltrock, S., Grudin, J. (eds) CSCW conference, pp 89-98. Seattle, 1998.
4. Schmidt, K., Bannon, L.: Taking CSCW Seriously: Supporting Articulation Work. CSCW 1(1-2), 7-40, 1992.
5. http://www.patrickmoisan.net/copains/course_sans_gagnant.html

Authoring Mobile Intelligent Tutoring Systems

Ramón Zatarain[1], M.L. Barrón-Estrada[1], Guillermo A. Sandoval-Sánchez[1], and Carlos A. Reyes-García[2]

[1] Instituto Tecnológico de Culiacán, Juan de Dios Bátiz s/n Col. Guadalupe, C.P. 88220 Culiacán, México, tel. +52 667 7131796
rzatarain@itculiacan.edu.mx
[2] Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE)
Luis Enrique Erro No. 1, Sta. Ma. Tonanzintla, Puebla, 72840, México
kargaxxi@inaoep.mx

Abstract. Tailoring material to the individual learner for delivery on handheld computers represents a major challenge. Most of the current work in mobile intelligent tutoring systems is more related with personalized systems because they do not support any artificial intelligent technique for implementing the adaptive part in the applications. In this paper, we present MLTutor, an author tool to facilitate the creation of adaptive learning material to be used in mobile devices applying an artificial intelligence approach.

1 Introduction

Adapting learning material to the individual learner for mobile computers is a current topic [1, 2, and 3]. However, while there is a growing interest in opportunities for adapting an interaction to the specific needs of the individual on handheld computers, this is mostly not concerned with authoring tools for self-regulated building of learning material.

The task of creating courses for mobile devices is not a simple work, and requires technical knowledge of the computer science field (design, programming, etc.).

In order to deal with the new challenge of producing adaptive learning applications, we implemented an author tool where a non-programmer instructor can more easily produce Intelligent Tutoring System for mobile devices.

Firstly, the author imports learning objects previously created under SCORM compliant and other files from different formats like PDF, DOC and HTML. The material is distributed under four different student learning styles according to the theory of Multiple Intelligences by Gardner [4]. This work can be performed in any computer with Java support. Secondly, the author exports the courses to the mobile device. The courses can dynamically recognize user learning characteristics and adaptively present customized learning material tailored to the learner. The tool has been tested extensively by creating several courses for different purposes and for different kinds of mobile technologies and architectures.

With respect to related work, there are some author tools used to create mobile applications like *MyLearning* [5], *zirada*[6], and Test Editor [3]. They are more focused to quiz editing or game-based learning. But, none of those author tools have the capability of adaptation to the user's form of learning and the tool's portability across different computer and operating system platforms.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 746–748, 2008.

2 Test and Results

Figure 1 shows a running example using **MLTutor** for an **Object-Oriented Analysis and Design** course. We show four different instances when building/editing the structure and the fuzzy sets and when executing the course in a mobile phone.

Fig. 1. A Running Example

3 MLTutor Architecture and Implementation

Figure 2 illustrates the architecture of the author tool. There exist two principal actors: the **author** and the **student**. The first one creates courses by editing the learning material to four different instances or student learning styles: Logical/Mathematical, Verbal/Linguistic, Visual/Spatial and Musical/Rhythmic. The **intelligent module** consists of three sub-modules: **fuzzy-logic, neural-network and linguistic-variables** sub-modules. The output of the editor can be a **Mobile** Course or a **SCORM** course. When a mobile course is exported to a mobile device, a XML **interpreter** is added to the course. This interpreter displays the material of the course into the mobile device according with the learning style of the student or learner. A **profile** of the student is produced according to linguistic variables stored in the intelligent module. The variables keep linguistic values on the learner's performance during the accessing to a course. Some values are: the time spent in a chapter or in a question, the number of correct answers, the answers selection order, etc.

MLTutor was implemented using Java and XML since both technologies are platform independent. The Intelligent Module was implemented by three components: a domain module, a user profile, and an inference engine.

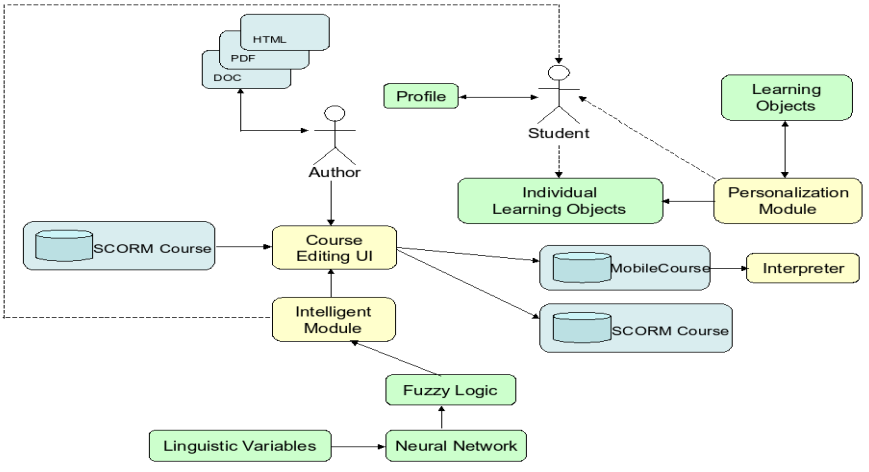


Fig. 2. General Architecture of MLTutor

4 Conclusions and Future Work

MLTutor is an author tool that allows an instructor to produce mobile intelligent tutoring systems. One feature of the tool is simplicity, allowing a fast creation of mobile applications. Future work includes implementing the part of the neural net so the intelligent tutoring system can also learn from experience. Another future step is to conduct a user study with learners of the products developed with the tool.

References

1. Bull, S., Cui, Y., McEnvoy, A.T., Reid, E., Yang, W.: Roles for Mobile Learners Models. In: Proceedings of the 2nd IEEE Workshop on Wireless and Mobile Technologies in Education (WMTE 2004), pp. 124–128. IEEE Computer Society Press, JungLi (2004)
2. Shih, K.-P., Chang, C.-Y., Chen, H.-C., Wang, S.-S.: A Self-Regulated Learning System with Scaffolding Support for Self-Regulated e/m-Learning. In: Proceedings of the 3rd International Conference on Information Technology: Research and Education (ITRE 2005), pp. 30–34. IEEE Computer Society Press, Hsinchu (2005)
3. Romero, C., Ventura, S., Hervás, C., De Bra, P.: An Authoring Tool for Building Both Mobile Adaptable Tests and Web-Based Adaptive or Classic Tests. In: Wade, V.P., Ashman, H., Smyth, B. (eds.) AH 2006. LNCS, vol. 4018, pp. 203–212. Springer, Heidelberg (2006)
4. Gardner, H.: Frames of Mind: The theory of multiple intelligences. Basic Books, New York (1983)
5. Attewell, J.: Mobile technologies and learning: A technology update and mlearning project summary. Learning and Skills Development Agency, London, `http://www.m-learning.org/reports.shtml`
6. Zirada Mobile Publisher, The Premier Mobile Content Creation Tool, `http://www.Zirada.com`

XTutor: An Intelligent Tutor System for Science and Math Based on Excel

Roxana Gheorghiu and Kurt VanLehn

Department of Computer Science, University of Pittsburgh
Sennott Square, Pittsburgh, PA 15260, USA
{roxana,vanlehn}@cs.pitt.edu

Abstract. VanLehn argued that an essential feature of many intelligent tutoring systems (ITSs) is that they provide feedback and hints on every step of a multi-step solution.But if step-based feedback and hints *alone* suffice for strong learning gains, as Anderson et al. conjecture ([1]), then perhaps a lightweight tutoring system that employ *only* feedback and bottom-out hints would have advantages. This motivates the current project.Using Excel there are some immediately advantages that can be obtained: most people is familiar with its user interface and its notation for mathematical expressions, Excel already contains facilities for solving some systems of equations and it can be easy combined with many other pieces of software, making it easier for instructors to include the tutor in their course activities. Finally, web-based delivery is simple because most students already have and use Excel.

1 Introduction

It was already tested on some well-kown ITS that providing feedback and hints on every step of a multi-step solution is an essential feature. It is not clear yet whether step-based feedback and hints alone suffice for strong learning gains, as Anderson at al. conjecture ([1]).

We wonder if the spreadsheet program Excel would allow easy construction of an ITS that would teach the same class of equation-based task domains as Andes (VanLehn at al. [3]) and Pyrenees (VanLehn et al. [2]). In these task domains, students solve a problem by defining variables, writing equations and finally solving the system of equations algebraically.

2 Using XTutor

As Fig. 1 illustrates, a problem in our Excel-based tutor, XTutor, is presented as an Excel worksheet with the statement of the problem on the first row of the sheet. Rows three and four briefly explain what should be entered in each column and how. All the other rows are filled in by the student.

Figure 1 shows the worksheet after it has been filled out by the student. Each row has:

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 749–751, 2008.

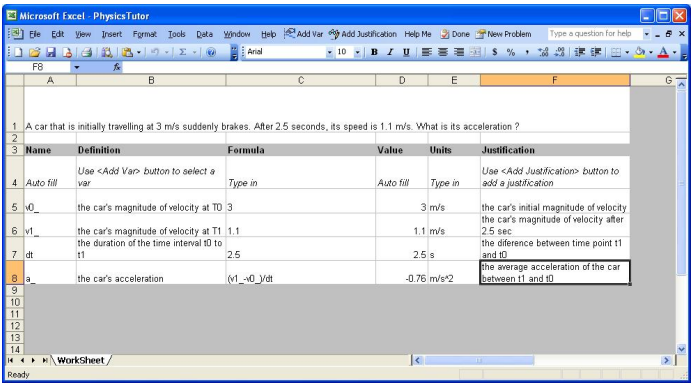


Fig. 1. Example of a problem in XTutor

1. a variable name that is automatically filled in by the tutor when the student selects its definition
2. a definition for the variable that is selected from a problem-specific menu of all possible variables definitions.
3. an algebraic formula, typed in by the student, to compute the variables value
4. a numerical value for the variable, filled in by the tutor, as the result of the previously introduced formula
5. units for the value, typed in by the student
6. the name of the principle that justifies the algebraic expression used for the variables value or other type of short justification selected by the student from a list of all possible justifications defined for that particular problem.

XTutor provides feedback on each student entry. Variables used in a formula that were not previously defined are signaled by a *#Name* type of error in the *Value* field. Algebraic expressions, units and justifications are also given immediate feedback. Moreover, if the student asks for a hint on a cell, the system can give a sequence of hints, where the final bottom out hint says exactly what should be entered in the cell.

At the end, the $<$ *Done* $>$ button checks if the problem was correctly and completely solved.

3 Authoring Domain Knowledge for XTutor

XTutor is an example-tracing type of tutor. That is, the domain knowledge of the system consists of a set of examples, one per problem. The example contains at least one and possibly several complete solutions to the problem.

The author enters an example exactly as a student would when solving the problem, except that the author must enter all the cells by herself. This suffices for giving immediate feedback and bottom-out hints. If the author wants any hints besides the bottom out hint on a cell, then the hints must be entered for that cell specifically.

4 Positive and Negative Features of XTutor

The main advantage of XTutor is its simplicity, for both students and authors. It takes only minutes to learn how to use it. A second feature is that once students have seen how to use Excel in this fashion to solve problems, they can continue to do so without the help of XTutor. Thus, we expect significant transfer from XTutor to unsupported problem solving using Excel.

However, there are certainly drawbacks to XTutor. In a solution, every variable appears in two or more equations. It is sometimes not clear to the student which equation should be written in the row for a variable. Students often prefer to work forwards from the given values. Usually, this eliminates all choices. However, even using this strategy, choices sometimes remain.

Take, for example, the following problem: *How many liters of 70% alcohol solution must be added to 50 liters of 40% alcohol solution to produce 50% alcohol solution?* Suppose the student is working on a row for tot_mix, which represents quantity, in liters, of the mixture. There are two choices: *tot_mix=tot_low+tot_high* or *tot_mix=alc_mix/pr_mix*, where *tot_low* and *tot_high* are the total quantity of low and high concentration solutions, *alc_mix* is the total quantity, in liters, of alcohol in the mixture solution and *pr_mix* is the percentage of alcohol in the mixture solution. If the student picks the first version then she must first think what formula to assign to *tot_high* (or *tot_low*) since *tot_high=tot_mix-tot_low* is not permitted anymore. Similar problems appear if the second formula is used to define *tot_mix*. When assigned to more skilled solvers, such problems may actually be beneficial, as they encourage students to plan their solutions before writing them down. Thus, we see this feature as positive but with a potential for misuse.

The domain generality of XTutor has already been tested in a preliminary way by authoring problems in three task domains: physics, math and chemistry. The next step is to try it out on students, first in pilot studies and then in comparison to paper-and-pencil and an established tutoring system, such as Andes.

References

1. Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive Tutors: Lessons Learned. The Journal of the Learning Sciences 4(2), 167–207 (1995)
2. VanLehn, K., Bhembe, D., Chi, M., Lynch, C., Schulze, K., Shelby, R., et al.: Implicit vs. explicit learning of strategies in a non-procedural skill. In: Lester, J.C., Vicari, R.M., Paraguaca, F. (eds.) Intelligent Tutoring Systems: 7th International Conference, pp. 521–530. Springer, Berlin (2004)
3. VanLehn, K., Lynch, C., Schultz, K., Shapiro, J.A., Shelby, R.H., Taylor, L., et al.: The Andes physics tutoring system: Lessons learned. International Journal of Artificial Intelligence and Education 15(3), 147–204 (2005)
4. Mitrovic, A., Suraweera, P., Martin, B.: Authoring constraint-based tutors in ASPIRE. In: Ikeda, M., Ashley, K., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 41–50. Springer, Heidelberg (2006)

Tying Ontologies to Domain Contents for CSCL

Seiji Isotani and Riichiro Mizoguchi

The Institute of Scientific and Industrial Research, Osaka University,
8-1 Mihogaoka, Ibaraki, Osaka, 567-0047, Japan
isotani@acm.org, miz@ei.sanken.osaka-u.ac.jp

Abstract. One of the main problems facing the development of ontology-aware authoring systems (OAS) is to link domain-independent knowledge (ontologies) with content of a specific domain. In collaborative learning (CL), this problem hinders the development of OAS that aids the design of pedagogically sound CL sessions with strong technological support. In this paper, we propose a framework to connect an ontology that represents CL explicitly with domain content without asking end-users to create their own ontologies from scratch.

Keywords: ontology-aware authoring system, ontological engineering, CSCL.

1 Introduction

One of the main problems facing the development of ontology-aware authoring systems (OAS) is to link well-designed domain independent knowledge (ontologies) with the contents of a specific domain [1]. In OAS for collaborative learning (CL), on one hand, we have a very powerful and sharable knowledge that can be used to support CL sessions with theoretical justifications. On the other hand, we have domain-specific content that needs to be adequately connected with theoretical foundations to provide a well-designed CL session. To solve this problem, this work proposes a framework that connects domain independent ontologies, specifically the CL ontology [2], with domain-specific content and learning objects (LOs) without the necessity of asking end-users to create new ontologies. This approach promotes a user-friendly way to implement the CL ontology by offering a graphical visualization of information along with templates that help users to link adequate LOs with the instantiated concepts in the ontology.

2 A Framework to Support Ontologies, Domain Content and LOs

In order to develop a system to support the design of CL activities based on ontologies, we have been developing CHOCOLATO (Concrete and Helpful Ontology-aware Collaborative Learning Authoring Tool)—an ontology-aware system that uses ontologies developed in the Hozo ontology editor (http://www.hozo.jp) to provide its theoretical knowledge [3]. One of its sub-systems, called MARI (Main Adaptive Representation Interface) allows the representation of learning theories on the screen using the Growth Model Improved by Interaction Patterns (GMIP). The GMIP is a graph model based on an ontological structure that describes an excerpt of learning

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 752–754, 2008.

theory. It represents, in a simplified way, the learner's knowledge acquisition and skill development processes, and explains the relationships between learning strategies, educational benefits and the interactions used to achieve these benefits.

Using the ontologies and the GMIP, MARI can select appropriate learning theories that support CL and suggest a consistent sequence of activities for learners in a group. The suggestions given by our system are guidelines that can be used to propose CL activities based on theories which (a) preserve the consistency of the CL process and (b) guarantee, to some extent, a suitable path to achieve desired benefits. MARI is strongly based on domain-independent ontologies. This means that it can provide domain-independent recommendations that can be used in different situations and are justified by theories, but it does not consider the actual domain in which the recommendations will be applied. Thus, to augment our research and show that a theoretically valid approach can be applied in real environments, we propose a framework to link domain-specific content into our model GMIP and our ontologies. The proposed framework, shown in Figure 2, has four linked layers. The top two layers are completely domain-independent, representing the knowledge about CL, learning theories and learning stages of a learner. The two bottom layers are related to domain-dependent content. One is related to the knowledge and skills of the domain-specific content and the other is related to the LOs connected with this content.

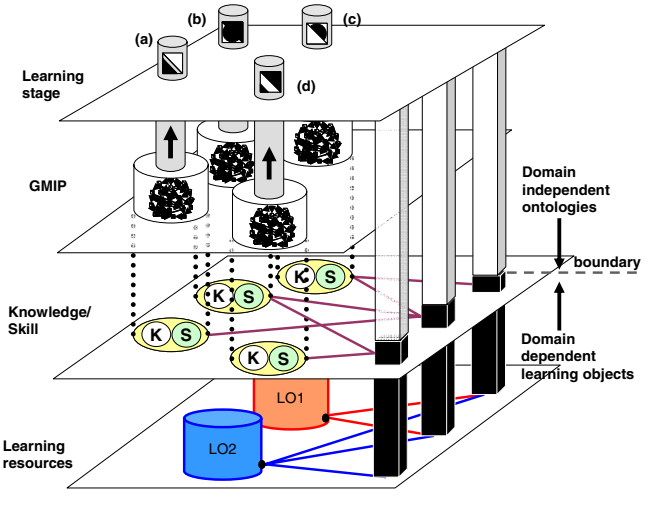


Fig. 1. Framework to link domain independent ontologies, domain specific content and LOs

We define the learning stage layer (top layer in Figure 2) as a set of nodes of different GMIPs where each node represents the stages of knowledge acquisition and skill development. The second layer is the GMIP. In this layer we show how learners can develop their knowledge/skills as transitions between nodes. In previous works we presented how this model was created and used to design CL activities [3].

To define the third layer and link it with the second layer, first of all, given a domain-specific content and a learning goal, we must separate the knowledge from the skills necessary to achieve this goal in the specified domain. The knowledge to

achieve the learning goal should be decomposed into different sub-knowledge pieces to be acquired. Similarly, the skills should be decomposed into sub-skills to be developed. The final structure will be a decomposition tree that identifies the minimum knowledge and skills necessary to achieve a domain-specific goal. The granularity of the decomposition tree depends on the learning goals and the expertise of the user who creates the tree. Note that this tree represents the knowledge and skills to be developed without any reference to how it will be developed. Using this approach, we can separate information about the content from information about how to learn the content. Such differentiation is important when we think about learner-centered environments where the environments adapt the way to provide information or the way to teach the same content according to learning/teaching preferences.

To complete the mapping of knowledge and skills into GMIP it is necessary to explicitly identify the relationship of the knowledge/skills in the tree. Each skill can be related to one or more pieces of knowledge and vice-versa. For each relationship knowledge-skill we can create an instantiated GMIP, which will then be able to support the development of this knowledge and skill in the specific domain. To facilitate such a task, we provide templates that help users to adequately understand the knowledge and skill development process. Furthermore, it helps to create a support system that semi-automatically maps specific knowledge/skill into GMIP and the CL ontology. The last layer in our framework is the learning resources layer. Each resource is a learning object that can be used to improve a domain-specific knowledge or skill. The LOs can be linked with the third layer and end-users can add/remove LOs to satisfy specific conditions in the environment where the learning will occur.

3 Conclusions

To create intelligent educational systems based on well-grounded theoretical knowledge and to apply them in real environments are two important challenges that research in the development of ontology-aware systems are facing nowadays. In order to solve these problems in the context of CSCL, we proposed a framework that intends to connect the CL ontology [2] with domain content and LOs intermediated by our model GMIP. By providing this connection, we can offer a more user-friendly way to design pedagogically sound CL sessions in a specific domain with strong technological support. Such an approach seems to be more reliable than other approaches, especially because it removes the burden of asking end-users to create ontologies for each domain of application.

References

1. Knight, C., Gasevic, D., Richards, G.: Ontologies to integrate learning design and learning content. Journal of Interactive Media in Education 2005(07), 1–24 (2005)
2. Inaba, A., Mizoguchi, R.: Learners Roles and Predictable Educational Benefits in Collaborative Learning. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 285–294. Springer, Heidelberg (2004)
3. Isotani, S., Mizoguchi, R.: Deployment of Ontologies for an Effective Design of Collaborative Learning Scenarios. In: Haake, J.M., Ochoa, S.F., Cechich, A. (eds.) CRIWG 2007. LNCS, vol. 4715, pp. 223–238. Springer, Heidelberg (2007)

One Exercise – Various Tutorial Strategies*

George Goguadze and Erica Melis

University of Saarland
and German Research Institute for Artificial Intelligence (DFKI)
george@activemath.org, melis@dfki.de

Abstract. Narciss' theoretical framework for informative tutorial feedback (ITF) suggests to adapt the feedback along the dimensions: content, procedure, form, and presentation according to the task, the learner's response and to the learner's characteristics and particular situation. As prerequisites for the adaptations we devised a knowledge representation for exercises to which various tutorial and presentation strategies can be applied. We also developed techniques for generating the procedure, form and presentation of feedback.

1 Introduction

Widely investigated feedback strategies in computer-based instruction are 'knowledge of result' (KR) that just informs the learner whether her answer is correct or incorrect, 'knowledge of the correct response' (KCR).[1] Informative Tutorial Feedback (ITF) refers to elaborate feedback types that provide strategically useful information. The empirical results about the benefits of those strategies are inconclusive.

[5] provides a theoretical framework for ITF that postulates the need to adapt the feedback's content, procedure, form, and presentation not only to the student's response and task but also to the learner's characteristics and to the particular situation. Hence, our ultimate goal is to adapt feedback in ActInMath along those dimensions.

As prerequisites for the adaption of feedback in interactive (multi-step) exercises we devised a knowledge representation for exercises to which a number of tutorial and presentation strategies can be applied. We also implemented functions that transform an exercise representation into a representation with a common tutorial stratgey extracted from teachers' practice, Naricss' experiments, and our own tutoring experience.

2 Generation of Tutorial Strategies

ActInMath's exercise system player can handle pre-scripted and generated exercises. Pre-scripted exercises are authored as finite state machines (FSM)

* This publication was supported by Deutsche Forschungsgemeinschaft, DFG, project ATuF ME 1136/5-1.

[1] Also called bottom out.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 755–757, 2008.

including edges representing correct and typical incorrect student input and its diagnosis, and nodes representing the system's (re)action to correct responses and to the causes of errors.

There are three types of states in an exercise FSM: tasks, so-called interactions representing interactive elements using which the learner enters the solution, and feedbacks of different types. As we see later, such separation provides a basis for reusability of different states when applying tutorial strategies.

Exercises can be also completely generated with the help of a domain reasoner. A partial FSM of such an exercise is dynamically generated from the solution space of the problem produced by the domain reasoner.

A Tutorial Strategy defines the *procedure* of the interactions, as well as *form* and *content* of feedback, presented to the learner. These dimensions of feedback are considered to be important for learning, as suggested in [5]. Therefore, our main research question is how to devise Tutorial Strategies and exercise representations enable adaptation to these substantiated dimensions.

Technically, in order to change the procedure and the form of feedback, (as part of) an exercise FSM is transformed into another FSM. In order to choose appropriate feedback content, a feedback generator component is invoked. Feedback generator first tries to find the feedback of the needed type in the exercise FSM and if such is not authored, generates it. Since interactions are separated from tasks, the strategy can vary the procedure of interactions without modifying the tasks, the form and content of feedbacks can be varied as well in which the feedbacks with needed properties only are shown or replaced with generated ones.

Two commonly used tutorial strategies that we have implemented are *decompose-into-subgoals* and *simpler-version*.

The algorithm for *decompose-into-subgoals* transforms a problem statement into subgoals - this is a procedure change. The algorithm for *simpler-version* strategy transforms the content. The resulting feedback suggests to solve an easier problem for the same concepts and competencies, and when the student succeeds with that the original problemis reinvoked.

The set of simpler versions of a problem may be dependent on several parameters, such as the focus concepts, the task, learner's mastery values etc.

A Presentation Strategy defines the GUI appearance of the exercise. It defines how parts of the exercise states are rendered. This includes windows, buttons, placement of feedbacks within a window and other presentational aspects such as different foreground and background color, highlighting, icons, etc. A Presentation Strategy can also define whether previous responses, feedback and hints should be visible or not.

3 Recent Related Work

Instructional benefits of elaborate feedback were obtained in empirical studies which selected the feedback components on the basis of cognitive task and error analyses, and assembled them as a multiple try feedback (e.g., [1,6]). The

relatively simple variations for elaborate feedback included a varying number of student trials plus location of error and KCR vs. KR and KCR only.

A corpus study of human-human dialogues in foreign language learning [3] elicited a number of feedback strategies teachers use in foreign language learning: for positive feedback this includes acknowledgement, acceptance, repetition, and rephrasing; corrective giving-answer strategies include repetition, recast, explicit correction, give answer, show (location of) error; corrective prompting-answer strategies include clues without giving the target form, clarification request, and elicitation.

Related work also includes Hefernan's Ms.Lindquist [4] that has four feedback strategies for algebra word problems: concrete-articulation, explain-verbally, abstraction-and-substitution, and worked-example. Its rich tutorial strategies resulting from observation of tutors are used in a fixed way and encoded in the exercise representation. The actual presentation is fixed too.

4 Future Work

The issue how and when Tutorial and Presentation Strategies should be used to optimize learning is future collaboration with our psychology partners. Based on their empirical results we shall devise a model for the adaptation of tutorial strategies and presentation strategies. This model will comprise not only the type and cause of error and task but also the student's competencies and motivation.

For instance, if a learner has a weak self-efficiacy, it is detrimental to provide negative feedback [2]. Therefore, an alternative feedback to an incorrect answer could be to pose a similar but simpler task which still trains the same skill.

References

1. Albacete, P., VanLehn, K.: Evaluating the effectiveness of a cognitive tutor for fundamental physics concepts. In: Proceedings of the Annual Meeting of the Cognitive Science Society, Mahwah, NJ, LEA (2000)
2. Eckert, C., Schilling, D., Stiensmeier-Pelster, J.: Einfluss des Fähigkeitsselbstkonzepts auf die Intelligenz- und Konzentrationsleistung (in German). Zeitschrift für Pädagogische Psychologie 20(1-2), 41–48 (2006)
3. Ferreira, A., Moore, J.D., Mellish, C.: A study of feedback strategies in foreign language classrooms and tutorials with implications for intelligent computer-assisted language learning systems. International Journal of Artificial Intelligence in Education 17(4), 389–422 (2007)
4. Heffernan, N.T.: Intelligent Tutoring Systems have Forgotten the Tutor: Adding a Cognitive Model of Human Tutors. PhD thesis, School of Computer Science, Carnegie Mellon University (2001)
5. Narciss, S.: Informatives tutorielles Feedback. Waxmann, Muenster (2006)
6. Narciss, S., Huth, K.: How to design informative tutoring feedback for multi-media learning. In: Instructional Design for Multimedia Learning, pp. 181–195. Waxmann, Münster (2004)

Bi-directional Search for Bugs: A Tool for Accelerating Knowledge Acquisition for Equation-Based Tutoring Systems

Sung-Young Jung and Kurt VanLehn

Intelligent System Program
University of Pittsburgh
{chopin,vanlehn}@cs.pitt.edu

Abstract. Authoring the knowledge base for an intelligent tutoring system (ITS) is difficult and time consuming. In many ITS, the knowledge base is used for solving problems, so authoring it is an instance of the notoriously difficult knowledge acquisition problem of expert systems. General tools for knowledge acquisition have shown only limited success, which suggests developing tools that apply only to specific kinds of knowledge bases. Pyrenees is an ITS whose knowledge base is composed mostly of conditioned equations. We have developed several tools for authoring Pyrenees knowledge bases. This paper focuses on a novel and particularly powerful tool that uses bidirectional search to locate bugs in the knowledge base. In several evaluations, human authoring was significantly faster when the tool was available than when it was unavailable

Keywords: Bi-directional search, authoring tools, automatic error detection on knowledge.

1 Introduction

ITS can be classified depending on how their inner loops represent domain knowledge. One classification, which is the focus of this paper, is tutors whose domain knowledge solves the same problems that the students does and thus "model" the desired ways to solve them. These ITS are sometimes called *model-tracing tutors*, although that term is often taken to denote the particular technology used by CMU tutors [8] and Carnegie Learning (http://www.carnegielearning.com/).

In this paper, we describe the authoring tools used with Pyrenees [10] [11]. Pyrenees uses a large set of knowledge components, called principles, to solve a problem. However, what makes Pyrenees unusual is that most of its principles are conditioned equations. That is, such a principle asserts that under certain conditions, a certain equation is true. This allows a sophisticated error detection method to be used to located buggy principles.

In Pyrenees, Each principle is represented by a condition and an equation, where the condition indicates when the equation holds (Fig. 1). A set of problems, like the principles, are expressed in terms of the ontology. Fig. 2 illustrates a problem named "isobaric expansion of water" which would be stated in English as "A reservoir contains a liquid,

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 758–762, 2008.

called water, of mass 0.001 kg and heat capacity 4,196 J/(kg*C). The pressure is held constant at 200,000 Pa. As the water is heated, it increases its volume by 0.000001 m^3 and its temperature by 31 C. What is the work done during this time?"

```
Pa_true(isobaric_expansion(Gas, T) ):-
      gas(Gas),
          time_interval(T),
      constant(var(at(pressure(Gas),
T)).
```

```
    Pa_equation(isobaric_expansion(Gas,          T),
W=P*Vdiff):-
                W=var(at(work_done_by(Gas), T)),
                P=var(at(pressure(Gas), T)),
                Vdiff=var(at(diff(volume(Gas), T))).
```

Fig. 1. Predicates defining a condition (pa_true) and equation (pa_equation)

```
p_definition(isobaric_expansion_of_water,
 [substance(liquid),
  reservoir(liquid, _, _),
  known(var(mass(liquid)), dnum(0.001, kg)),
  %heat capacity
  known(var(heat_capacity(liquid)), dnum(4196, 'J/(kg*C)')),
  known(var(pressure(liquid)), dnum(200000, 'Pa')),
  known(var(diff(volume(liquid))), dnum(0.00000001, 'm^3')),
  known(var(diff(temperature(liquid))), dnum(31, 'C')),
  sought(var(work(liquid)), dnum(0.002, 'J'))          ]).          %answer; 0.002
```

Fig. 2. An example of domain problem

Pyrenees solves problems via a version of backward chaining called the Target Variable Strategy [10] [11]. The basic idea is simple: Given a sought quantity, generate an equation that applies to this problem and contains the sought quantity. Include it in the set of equations that comprise the solution. If the equation has any quantities in it that are neither known nor previously sought, then treat them as sought and recur. When the Target Variable Strategy stops, it has generated a set of equations that are guaranteed to be solvable.

2 Error Detection Using Bidirectional Search

Whenever we add a new problem, we call the problem solver to see if it can be solved. The most frequent sign of a bug is that a problem cannot be solved. This occurs when some principle that should have applied did not get applied. One heuristic to try to find the spot where the principle should have applied is to focus on the dead ends. A dead end is a sought quantity that cannot be calculated from the known quantities. It is likely, but not certain, that a principle should be applied to the dead end, but it failed to apply because the principle was buggy. Although one could examine the dead ends of the tree by hand, there can be hundreds of them.

Bugs that prevent a principle from applying can appear in 3 locations. The bug could be in (1) the principle's condition or (2) in the problem statement. Pyrenees also has a few knowledge components that are now conditioned equations, but instead draw inferences that bridge between a principles condition and the problem statements. (3) Bugs in

these rules can also prevent a principle from applying. Fig. 3 illustrates 3 different kinds of errors. These errors are common, hard to notice by inspection and will block a principle from applying.

The bi-directional tool starts by creating all possible forward chaining trees. That is, it generates all applicable equations, then use them repeatedly to generate values for all possible quantities. None of the dead end quantities will be among these quantities with known values, because otherwise they would not be dead ends.

```
pa_true(isobaric_expanson(Gas, T) ):-          %  Typo. Should be "expansion"
        gas(Gas),
         time_interval(T),
        constant(var(at(pressure(Gas)), T)).   % Wrong parenthesis.  Should be …Gas), T)))
constant(Quantity):-
        known(Quantity, _),
        \+(    Quantity=at(_, T)
           ;     time_interval(T)   ).         % OR (;) should be AND (,)
```

Fig. 3. Examples of bugs

However, a dead end may be "one equation away" from the known quantities. Thus, given a dead end quantity, we search for a principle such that one (or more) of its variable specifications unifies with a dead end quantity (*W* in Fig. 4) and all of the remaining variable specifications unify with known quantities (*P* and *Vdiff* in Fig. 4). If we find such a principle, then we know that if it had applied, the dead end would not exist. Thus, a bug must be preventing the principle's condition from unifying with the problems' description.

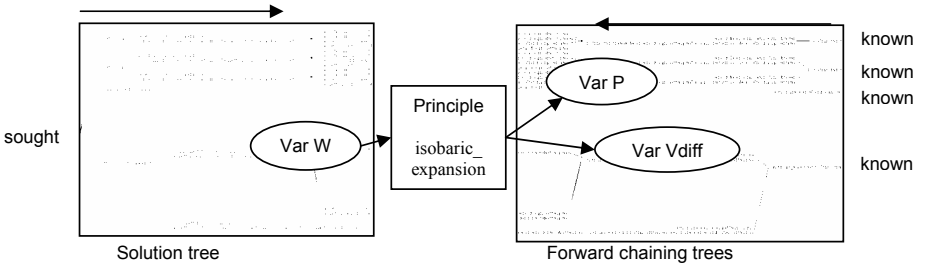
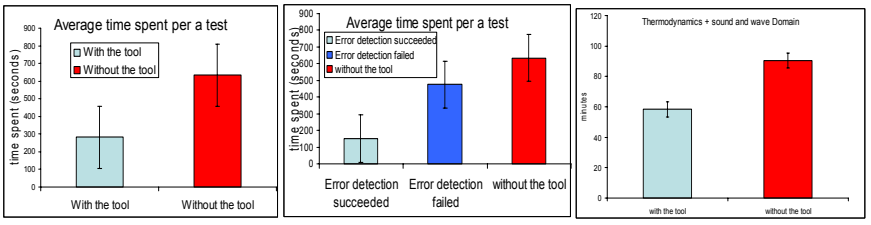


Fig. 4. Bi-directional search identifies a principle whose condition isn't satisfied

3 Evaluation and Conclusions

In order to evaluate the effects of using the bi-directional tool, two kinds of experiments were performed. The task domain was thermodynamics, and it included 14 problems and 15 conditioned-equation principles. The lead author of this paper was the debugger.



(a) (b) (c)

Fig. 5. The result of experiments

Fig. 5-(a) and (b) shows the results for error detection tests on automatically generated errors. The average time spent per a test with the tool was shorter than without the tool (281 vs. 634 seconds, p = 0.00008). Fig. 5-(b) shows debugging time using the tool either the cases when error detection succeeded or failed. The average time spent when error detection succeeded was shorter than when failed (151 vs. 474 seconds). When error detection failed, the time spent didn't show significant difference from the time without using the tool (p = 0.35).

For the authoring test, the number of domain problems was 13 for each domain (total 26). Thermodynamics and sound/wave domains were implemented in each of two passes. Fig. 5-(c) shows the result of knowledge adding tests using the alternation scheme. Authoring the tool was significantly faster than without using the tool (58.38 minutes vs. 90.36 minutes, p =0.017).

Although Pyrenees already had state-of-the-art tools for authoring, including a visualization tool, ontology-based checking of principle semantics and regression testing, we found that a new tool, based on bi-directional search, significantly accelerated authoring. When a problem cannot be solved, the tool searches through the dead ends in the tree generated by the normal, backward chaining problem solver used by Pyrenees. If it can find a dead end that is "one principle away" from a known quantities developed by forward chaining, then it is likely that this principle should have applied but did not because its condition failed to match the problem's description. This localizes the bug, which makes it much easier to spot. Evaluations indicated that the tool saved time, and the difference was statistically reliable.

References

1. Murray, T.: Authoring Intelligent Tutoring Systems: An Analysis of the State of the Art. International journal of Artificial Intelligence in Education 10, 98–129 (1999)
2. Gil, Y., Melz, E.: Explicit Representations of Problem-Solving Strategies to Support Knowledge Acquisition, ISI Technical Report ISI/RR-96-436 (1996)
3. Yost, et al.: Acquiring Knowledge in Soar. IEEE EXPERT (1993)
4. Noy, N.F., Grosso, W.E., Musen, M.A.: Knowledge-Acquisition Interfaces for Domain Experts: An Empirical Evaluation of Protege-2000. In: Twelfth International Conference on Software Engineering and Knowledge Engineering (SEKE 2000), Chicago, IL (2000)
5. Tallis, M., Kim, J., Gil, Y.: User studies of knowledge acquisition tools: methodology and lessons learned. J. Expt. Theor. Artif. Intell. 13, 359–378 (2001)

6. Gil, Y., Kim, J.: Interactive Knowledge Acquisition Tools: A Tutoring Perspective. In: Proceedings of the 24th Annual Meeting of the Cognitive Science Society (CogSci), Fairfax, VA, August 8-10 (2002)
7. Turner, T.E., Koedinger, K., et al.: The Assistment Builder: An Analysis of ITS Content Creation Lifecycle. In: The 19th International FLAIRS Conference, May 11-13 (2006)
8. Anderson, J., Skwarecki, E.: The Automated Tutoring of Introductory Computer Programming. Communications of the ACM 29(9), 842–849 (1986)
9. Aleven, V., McLaren, B., Sewall, J., Koedinger, K.R.: The Cognitive Tutor Authoring Tools (CTAT): Preliminary evaluation of efficiency gains. In: Ikeda, M., Ashley, K., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 61–70. Springer, Heidelberg (2006)
10. Chi, M., VanLehn, K.: Accelerated Future Learning via Explicit Instruction of a Problem Solving Strategy. In: Koedinger, K.R., Luckin, R., Greer, J. (eds.) Artificial Intelligence in Education, pp. 409–416. IOS Press, Amsterdam (2007)
11. Koedinger, K.R., Aleven, V., Heffernan, N.T., McLaren, B., Hockenberry, M.: Opening the door to non-programmers: Authoring intelligent tutor behavior by demonstration. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 162–174. Springer, Heidelberg (2004)
12. VanLehn, K., Bhembe, D., Chi, M., Lynch, C., Schulze, K., Shelby, R., et al.: Implicit vs. explicit learning of strategies in a non-procedural skill. In: Lester, J.C., Vicari, R.M., Paraguaca, F. (eds.) Intelligent Tutoring Systems: 7th International Conference, pp. 521–530. Springer, Berlin (2004)

Design of a System for Automated Generation of Problem Fields

Ildikó Pelczer and Fernando Gamboa Rodríguez

Instituto de Ingeniería, Universidad Nacional Autónoma de México, Av. Universidad 3000, Coyoacán, 01450, México D.F., México
IPelczer@iingen.unam.mx, fernando.gamboa@ccadet.unam.mx

Abstract. We describe the design of a computer system for problem field generation in mathematics. The concept of *problem field* refers to a sequence of related problems. The relationship between the problems assures that their successive solution promotes deep learning of the topic (sequence problems). We define four relations between the field's problems: difficulty and rule, question or property preservation. The problem's difficulty is given by problem-type, algebraic complexity and difficulties of the problem's question and rule used for solution. A problem field based on difficulty will modify one of the above elements in order to get new problems from existing ones. In rule- and property-preserving generation the field's problems differ in their type or difficulty level. In question preserving generation the problems maintain the initial problem's unknown. Once the details of the change are established the new problem will be generated using templates, rules and examples.

Keywords: automated problem generation, problem fields, mathematics.

1 Introduction

During the last decade automated problem generation received considerable attention, not only from specialists involved in the development of large scale tests but also from researchers in the field of Intelligent Tutoring Systems. The employed generation methods vary considerably [1, 2] as well the application domains.

Our approach is different since we part from a cognitive account of problem generation, that is, we build on empirical experiments of problem generation done by students and teachers [3]. By such way we expect to extend the systems capabilities beyond that of the use of predefined rules. The application domain is mathematics.

The main purpose of mathematical education is to lead to deep learning [4]. Pehkonen suggests to use open-ended tasks for such purpose and propose the use of a set of connected problems (*problem field*). There is no explicit specification of the type of connection; but it is important to draw the student's attention to an aspect that would not be so evident with isolated problems. In the paper we describe the design of our system for problem field generation in the domain of mathematical analysis, specifically in the topic of numerical sequences.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 763–765, 2008.

2 System Design

In this section we describe the system's design from two point of view: the process of problem field generation and the employed knowledge structures.

The automated process starts with selecting the connection between the problems, defining the problem templates and enters a "pose-improve" cycle. There are four types of relations defined in the system: increasing difficulty, rule, question and property-preservation. The term strategy denotes the process of electing one of these relations and describes the generation process at high level. The selected strategy imposes constrains on problem characteristics. For example, the rule-preservation strategy restricts the types of problems that can be used. Once a problem type has been selected, the system draws on the knowledge stored in the system to fill in the missing elements. The "pose-improve" cycle it is necessary, since it can happen that the proposed problem is not satisfactory (it is not enough difficult). The process is synthesized in figure 1. The knowledge base is stored in the long-term memory (LTM), meanwhile the problem under construction is in the short time memory (STM). The "pose-improve" cycle transfers information from LTM to STM.

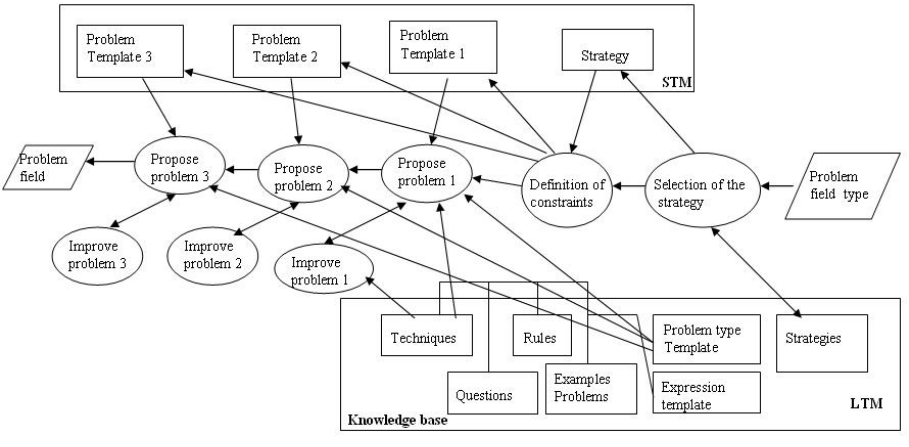


Fig. 1. The problem field generation process

It has to be said that the cyclic view of problem field generation requires to define some criteria for assessing problem difficulty and ending the generation process. Problem difficulty arises from the employed rule, expression, question and the type of the problem. Rules have associated a difficulty level in function of their use in textbooks: more common rules are easier. The difficulty of the problem's expression depends on the presence of parameters, number of involved concepts, needed transformations and algebraic complexity. Question difficulty depends on problem type. Rules, questions and problem types were identified from textbooks. The condition for problem field generation is to propose problems with increasing difficulty. The generation process ends when problems have the desired difficulty.

The knowledge base contains: strategies, templates for problem types and expressions, examples, domain-specific rules, questions and techniques for modifying problems. The problem type templates describe the form of the general term. Expression templates refer to generic expressions that can be instantiated by replacing variable values with constants. The rules allow to combine previous problems into a new one. Questions refer to some property of the sequence. The techniques describe ways to obtain some expression with desired property. The frame of the rules and techniques have two special slots. One stands the condition of application and the second for the result. This information helps to connect the steps in the generation process. In figure 2 we give an example of the generation process.

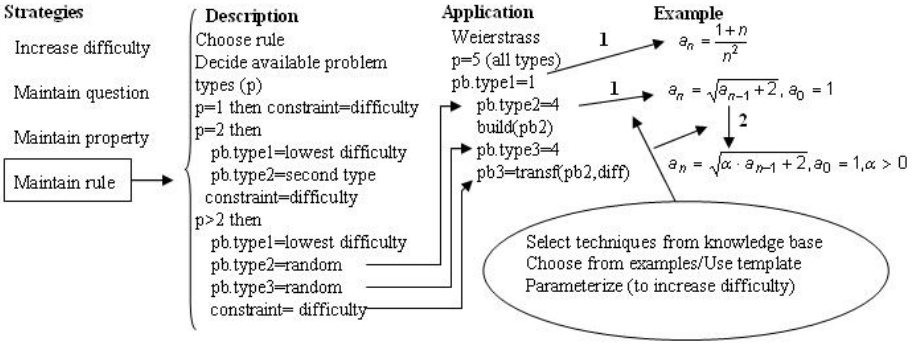


Fig. 2. The outline of an example generated by the "maintain rule" strategy

The strategy specifies the set of actions to be carried out. Next the rule is chosen and then decided the available problem types. The system starts with the type of lowest difficulty as problem 1 and then the next problems are chosen from types with higher difficulty. An example generation is given for the Weierstrass rule.

In conclusion, the systems' design is based on empirical studies carried out with pupils, students and teachers. The generation process is modeled as a "pose-test" cycle that starts with the selection of a strategy. At its turn, the strategy impose restrictions on the problems. During "pose-test" cycle the problem is constantly modified until the desired level of difficulty is achieved.

References

1. Cristea, P., Tuduce, R.: Automatic generation of exercises for self-testing in adaptive e-learning systems: exercise on AC circuits. In: Proceedings of 3rd International Workshop on Authoring for adaptive and adaptable educational software, Amsterdam (2005)
2. Le, N.T., Menzel, W.: Constraint-based problem generation for a self-assessment system. In: ICCE 2006 Workshop Proceedings of Problem-Authoring, -Generation and Posing- in a computer-based learning environment, The 14th Conf. on Computers in Education (2006)
3. Pelczer, I., Gamboa, F.: Problem Posing Strategies of Mathematically Gifted Students. In: 5th Conf. on Math Education and Education of the Gifted, Haifa, pp. 174–182 (2008)
4. Pehkonen, E.: Using Problem Fields as a Method of Change. Mathematics Educator 3(1), 3–6 (1992)

Lessons Learned from Scaling Up a Web-Based Intelligent Tutoring System

Jozsef Patvarczki, Shane F. Almeida, Joseph E. Beck, and Neil T. Heffernan

Worcester Polytechnic Institute
Department of Computer Science
100 Institute Road
Worcester, MA 01609
{patvarcz,almeida,josephbeck,nth}@wpi.edu

Abstract. Client-based intelligent tutoring systems present challenges for content distribution, software updates, and research activity. With server-based intelligent tutoring systems, it is possible to easily distribute new and updated content, deploy new features and bug fixes, and allow researchers to more easily perform randomized, controlled studies with minimal client-side changes. Building a scalable system architecture that provides reliable service to students, teachers, and researchers is a challenge for server-based intelligent tutors. Our research team has built Assistment, a Web-based tutor used by hundreds of students every day in the Worcester and Pittsburgh areas. Scaling up a server-based intelligent tutoring system requires a particular focus on speed and reliability from the software and system developers. This paper discusses the evolution of our architecture and how it has reduced the cost of authoring ITS and improved the performance and reliability of the system.

Keywords: ITS, Web-based tutor, scaling, architecture.

1 Introduction

The idea of Web-based intelligent tutoring systems has been around for many years. Ritter and Koedinger suggested the possibility transitioning to Web-based tutors as early as 1996 [4] and he predicted many of the performance problems that we experience today.

Server-based tutoring systems have many advantages over client-based approaches [1]. Accessibility is an important concern for tutoring systems. Students, teachers, and content creators all must have access to the system. Because of widespread Internet access, Web-based tutoring systems have the potential to provide access to many more users than can be reached with client-based software. Brusilovsky has claimed that Web-based systems have longer lifespans and better visibility than client-based [2]. Web-based systems virtually eliminate much of time and cost of installing software on individual client machines. Another advantage of server-based architectures is greater control over content distribution. Software updates and configuration changes are easily manageable

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 766–770, 2008.

with server-based architectures. Data collection is simplified by a centralized system. In addition, the data are available immediately in the form of reports.

The disadvantage of server-based systems is scalability as centralization of resources can create bottlenecks. This paper describes our ongoing efforts and solutions for addressing scaling problems in our system. The first part of the paper discusses different architecture problems and solutions through our research project evolution. Second, we propose a new hybrid architecture that addresses many scalability issues for ITS researchers.

2 Architectures

We were interested in architecture models for load-balancing and for fault-tolerance for online learning, which is becoming increasingly important [6]. In the case of a 24 hour/7 day per week service, reliability is of great importance. The system needs to be active and responsive otherwise we can lose valuable students, teachers, and schools who are unwilling to wait for service.

2.1 First and Second Generations — Tomcat Implementation with Scalability

The first version of the system was in Java. Our infrastructure contained two Web servers, two Tomcat application servers, two load balancers (master-slave), Web cache system, and one back-end database server [3]. Our software has four main components: 1) a "portal" though which students and teacher log into, 2) a "runtime" where students get tutored, 3) a "reporting" system where teachers get reports on what their students have done, and 4) a "content builder" used by teachers and researchers at CMU and WPI.

Given that we run about 100 classes, we wish to see how far we are from supporting all the 8th grade students in the state of Massachusetts. First, we were interested in the communication between the application server and the database server. We set up a test environment with one application and a database server that used a browser to play student and tried to capture the time and the number of reads and writes at the back-end.

With the old system and infrastructure the most important result shown in Figure 1 is that all but a handful of these 10-minute interval have average page creation times that are less than one second. Most of these are clustered around 100 milliseconds. Since each public school classes have about 20 students, we noticed clusters (shown in ovals in the bottom left) of intervals where a single class was logged on. We noticed a second cluster of around 40 users, which most likely represents instances where two classes of students were using the system simultaneously. Surprising to us, even the intervals where 80–100 users were on the system simultaneously, there was no appreciable pattern towards a slower page creation time.

With our second generation system, we wanted to evaluate our decision to increase scalability by adding an application server to our load-balanced infrastructure. Our research question was whether we could get a linear speed-up with

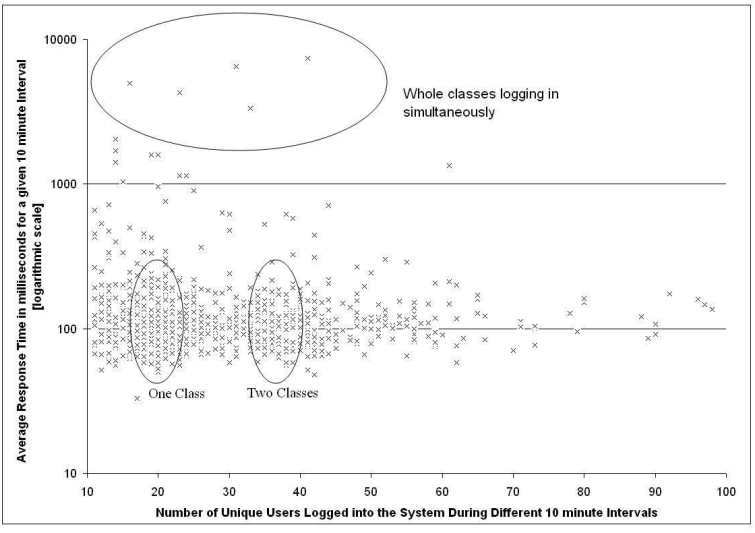


Fig. 1. Average response time of the first generation Assistment system

additional Tomcat servers or not. We also wanted to see if a caching method could increase performance.

It does seem that the load balancer did appear to cut the response time in half; however, the caching method had a much less significant role in reducing the average response time. We seem to have able to get linear speed-up with the help of the load-balancer and an additional Tomcat server.

2.2 Third Generation — Reimplementation in Ruby on Rails

As our project progressed, the complexity of our Java source code reached the limits of maintainability. The size of the code base, contributions by many different developers, and thousands of lines of uncommented code made the system difficult to learn. A reimplementation was need and Ruby on Rails, a free Web application framework based on the open-source Ruby programming language, was chosen for the rewrite.

Because Rails and Java are fundamentally different systems, our infrastructure required a redesign to accommodate the new system. Mongrel, a single-threaded Web server for Ruby applications, is used to serve content. Because it is single-threaded, multiple Mongrel application servers are used concurrently in a cluster. A simple Apache Web server provides load balancing to the cluster and a single PostgreSQL database server stores all content and data.

As more students began using the system simultaneously, we encountered severe scaling problems. Examining loads on the application and database servers revealed that the database was a serious bottleneck in the system. During peak times, the database server was constantly at maximum capacity while the

application servers remained nearly idle waiting for data. Because of the slow database response time, the incoming request rate quickly exceeded our service rate. Examining the most expensive operations in the system revealed some implementation flaws in our system.

According to log data, the tutoring portion of our system spent an average of 2.27 seconds processing a request (standard deviation of 3.39). With six Mongrel servers, our system could only handle approximately three requests per second. In addition, analysis demonstrated that we encountered scalability problems at very low usage.

2.3 Fourth Generation — Rails with Thicker Client Architecture

In light of serious design problems, we focused on fundamental changes that would allow the system to scale well beyond our current user base. Instead of handling all data processing ourselves, we decided to take advantage of the computing power of our users by pushing many of the basic tasks to the clients. However, we still wanted to avoid a thick-client implementation for the reasons discussed previously.

For the reasons described, our servers are still responsible for on-demand content distribution and log gathering and retention, while data processing (e.g., determining the correctness of answers) has been re-implemented in JavaScript and is now the responsibility of the client system. This redesign has reduced load on the database dramatically.

Latency is a frequent problem for Web users. Stern, Woolf, and Kurose discussed the idea of prefetching content in their MANIC system [5]. To address this concern, at the start of an assignment, the client requests and stores questions asynchronously while the student begins working. Network or server delays are hidden by the asynchronous communication. In initial testing of the JavaScript implementation of the tutor, our system spends an average of 0.08 seconds processing a request (standard deviation of 1.43). With just six Mongrel servers, our system is now posed to handle 75 requests per second. Even if our throughput does not match this estimate, latency hiding via asynchronous requests will help provide a fast user experience.

3 Future Work

Our goal is to provide reliable and effective tutoring to hundreds of thousands of users. Although we do not have a total solution yet, we are actively exploring architecture designs that will allow us to meet the goal. Our client-server hybrid solution is a step in this direction. However, the database is a single point of failure and a significant bottleneck in the system. Future work will investigate database scalability.

We currently have Assistment content hosted as a Web service. Because of this, by implementing a Web service interface, another ITS could incorporate Assistment content into their system. The Assistment project is looking to collaborate with other systems to include various features. This is just one example

of how Web services have enabled the Assistment project to become more collaborative and extensible. Intelligent Tutoring Systems are customarily thought of as closed systems. With such a strong research community, it would seem beneficial for researchers to be able to work together; using contents and tutoring methods from within other existing system. With this collaboration, existing tutors could reach broader audience.

4 Conclusions

We have reported some positive results with regard to scaling up ITS. Clearly, many of the issues discussed in this paper are specific to our implementation. However, the underlying architecture decisions we have made are of general interest to the Web-based tutoring community. Server-based systems allow greater control of content distribution, system configuration, and software maintenance than client-based systems. In addition, server-based systems, particularly Web-based systems, offer greater accessibility to users. Finally, a server-based system has the benefit of immediate data collection: teachers and researchers have access to the same information in real-time. Unfortunately, server-based systems face scalability problems. The hybrid design we are currently exploring retains many of the advantages of server-based systems while addressing these scalability concerns. The evolution of our system and the result of this research is useful for the entire ITS community. Sharing the results of our work with other ITS researchers will help expand the development of server-based tutoring systems.

Acknowledgments

We would like to acknowledge the Department of Education, National Science Foundation, Office of Naval Research, and the Worcester Public School System for providing the means to accomplish this research.

References

1. Bacic, B.: Constructing intelligent tutoring systems: design guidelines. In: Information Technology Interfaces, 2002. ITI 2002. Proceedings of the 24th International Conference on Information Technology Interfaces, vol. 1, pp. 129–134 (2002)
2. Brusilovsky, P., Peylo, C.: Adaptive and intelligent web-based educational systems. J. Artif. Intell. Educ. 13(2-4), 159–172 (2003)
3. Patvarczki, J., Almeida, S.F., Beck, J.E., Heffernan, N.T.: Lessons learned from scaling up a web-based intelligent tutoring system. Technical Report WPI-CS-TR-08-02, Worcester Polytechnic Institute (2008)
4. Ritter, S., Koedinger, K.R.: An architecture for plug-in tutor agents. J. Artif. Intell. Educ. 7(3-4), 315–347 (1996)
5. Stern, M., Woolf, B., Kurose, J.: Intelligence on the web. In: 8th World Conference of the AIED Society (1997)
6. Suthers, D.: Representational support for collaborative inquiry. In: Proceedings of the 32nd Annual Hawaii International Conference on System Sciences, 1999. HICSS-32, p. 14 (1999)

Tailoring of Feedback in Web-Based Learning: The Role of Response Certitude in the Assessment

Ekaterina Vasilyeva[1,2], Mykola Pechenizkiy[2], and Paul De Bra[2]

[1] Department of Computer Science and Information Systems, University of Jyväskylä, P.O. Box 35, 40351 Jyväskylä, Finland
[2] Department of Mathematics and Computer Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
{e.vasilyeva,m.pechenizkiy}@tue.nl, debra@win.tue.nl

Abstract. This paper analyzes the challenges of tailoring feedback to the student's response certitude during the assessment in Web-based Learning systems (WBLSs). We present the summary of the results of a series of experiments related to the online assessment of students through multiple-choice quizzes, where students had to select the confidence level and were able to request different kinds of feedback for each of the answered questions.[1]

Keywords: online assessment, response certitude, feedback personalization.

1 Introduction

Online assessment becomes an important component of modern education complementing traditional methods of (self-)evaluation of the student's performance.

Feedback is usually a significant part of the assessment as students need to be informed about the results of their (current and/or overall) performance. The existing great variety of the feedback functions and types that the system can actually support make the authoring and design of the feedback in e-learning rather complicated [4]. Another important issue is that different types of feedback can be differently effective up to having negative influence on the learning and interaction processes [1].

Feedback personalization becomes a challenging perspective for the development of feedback in the assessment components of WBLSs as it is aimed to provide a student with the feedback that is most suitable and useful for his/her personality and the performed task [4]. In this work we study how the feedback in online assessments can be personalized to the student's *response certitude* (*confidence* or *certainty*) that specifies her certainty in the answer and helps in understanding the learning behavior.

The traditional scheme of multiple-choice tests evaluation, where the responses are being treated as absolutely correct or absolutely wrong, ignores the obvious situations when the correct response can be the result of the random or an intuitive guess and luck, and the incorrect answer can be given as due to the careless mistake as due to some misconceptions a student may have.

[1] An extended version of this paper can be found at http://wwwis.win.tue.nl/~debra/its08/

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 771–773, 2008.

The use of feedback in certitude-based assessment in traditional education has been actively researched for over 30 years [2][3]. In spite of this intensive research, the methods and guidelines for designing and implementing feedback in confidence-based assessment remain scarce so far. In this work we discuss the results of a series of experiments related to the online assessment of students through multiple-choice quizzes, where students had to select the confidence level and were able to request different kinds of feedback for each of the answered questions.

2 Method and Results

The data for this study were collected from seven online multiple-choice tests (partial exams or mandatory individual exercises for three different courses). Each question (answered strictly one after another) in a test was accompanied by the compulsory question about response confidence that affected the grade[2] (2 points for a HCCR, 1 point for a LCCR, -1 point for a HCWR, and, 0 for a LCWR). After giving the response (with a specified certainty) the student could either go directly to the next question or request immediate (KCR/KR/EF) and delayed (EF) feedback. Students could optionally answer to the questions whether EF was useful or not.

Types of feedback requested. Most of the students were eager to get KCR and/or KR. There were usually only a few students who did not check their answers for most of the questions in each test. In two tests we analyzed whether the students were eager to only KR feedback or also KCR+EF by separating these possibilities. In this scenario more students requested only the KR, without KCR+EF. In another test where KCR/KR could only obtained from studying the EF students did request the EF to extract the KCR/KR.

After students knew whether their answer was correct they tended not to request any EF (and this was independent of the response certitude). For incorrect responses, especially for the HCWRs, the frequency of "ignoring" the EF was lowest.

In two tests we experimented with EF recommendation (based on students' learning style, response correctness and certitude) when a few different types of EF for a question were available. Corresponding results can be found in [5].

Time used for examining feedback. Students were spending less time on average for reading EF when giving HCCRs vs. LCCRs, and more time for reading EF for HCWRs vs. LCWRs (this would not be the case if the students were simply not careful with HCRWs). Having a misconception, it takes more time to understand the problem and "patch" the knowledge of certain concepts rather than simply get an understanding of some concept having no strong (incorrect) opinion about it before.

Students with many LCWRs were interested much less in the EF and more often spend just a few seconds for scanning through the explanations.

Effectiveness and usefulness of the immediate EF. The corresponding grades of the students were sufficiently higher in those cases, when the EF for the preceding related

[2] The results have shown that students were able to estimate the level of the confidence in their answers reasonably well. Used acronyms: H(L)CC(W)R – high (low) confidence correct (wrong) response; K(C)R – knowledge of (correct) response; EF – elaborated feedback.

questions was examined. This positive effect is due to the facts that EF helped many students *to answer correctly* the forthcoming related questions, and to choose appropriately low (i.e. if EF could not help to fix the knowledge problem it was still useful to choose the "correct" certainty for the answers) or high (i.e. EL indeed helped to fix the problem or confirmed the correctness of students thinking) certainty.

3 Conclusions

Designing and authoring feedback and tailoring it to students is an important problem of the online learning assessment. In this paper we addressed this problem focusing on the issues of the response certitude and the response correctness, in particular studying how they affect (1) the types of feedback the students preferred to request; (2) time the students used for examining the feedback; and (3) effectiveness of the immediate EF on the overall performance of the students during the test.

The obtained results confirmed our expectations regarding the main functions that EF may play in the online assessment depending on the combination of correctness and certitude: (1) "patching" the student's knowledge, (2) filling the gaps in the student's knowledge, and, (3) simply providing KR and KCR information.

The results strongly suggest that (1) students are able to estimate the certainty of their responses fairly well, (2) knowledge of response certitude together with response correctness allows determining what kind of feedback is more preferable and more effective for the students, and (3) elaborated feedback may sufficiently improve the performance of students within the online tests.

Concluding the stated above, the results obtained in our study strongly advocate the benefits and necessity of evaluation of the response certitude during the online assessment, and reveal the additional possibilities of feedback personalization [6].

Acknowledgments. This research is partly supported by the COMAS Graduate School of the University of Jyväskylä, Finland.

References

1. Hatie, J., Timperley, H.: The power of feedback. J. Review of Educational Research 87(1), 81–112 (2007)
2. Kulhavy, R.W., Stock, W.A.: Feedback in written instruction: The place of response certitude. Educational Psychology Review 1(4), 279–308 (1989)
3. Mory, E.H.: Feedback research review. In: Jonassen, D. (ed.) Handbook of research on educational communications and technology, Mahwah, NJ, pp. 745–783 (2004)
4. Vasilyeva, E., Puuronen, S., Pechenizkiy, M., Räsänen, P.: Feedback adaptation in web-based learning systems. Special Issue of Int. J. of Continuing Engineering Education and Life-Long Learning 17(4-5), 337–357 (2007)
5. Vasilyeva, E., De Bra, P., Pechenizkiy, M., Puuronen, S.: Tailoring feedback in online assessment: influence of learning styles on the feedback preferences and elaborated feedback effectiveness. In: Proc. of 8th IEEE ICALT 2008 (to appear, 2008)
6. Vasilyeva, E., Pechenizkiy, M., De Bra, P.: Adaptation of Elaborated Feedback in e-Learning. In: Proc. of Int. Conf. on Adaptive Hypermedia AH 2008 (to appear, 2008)

Trying to Reduce Bottom-Out Hinting: Will Telling Student How Many Hints They Have Left Help?

Yu Guo, Joseph E. Beck, and Neil T.Heffernan

Wocester Polytechnic Insititute, Department of Computer Science
100 Institute Road, Worcester, MA 01609
phone: 508-831-5006
{yuguo,nth}@wpi.edu, joseph.beck@gmail.com

Abstract. Many model-tracing intelligent tutoring systems give, upon demand, a series of hints until they reach the bottom-out hint that tells them exactly what to do (exact answer of the question). Since students don't know the number of hints available for a given question, some students might be surprised to, all of a sudden, be told the final answer; letting them know when the bottom out hint is getting close should help cut down on the incidence of bottom-out hinting. We were interested in creating an intervention that would reduce the chance that a student would ask for the bottom out hint. Our intervention was straightforward; we simply told them the number of hints they had not yet seen so that they could see they were getting close to the bottom out hint. We conducted a randomized controlled experiment where we randomly assigned classrooms to conditions. Contrary to what we expected, our intervention led to more, not less, bottom out hinting. We conclude that the many intelligent tutoring systems that give hints in this manner should not consider this intuitively appealing idea.

1 Introduction

Providing help to students when they get stuck in the learning process is a pedagogical strategy in Intelligent Tutoring System (ITS). Showing student hints is a popular way of guiding students in most tutoring system. The Andes Physics Tutoring System gives learners different levels of hints on demand in their process of learning [1]. Cognitive tutor also gives hints on demand [2].Our system ASSISTment also provides, for each question, some unspecified number of hints, which eventually will terminate with the student being told exactly what to do. However, none of them informs students when they will get the bottom-out hint. Our hypothesis is that students might not want to be told the exact answer, but in the current systems they can never be sure if the next time they ask for a hint, they will get told the answer. We want the student to think over the problem before he goes through all the hints to the bottom one. In other words, the intelligent tutoring system needs to prevent the student from "gaming the system" [3] by cheating the system to reach a correct answer. Others have also investigated ways to reduce students' gaming behavior ([4, 5]).The goal of this research is to find if telling students how many hints there are left in the problem will lead students to less often reach bottom-out hints.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 774–778, 2008.

We carried out the experiment on ASSISTment, a web-based tutoring system developed by Worcester Polytechnic Institute. It assists student learning math by providing individual help in the manner of giving a sequence of hints and scaffolding questions while assessing student. Teachers are able to keep track of students' performance in the system through several up-to-date reports generated by the system. Content builders are provided the tool to build main questions, scaffolding questions, hints, answers, buggy messages and so on so forth in a convenient way. The system is freely available at www.assistment.org. ASSISTment is now used by thousands of kids and teachers as a normal part of their classes. We changed the system's help request interface for this experiment and built a logistic regression model to analyze the data.

2 Experiment

Our approach is as following. Instead of showing the button labeled with "Request Help," we count how many hints are left in the problem for a specific student and label the help button as "Show me hint <N1> out of <N2>". N2 equals total hints of the problem and N1 means how many hints the student has asked if he clicks on the button. When student is about to reach the bottom hint, we will label the button as "Show me the last hint", explicitly telling them this hint will show the exact answer of the question. We collected data from 2007/9/30 to 2007/11/29. Roughly halfway through this time period, on 10/29, we turned this intervention "on" in half the classrooms, leaving the other classrooms to act as a control to control for any existing trends over time in student behavior. For each teacher in our system, we randomly selected half of the classrooms to be in the experiment, while the other classes served as the control. We had 320 students that did more than 20 problems both before and after 10/29 with 88 students in experiment group and 232 students in control group. We filtered the data by 20 problems for the reason that we wanted to focus on the behavior of consistent users of ASSISTment during the time period.

The data was collected on question level. Each row includes the information about how one specific student performs on one particular problem. We collected the information of "student_id", "hit_bottom" (whether the student reaches bottom-out hint in this problem), "easiness" (easiness of problem which is measured by correctness over all the students in the system), condition (whether the student in experiment group or control group), "beforeEx" (whether this problem is done pre-intervention or post-intervention). The reason we collected data in this manner is that in our expectation, whether the student will hit the bottom-out hint relates to the problem easiness level, student himself, which condition the student is in and when this problem is done. There are more than 37,000 rows in our dataset.

3 Result and Analysis

We built a logistic regression model to predict dependent variable "hit_bottom" with factors "student_id", "condition", "beforeEx" and covariate "easiness". Since these are elements affecting students' bottom-out hinting behavior, we consider them as main effects in our model and moreover, we take beforeEx * condition into account as an interaction effect.

The parameter estimates of the model are shown in Table 1. The higher the β value is, the less likely the student will hit the bottom-out hint. The fact that the β value for control group is 0.280 and with P=0.547 suggests that there is no significant change on students hinting behavior whether he is in control group or experiment group when ignoring the time period. The fact the β value for "After Intervention" is -0.092 with P =0.151 tells us that students hit more bottom-out hints after the intervention. However, this is not statistically reliable. We can not get conclusion related to the effect of intervention by simply looking at these two parameters, which is the reason that we take the interaction of these two parameters into consideration. The Easiness parameter of +5.441 suggests that there is more bottom out hinting on harder items, which is to be expected. Finally, and most interesting, the interaction between beforeEx*condition, is the crucial parameter that allows us to see if there is a change in student behavior due to the experiment, after taking into account all the other parameters in the model. The fact the parameter for [Condition=C]*[beforeEx=0] is 0.310 indicates that the experiment caused more bottom out hinting, contrary to our hypothesis. This difference was statistically reliable.

Table 1. Parameter Estimates

	β	Sig.
Control Group(Condition=C)	0.280	0.547
Experiment(Condition=E)	0*	-
After Intervention(beforeEx=0)	-0.092	0.151
Before Intervention(beforeEx=1)	0*	-
Control Group and After Intervention([Condition=C]*[beforeEx=0])	0.310	<0.0002
Control Group and Before Intervention([Condition=C]*[beforeEx=1])	0*	-
Experiment Group and After Intervention([Condition=E]*[beforeEx=0])	0*	-
Control Group and After Intervention([Condition=E]*[beforeEx=1])	0*	-
Easiness	5.441	<0.00001

*This parameter is set to zero because it is redundant.

For further analysis, we divided the students into three groups with high, medium and low math proficiency according to the student proficiency parameter estimated by the one-parameter Item Response Theory model (Rasch model)[1]. The Rasch model was trained based on data collected in ASSISTment system from Sept. 2004 to Jan. 2008 that includes 14273 students' responses to 2796 main questions [6]. Since the intervention might have different effect on distinct population, the analysis in the view of student's proficiency came next.

We built a logistic regression model for each group. We will put our main focus on the interaction effect. In Table2, we show the β values for interaction effect in each

[1] We won't go further into "Item Response Theory model" in this paper. Basically, it gives us a statistically reliable estimate of student's math proficiency based on their entire responses in the interaction with ASSISTment.

group. The β value of interaction is 0.274 with P=0.013 indicates that for the population with low math proficiency, the intervention led to more bottom-out hints. It is the same case as population with medium math proficiency, giving β value equals 0.477 and P value less than 0.002. Moreover, the effects are statistically reliable. However, for population with high math proficiency, the intervention has no statistically reliable effect.

Table 2. Parameter Estimates for three groups

[Condition=C]*[beforeEx=0]	β	Sig.
Low proficiency group	0.274	0.013
Medium proficiency group	0.477	<0.002
High proficiency group	0.061	0.777

4 Conclusion and Future Work

This experiment shows that telling student how many hints they have left affects student behavior regarding the bottom-out hint. The logistic regression shows that the change will in general lead to **more** bottom-out hints, contrary to our hypothesis. Further analysis in terms of different population indicates that the intervention will lead to **more** bottom-out hints in the population with low and medium math proficiency while has no statistically reliable effect on the students with high proficiency. According to the results, telling students how many hints left is not a suggestion that other tutoring systems designers should implement. One explanation of this result is that students might not believe that asking for hints, or getting the bottom out hint, hurts their learning. For those students who are aimed at "gaming the system," showing the number of remaining hints might assist or encourage their gaming behavior.

We will continue working on how to prevent students from asking for more help than they probably need in the learning process. As it has been stated that students might not consider that asking for more help than they need hurts their learning, an intervention as following seems to be a potential strategy. For instance, we can provide a partial credit for each question with some initial value and show it to the student. Whenever student asks for help, we will take some points off the credit. Hopefully, this intervention would to some extent guide students to contribute more efforts to get the question done correctly by themselves or based on the help they've already attained from the system. Moreover, we will focus on analyzing if students gain more learning through the intervention, which is an important metric of intelligent tutoring systems.

Acknowledgement. We would like to thank Worcester Public Schools and all of the people associated with developing the ASSISTment system listed at www.ASSISTment.org who made this experiment possible. We would also like to acknowledge funding from the US Department of Education, the National Science Foundation, the Office of Naval Research and the Spencer Foundation. All of the opinions expressed in this paper are those solely of the authors and not those of our funding organizations.

References

1. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The Andes Physics Tutoring System: Lessons Learned. International Journal of Artificial Intelligence in Education 15(3), 1–47 (2005)
2. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.: Intelligent tutoring goes to school in the big city. International Journal of Artificial Intelligence in Education 8, 30–43 (1997)
3. Baker, R.: Is Gaming the System State-or-Trait? Educational Data Mining Through the Multi-Contextual Application of a Validated Behavioral Model. In: Workshop on Data Mining for User Modeling, Educational Data Mining Track, at User Modeling, pp. 76–80 (2007)
4. Walonoski, J., Heffernan, N.T.: Detection and analysis of off-task gaming behavior in intelligent tutoring systems. In: Ikeda, Ashley, Chan (eds.) Proceedings of the Eight International Conference on Intelligent Tutoring Systems, pp. 382–391 (2006a)
5. Lloyd, N., Heffernan, N.T., Ruiz, C.: Predicting student engagement in intelligent tutoring systems using teacher expert knowledge. In: The Educational Data Mining Workshop held at the 13th Conference on Artificial Intelligence in Education, pp. 40–49 (2007)
6. Feng, M., Beck, J., Heffernan, N.T., Koedinger, K.R.: Can an Intelligent Tutoring System Predict Math Proficiency as Well as a Standardized Test? In: The 1st International Annual Conference on Education Data Mining, Montreal (DRAFT) (submitted, 2008)

Leveraging C-Rater's Automated Scoring Capability for Providing Instructional Feedback for Short Constructed Responses

Jana Sukkarieh and Eleanor Bolge

Educational Testing Service,
Princeton NJ, 08541, USA
http://www.ets.org

Abstract. Due to some progress on the natural language processing (NLP) front, researchers are able to pursue the problem of automatic content assessment for free text responses with some success. In particular, a concept-based scoring method implemented in c-rater, Educational Testing Service's (ETS) technology for content scoring of short free-text answers makes c-rater capable of giving instantaneous formative individualized feedback without going fully into a dialog-based system nor restricting itself to just canned hints and corrective prompts.

1 Introduction

In the last few years, a keen interest in automatic content scoring of constructed response items has emerged. Progress in NLP has made it possible to 'judge' content without having to fully understand the text. Several systems for content scoring exist. However, the only four systems that deal with both **short answers** and **analytic-based content** (i.e. look at content in terms of the main points or concepts expected in an answer) are Automark at Intelligent Assessment Technologies [2] , c-rater at ETS [1], the Oxford-UCLES system at the University of Oxford [5] and CarmelTC at Carnegie Mellon University [4]. Though the first 3 systems were developed independently, their first versions worked very similarly using a knowledge-engineered IE approach taking advantage of shallow linguistic features that ensure robustness against noisy data (students' answers are full of misspellings and grammatical errors). Later on, OXFORD-UCLES experimented with machine learning techniques similar to the ones in CarmelTC. Though these latter techniques proved very promising in categorizing students' answers into classes (corresponding to the main points expected in an answer - or none of the concepts), we believe the models of such techniques are not transparent enough in order to devise individualized feedback except probably a canned set of hints and prompts corresponding to a certain main point. This year we have changed the way c-rater scores students' answers in order to increase the accuracy in tracking down the concepts that students get wrong or right. This allow us to give more individualized formative feedback for each answer c-rater scores without having to go into a full dialog-based system yet not restrict ourselves just to canned hints and prompts.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 779–783, 2008.

2 Current C-Rater in a Nutshell

In Table 1 please see examples of items that c-rater deals with. On the left, the prompt or the question is listed while on the right there is a list of main/key points or concepts (We use these words interchangeably). A non-ambiguous scoring guide for the item is also provided. The number, N, is the number of concepts that examiners are expecting in a student's answer and C_i is the ith concept.

Table 1. Sample items in Biology and Reading Comprehension

Statement of the item	Rubric
Prompt 1. Full Credit is 2	**Concepts: 6**
Identify TWO common ways the body maintains homeostasis during exercise	*C1:cellular respiration ; C2:increased breathing rate; C3:decreased digestion;C4:sweating C5:dilation of blood vessels;C6:increased circulation rate*
Scoring Guide	2 points for 2 or more elements 1 point for 1 element else 0
Prompt 2. Full credit is 3	**Concepts: 24**
Identify three ways in which an animal's body can respond to an invading pathogen.	*C1:temperature change or fever; C2:water loss; C3:production of more mucous;* 21 other concepts that we will not list here
Scoring Guide	1 point for each element
Prompt 3. Full credit is 2 (a reading is given) Explain what you think the delegates may be trying to persuade the Native Americans to believe or to do. Then, name an example of how the delegates attempt to persuade through their speech.	**Concepts: 11** *C1:to understand the conflict with England; C2:to take their side against England; C3:to appeal to the Native Americans; C4:by calling them as brothers; C5:use authoritative language;* 6 other concepts
Scoring Guide	1 point for 'what' concept 1 point for 'how' concept

In 2007, we started viewing c-rater's task as a textual entailment problem (TE). We use TE here to mean either a paraphrase or an inference up to the context of the item. c-rater's task is reduced to a TE problem in the following way:

> **Given:** a concept, C, (for example, *"body increases its temperature"*) **and** a student answer, A, (for example either *"the body raise temperature"*, *"the bdy responded. His temperature was* 37° *and now it is* 38°*"* or *"Max has a fever"*) **and** the context of the item **the aim is** to check whether C is an inference or a paraphrase of A (in other words A implies C and A is true)

To solve this problem, the following is undertaken. A human writes a set of model answers (in terms of a set of model sentences) by looking at a sample of manually annotated students' answers. The annotation consists of finding, for each concept C, evidence E in an answer such that C is a TE of E (if such evidence exists). Hence, we call this way of model building concept-based and consequently we call this method concept-based scoring. The size of the training sample varies depending on the linguistic complexity of the concepts expected in an answer. An unseen answer is then compared to a model answer. The current comparison module, Goldmap (version implemented in 2006 and 2007), is based on maximum entropy modeling [3]. Basically, given a set of attributes or

features, constraints over these attributes and a set of instances or training data consisting of pairs of sentences that are both manually-annotated for match or no-match and automatically-annotated according to the set of attributes, Goldmap learns a matching model. Given an unseen answer, the matching model outputs a probability on the match between the unseen answer and a model answer. In general, we use a threshold of 0.5 to determine a match. Scoring rules are then applied to obtain a score.

The attributes in Goldmap depend on a set of linguistic features that are obtained by various NLP modules in c-rater. These attributes comprise some syntactic variations like active/passive, morphological variations like went/go/gone, semantic variations depending solely on synonyms and detection of negation of heads of noun phrases, verb phrases and adjective phrases.

A student answer is processed as follows. Consider an answer like *the body could raise tempratuer. John sweats.* This gets split into two sentences then each sentence is taken as a separate input. Then, *the body raise tempratuer* goes through a spelling corrector (this is one way to minimize noise for subsequent NLP). Once words in the sentence are corrected, the sentence goes through a part-of-speech tagger and a parser to output a parse tree (OpenNLP parser, Baldridge and Morton http://opennlp.sourceforge.net/). The OpenNLP parser attempts to output a deep constituent parse tree. A preliminary evaluation of the parser revealed that it is robust enough towards misspellings and ungrammaticality. Risking losing some information from a parse tree yet being able to add some dependency features, a parse is reduced to a flat structure representing phrasal chunks annotated with some syntactic and semantic roles. The structure also indicates the links between various chunks and distributes links when necessary. For example, if there is a conjunction, a link is established. The next stage is an attempt to resolve pronouns and definite descriptions. Hence, *the body* is resolved to *an animal's body* (this appears in the prompt of the item). Currently, each Goldmap attribute is an aggregate of various features obtained from the above process[1]. We tested our current implementation on items 3 and 4 in Table 1 with 24 and 11 concepts, respectively, with 500 answers as training data and 500 as blind data for each. We asked two human raters to annotate and score the data according to the concepts and we implemented c-rater's model answer building process to be driven by these annotations as explained earlier. The results were very promising with an average unweighted kappa agreement of c-rater with human raters of 0.71 vs human-human agreement of 0.76 for the Biology item and 0.54 c-rater with humans vs 0.69 human-human kappa agreement for the Reading.

The motivation behind concept-based scoring has many aspects. The most important is that we expect better accuracy with which the matching algorithm decides about whether Concept C is a TE of Answer A since it is learning from a much more accurate set of linguistic features about the TE task than it does without this correspondence. Furthermore, the features obtained from the various NLP

[1] We leave the description of a detailed process and the inclusion of the context of the item for another occasion.

components and grouped in various Goldmap attributes form powerful linguistic information. Consequently, we expect a more informed technique to give individualized feedback for students. Also, a concept-based scoring allows us to put more **weight** on one concept than another, if necessary, depending on what a tutor believes should be concentrated on in a student's knowledge at one particular time. In the following, we show how c-rater can be seen as a kernel for an interactive tutorial tool.

3 C-Rater for Learning

In the past, c-rater's feedback consisted merely of a message indicating *right or wrong*. Currently, enhanced by concept-based scoring and a linguistically-driven learning algorithm, c-rater gives quality feedback indicating to the students which concepts in their answers they get right and which concept they get wrong with the capability of scaffolding additional individualized questions, hints and feedback. When students get an answer or a concept wrong c-rater has different modes to choose from (to give hints, questions or feedback) depending on the application at hand, the grade level and the difficulty of the item. The following cases occur:

- ☞ A concept that is partially wrong. There are several key elements and the student gets all except one of them correct and that one s/he got it partially right.
  - **Scenario 1:** Assume one student enters 5 out of 6 correct elements in item 1 and for the 6th element s/he enters a partially-right answer. c-rater prompts her/him with the correct parts and acknowledges the partially correct part while correcting the part that is not correct by providing canned tutoring session on the "missing" concept.
  - **Scenario 2:** Assume a student gives an answer like *increased digestion* for *decreased digestion*. In that case, c-rater tells the student that *increased digestion* does the opposite of what the body needs to do and asks the student to try again. Instead of giving the right answer, the idea is to give a hint that is most specific and suitable for the answer that the student provided, e.g., if for the same item, the student writes *the digestive process changes* then c-rater's prompt would be either *give a qualification for that change* or simply *changes how?*.
- ☞ A particular concept is completely wrong. There are two feedback modes in c-rater.
  1. c-rater just provides the correct concept(s) or
  2. c-rater gives hints to the student that are specific and most suitable for the answer and ask him/her to try again (see 2(a) below)
- ☞ All that the student enters is wrong. Again, there are two feedback modes in c-rater.

1. c-rater simply lists the right concepts or key elements
2. c-rater asks scaffolded questions to the student to check whether the student understands the question or not (if a student does not understand the question then obviously s/he cannot reply), e.g., c-rater prompts the student: *do you know the definition of homeostasis?* c-rater expects a yes or no answer.
   (a) if YES then c-rater asks the student to provide the definition. c-rater then scores the answer that the student provides (treats it as another short item to score / layers of scaffolding can be introduced). If c-rater decides the answer is wrong then it provides the definition and asks the student to try the question again. If c-rater decides the student knows the definition then it starts giving the student some hints to help him/her.
   (b) if NO then c-rater provides the definition and gives the student another chance to answer [repeat process (a)].

4 Conclusion

c-rater's new way of scoring, namely, concept-based scoring allows it to give a more powerful individualized feedback on concepts expected in the space of knowledge of a student. The fact that c-rater automatically scores the content of short free-text means that introducing scaffolded prompts and scoring these prompts are in c-rater's nature. Assessment and learning go in tandem in a literal sense in c-rater. Though the ASSISTment system at CMU (www.assistment.org) is only for multiple choice and feedback cannot be individualized in the same way c-rater's feedback can, their experience could be very valuable for us. The experience Autotutor group (http://www.autotutor.org/) could be very valuable for us too. Finally, we would like to conduct a more comprehensive evaluation for concept-based scoring and linguistically-driven feedback before making any wide claims.

Acknowledgments

We would like to thank Tom Morton for his initial work on Goldmap.

References

1. Leacock, C., Chodorow, M.: C-rater: Automated Scoring of Short-Answer Questions. Computers and Humanities 37(4) (2003)
2. Mitchell, T., Russel, T., Broomhead, P., Aldrige, N.: Towards robust computerised marking of free-text responses. In: Proceedings of the 6th Computer Assisted Assessment Conference (2002)
3. Ratnaparkhi, A.: Maximum Entropy Models for Natural Language Ambiguity Resolution. Ph.D. thesis, University of Pennsylvania, Philadelphia, USA (1998)
4. Rosé, C.P., Roque, A., Bhembe, D., VanLehn, K.: A hybrid text classification approach for analysis of student essays. Building Educational Applications Using NLP (2003)
5. Sukkarieh, J.Z., Pulman, S.G., Raikes, N.: Auto-marking: using computational linguistics to score short, free text responses. In: 29th IAEA (2003)

An Authoring Tool That Facilitates the Rapid Development of Dialogue Agents for Intelligent Tutoring Systems

Yue Cui[1] and Carolyn Penstein Rosé[1,2]

[1] Language Technologies Institute
[2] Human-Computer Interaction Institute, Carnegie Mellon University,
5000 Forbes Avenue, Pittsburgh, PA 15213
{ycui,cprose}@cs.cmu.edu

Abstract. Natural language dialogues have been demonstrated to be effective for instruction in a variety of settings. However, previous research has demonstrated that the hierarchical structure of dialogue, which is a natural representation for linguists, is difficult for non-linguists to understand and work with. Even for linguists, it is possible to lose track of where one is in this hierarchical structure during authoring. We refer to difficulty in working with this hierarchical representation as the hierarchical navigation problem. In this short paper we present a simple but powerful tool[1] designed to facilitate the authoring process, which includes innovative interface elements that specifically target the hierarchical navigation problem. Our user study demonstrates progress towards overcoming this difficulty.

1 User Study

In the past decade, tutorial dialogue has become a well established mode of instruction within the intelligent tutoring community for supporting learning with technology. The dynamic nature of dialogue enables dialogue to be tailored to the specific needs of students in a highly flexible way.

Nevertheless, authoring dialogue agents is not trivial. The dynamic and adaptive nature of dialogue arises from a relatively complex, hierarchical structure that lies beneath the deceptively simple appearance of a dialogue from the surface. Beyond the complexity of the dialogue phenomena itself, authoring dialogue agents comes with technical challenges as well. Many dialogue systems provide modules to support the development of dialogues [1,2]. However, they do not provide users who have little or no programming expertise an easy and effective way of authoring complex dialogue behavior. In order to address this problem, prior research in the intelligent tutoring community has produced authoring environments capable of shielding authors from the underlying technical details of the dialogue representations and machinery they are authoring by means of graphical interfaces [3,4]. Nevertheless, in these environments, authors must still specify

[1] This research was supported by Office of Naval Research, Cognitive and Neural Sciences Division, grant N00014-05-1-0043.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 784–786, 2008.

the hierarchical structure of the dialogues they are creating. Because of this, non-linguists struggle to keep from getting lost in the complexity, even in graphical environments meant to simplify the task for them. Because the hierarchical structure of dialogue is essential for facilitating the dynamic and adaptive nature that makes it so valuable, an important goal for this community is to develop authoring environments that are capable of supporting authors in navigating this complex structure without getting lost in the process.

The important contribution of this work with respect to the design is the support for navigation within the hierarchical representation of dialogue behavior. There are three main parts in the Tutalk authoring tool: the Author panel, the Preview panel and the Test panel. The Author panel is the main panel that authors use to build the dialogue agents. The Preview panel provides a way to preview the dialogues that will be generated by the agent and check for errors in the dialogues. The Test panel actually runs the dialogues and provide a platform where users can interact with the dialogue agents they have created. The navigation difficulties we address in this paper mainly relate to the Authoring panel. The navigation support we have built in works similarly to navigation within a browser window so that links to subdialogues take the author to a page that focuses on the indicated subdialogue, and a back button pops the user back to the higher level of representation. During the process no new windows are opened. The author simply navigates the structure within the same window. In this way, the author can go very deep in the embedded sub-dialogues and easily return back to the main dialogue using the back button as in a web browser. As simple as this approach appears, this turns out to be the most effective way of navigating in the complex hierarchical structure of the dialogues. The Preview panel is a second aspect of this navigation support as is allows authors to preview alternative flows through the authored dialogue representation in order to verify that it makes sense under all possible conditions of input from students.

In order to measure the contribution of the innovative navigation support that we added to our authoring environment, we ran a user study in which we compared the performance of participants using a version of the tool with the new interface elements with that of participants using an otherwise identical version of the tool that did not include this navigation support. The results provide a clear indication that participants are able to work more efficiently with the tool that includes navigation support.

Experimental Design. The user study was a simple 2 condition, between subjects design, which contrasted a version of the authoring environment that included the innovative navigation support (for use in the Experimental Condition) with an otherwise identical version that did not include this support (for use in the Control condition). This baseline version of the authoring environment that was used in the Control condition had been evaluated during use in a week long intelligent tutoring summer school, hosted at our institution in the summer of 2006. While participants of that summer school were able to learn to use the tool and author their own dialogues as part of their involvement in the summer school, we observed difficulty with navigation, which we seek to overcome with the navigation support we evaluate here.

Participants. During our pilot testing phase, we noticed that graduate students and undergraduate students displayed a different pattern of behavior from one another. Thus, for the formal study, we were careful to avoid having this difference bias the results we would observe. We recruited 10 graduate students and 10 undergraduate students for the

formal user study from a variety of technical and non-technical fields of study, including Fine Art, Psychology, Management, Engineering, and Science, but purposefully none from Computer Science. All of the participants were comfortable with using computers but were not experts in dialogue authoring technology. Participants from both the graduate and undergraduate subject pools were randomly assigned to the experimental or control conditions such that we balanced the distribution of undergraduate and graduate students between conditions.

Experimental procedure. The experimental procedure was composed of eight main segments, namely: (1) Questionnaire (5 minutes), (2) Training (10 minutes), (3) Pre-test (10 minutes) (4) Coached Tool use (20 minutes), (5) Informal interview (5 minutes), (6) Uncoached tool use (70 minutes split into 40 minutes for task 1 and 30 minutes for task 2), (7) Post-test (10 minutes), and (8) Evaluation questionnaire.

Results. We evaluated whether participants in the two conditions differed with respect to their success as completing task 1 and/or task 2 without errors within the allotted time. In both cases, the trend was in favor of the Experimental condition, however it was only significant in the case of task 1, based on a binary logistic regression analysis. For task one, 90% of participants in the Experimental condition were able to complete Task 1 without errors, whereas only 30% of Control condition subjects were ($p < .005$). For task 2, 70% of Experimental condition participants were able to complete the task perfectly within the allotted time, whereas only 50% of the Control condition participants were. Results from the questionnaire further supported the finding that authoring with navigation support was more effective.

References

1. Larsson, S., Traum, D.: Information state and dialogue management in the trindi dialogue move engine toolkit. Natural Language Engineering Special Issue on Best Practice in Spoken Language Dialogue Systems Engineering, 323–340 (2000)
2. Walker, M., Rudnicky, A., Prasad, R., Aberdeen, J., Bratt, E., Garofolo, J., Hastie, H., Le, A., Pellom, B., Potamianos, A., Passonneau, R., Roukos, S., Sanders, G., Seneff, S., Stallard, D.: DARPA communicator: Cross-system results of the 2001 evaluation. In: Proceedings of International Conference on Spoken Language Processing (ICSLP) (2002)
3. Gweon, G., Arguello, J., Pai, C., Carey, R., Zaiss, Z., Rose, C.P.: Towards a Prototyping Tool for Behavior Oriented Authoring of Conversational Interfaces. In: Proceedings of the ACL Workshop on Educational Applications of NLP (2005)
4. Jordan, P.W., Hall, B., Ringenberg, M., Cui, Y., Rose, C.P.: Tools for Authoring a Dialogue Agent that Participates in Learning Studies. In: Proceedings of AIED 2007, pp. 43–50 (2007)

Using an Emotional Intelligent Agent to Reduce Resistance to Change

Ilusca Lima Lopes de Menezes and Claude Frasson

HERON Laboratory, Computer Science Department, University of Montréal
CP 6128 succ. Centre Ville, Montréal, QC, Canada. H3C 3J7
{lopesdei,frasson}@iro.umontreal.ca

Abstract. Organizations need to change for developing and surviving in the marketplace, but changes bring uncertainty and not only produce feelings of apprehension, anxiety and stress, but also create resistance among employees. Resistance can delay and increase the costs of a change process. So, it is essential to present organizational changes in a manner that reduce the potential for resistance. In this paper, we propose an intelligent agent able to reduce employee's resistance to change by presenting the best corrective emotional strategy according to his personality in order to temper his negative emotion.

1 Introduction

Organizational changes are defined as one or more efforts to transform structure, goals, technology or tasks of a company [1]. Although the organizations need to change for developing and surviving in the marketplace, a change brings uncertainty and evokes apprehension, anxiety and stress. Face a change, employees may be afraid of new situations and can have a belief that change will have a negative impact on them and so they show a resistance. Several studies on resistance to change have considered that negative emotions are an indicator of an individual's resistance [2]. Moreover, resistance can block and increase the costs of a change process [3]. So, it is crucial to present changes in a manner that reduces the potential for resistance [4].

In this present work, we propose to reduce employee's resistance to change using an intelligent agent able to apply a *corrective emotional strategy* according to various factors including the personality of the employee. First, we present the architecture of our agent and then the experiment that we have set up to determine the best corrective emotional strategy. Finally, we discuss the evolution and perspective of our next works.

2 The EMIARC Architecture

In order to reduce employees resistance to change, we created an *Em*otionally *I*ntelligent *A*gent for *R*esistance to *C*hange (EMIARC) aiming to propose the *best corrective strategy* to be applied to an employee subject to a change process. This strategy is defined as the strategy that will minimize the emotional impact of change. Figure 1 illustrates the EMIARC architecture composed of 3 main modules.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 787–789, 2008.

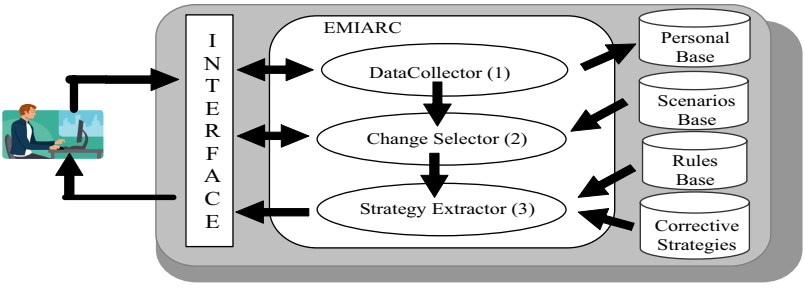


Fig. 1. Architecture of EMIARC

The agent is an autonomous system which interacts with the employee who is faced to a specific change in his environment. This agent will progressively acquire expertise about the adequate corrective strategy to apply by experimenting different cases. This expertise will be stored in the rules base which is initially empty; the rules will be created by successive runs of the agent on specific change situations and different employees. EMIARC will run according to two successive phases. In the first phase it will acquire data (training data) then, in a second phase, he will be able to send the corrective strategy directly to the employee or to his manager. The different steps of EMIARC intervention are the following:

1. The first module (Data Collector) acquires the employee's personality traits by sending to him the personality questionnaire called Saucier's Big-five Mini-Markers [5].

2. After collecting personal data, the second module (Change Selector) extracts a scenario of organizational change from the base of scenarios and presents it to the employee. This base contains various types of scenarios which are credible stories about the future changes in an organization. Presented to the employee, this scenario will trigger successively an *emotional reaction* and a *behavioural reaction*. If the emotional reaction is negative then the behavioural reaction will consist in resistance to change. Moreover, the type of reaction will vary according to the employee's personality. The emotional reaction and the behavioral reaction of the employees are obtained by asking them to imagine their emotional state and indicate their behavioral intentions towards the change.

3. Knowing the reactions of the employee, the agent aims now to minimize the employee's resistance. The third module (Strategy Extractor) will select a corrective emotional strategy that will be presented to the employee with the scenario of change in order to reduce the emotional impact. Negative emotions are due in general to a lack of explanation and fear of uncertainty. So, this strategy is a sequence of additional information giving more explanation to the change, how the employee can adapt, which compensation he will be able to get, etc. Now, our goal is to select the *best corrective strategy* which is the strategy able to reduce the most the negative emotion. We have then conducted an experiment in order to provide the agent with a mechanism able to select the best corrective strategy.

3 Experiment

We have applied EMIARC in two steps, as indicated above:

(1) acquiring personal data and emotional reactions to identified scenarios, using a machine learning technique, the Naïve Bayes Classifier [6], to determine the highest negative emotion class given the user's personality, the scenario of change, and the user's negative emotion with the highest value for this scenario, then

(2) by proposing different corrective strategies the system was able to determine (using again the same machine learning technique) their impact to reduce the emotions and consequently the best corrective strategy class. Results were stored as rules in the rules base and have shown a great reduction, and even a suppression, of the negative emotions. This experiment was implemented and tested by twenty three volunteers.

4 Conclusion and Future Work

The agent provides a double capability:

- The agent is a tool able to create a database of change scenarios specific to the company and personality traits of the employees.
- Once the acquisition phase is realized, the agent can recommend to the managers the corrective strategy to apply to reduce the impact of the change process.

Results have shown a clear impact of the agent on change management. For future research, we plan to keep running the experiment in order to obtain more training data and create more rules.

Acknowledgements. We acknowledge the support of the FQRSC (Fonds Québécois de la Recherche sur la Société et la Culture) and NSERC (National Science and Engineering Research Council) for this work.

References

1. Carnall, C.A.: Towards a theory for the evaluation of organizational change. Human Relations 39, 745–766 (1986)
2. Kiefer, T.: Feeling bad: Antecedents and consequences of negative emotions in ongoing change. Journal of Organizational Behavior 26(8), 875–897 (2005)
3. Ansoff, I.H.: Implanting Strategic Management. Prentice Hall International, London (1990)
4. Jex, S.M.: Organizational Psychology. Wiley & Sons, New York (2002)
5. Saucier, G.: Mini-markers: A brief version of Goldberg's unipolar Big-Five markers. Journal of Personality Assessment 63, 506–516 (1994)
6. Rish, I.: An empirical study of the naive Bayes classifier. In: Workshop on Empirical Methods in AI (2001)

Story Generation to Accelerate Math Problem Authoring for Practice and Assessment

Yue Cui[1], Rohit Kumar[1], Carolyn P. Rosé[1,2], and Kenneth Koedinger[2]

[1] Language Technologies Institute and
[2] Human-Computer Interaction Institute,
Carnegie Mellon University,
5000 Forbes Avenue, Pittsburgh, PA 15213
{ycui,rohitk,cprose,krk@cs.cmu.edu}

Abstract. We present a novel authoring infrastructure[1] for accelerating the rate of content creation for coached practice environments. In an initial authoring stage, the tool supports quick authoring of scaffolded problems, drawing upon principles established in prior intelligent tutoring authoring research. However, in contrast to earlier approaches, the result of this stage is a structured template than can be used to generate a multitude of variations on the same problem concept.

1 Morph Generation Tool

Coached problem solving environments employing some form of "model tracing" inspired by theories of problem solving from cognitive psychology have proven effective for learning, and have recently been used for developing assessment systems that can be used to do assessment during instruction. A variety of authoring environments have been developed to greatly accelerate the rate at which content for such systems can be authored [2,4]. These environments make development of coached problem solving environments easier and more cost-effective by making it possible for non-programmers to create the content. Nevertheless, authoring an item involves demonstrating alternative correct and incorrect problem solving steps and then annotating steps in the resulting representation with hint messages and feedback messages. Often it is desirable for students to work through several similar problems, which we refer to as "morphs", in the course of a year in order to perfect the relevant skills. Currently only rudimentary support for reusing the effort involved in authoring a single item is available. Research related to generating morphs has focused on generalizing the numbers, equations, or graphs that are included in a problem, sometimes in order to manipulate problem difficulty, but not the cover story [3]. We see this as an opportunity to build on earlier successes by using story generation technology to multiply the fruits of authoring effort by augmenting existing item authoring technology. Thus, in this poster we present an authoring infrastructure that was designed to templatize the output from such authoring environments so that the effort expended to author a single problem can produce multiple problems with an interesting variety of cover stories.

[1] This research was supported by a grant from the GE Foundation.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 790–792, 2008.

The authoring process for content creation in existing environments such as the Cognitive Tutor Authoring Tools and the Assistments Builder is a two stage process in which the texts that are presented to students in the final running system are authored directly, and then these texts are embedded in the "machinery" that transforms them into fully functioning items that can be used within coached problem solving environments. In contrast, in our approach, templates are authored that can be used to instantiate a variety of texts. And a middle stage is inserted in between the initial and final stages of the original authoring process, in which the templates are instantiated in multiple ways in order to produce a variety of cover stories. From a high level, the authoring process is illustrated in Figure 1.

Fig. 1. Morph generation process

The morph authoring interface is illustrated in Figure 2. Notice that at the simplest level, authored texts can be templatized by inserting variables that can be instantiated in a variety of ways. The template appearing in Figure 2 could be instantiated in such a way as to produce the following two texts:

[1] "Seeing a laptop for a great discount price, Fred bought it on-line. There is a discount of 25% off the price of $700 for the laptop. How much did Fred pay after the discount?"
[2] "Seeing a bicycle for a great discount price, Alice bought it at Walmart. There is a discount of 10% off the price of $100 for the bicycle. How much did Alice pay after the discount?"

This is the level of variation that can be achieved with the simplest use of the authoring environment. However, the environment includes a hierarchical planner as a control structure that allows morphs to be authored in a more flexible fashion, as in the TuTalk tutorial dialogue authoring environment [1]. For example, the template in Figure 2 contains 3 sentences. Instead of authoring the template in this way, it could have been authored as 3 separate story steps. And each of those story steps could have been specified to have a range of alternative step definitions, rather than being defined directly as texts with variables. Each of those step definitions could have then been defined as texts with variables, like the text contained in Figure 2. In that way, then a great deal more variation can be achieved. For example, for the first sentence, there may be three alternative step definitions, which include the following template texts:

[1] "Seeing a ?purchase-object for a great discount price, ?person-name bought it? purchase-location."
[2]"Let's think about the discount?person-name got on a ?purchase-object? purchase-location."
[3] "?person-name wanted to buy a ?purchase-object. Buying it ?purchase-location had the advantage of an attractive discount."

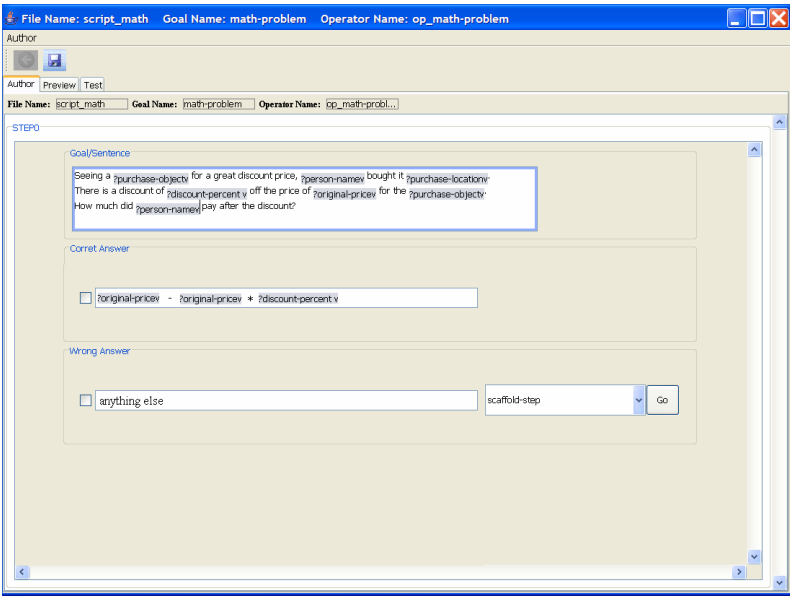


Fig. 2. Morph generator interface

By defining the morph at an even higher level of abstraction, different types of problems can be generated. For example, a slight variation would allow a problem about adding on tax rather than subtracting a discount. The same alternatives for the first sentence can be reused. Furthermore, work specifying lists of possible purchase objects, names, stores, etc., can be reused across these two story types. In pilot evaluations of our morph generation approach, we have been able to generate hundreds of alternative stories from a hierarchically structured template specification.

References

1. Jordan, P., Hall, B., Ringenberg, M., Cui, Y., Rosé, C.P.: Tools for Authoring a Dialogue Agent that Participates in Learning Studies. In: Proceedings of AI in Education (2007)
2. Koedinger, K.R., Aleven, V., Heffernan, T., McLaren, B., Hockenberry, M.: Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration. In: Proceedings of AI in Education (2004)
3. Shute, V., Graf, E.A., Hansen, E.G.: Designing Adaptive, Diagnostic Math Assessments for Individuals With and Without Visual Disabilities. ETS Research Report (2006)
4. Turner, T.E., Macasek, M.A., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K.: The Assistment Builder: A Rapid Development Tool for ITS. In: Proceedings of ITS (2005)

Supporting the Guide on the SIDE

Moonyoung Kang[1], Sourish Chaudhuri[1], Rohit Kumar[1], Yi-Chia Wang[1], Eric R. Rosé, Carolyn P. Rosé[1,2], and Yue Cui[1]

[1] Language Technologies Institute
[2] Human-Computer Interaction Institute,
Carnegie Mellon University,
5000 Forbes Avenue, Pittsburgh, PA 15213
{moonyoun,schaudhu,rohitk,yichiaw,cprose,ycui@cs.cmu.edu}

Abstract. We present SIDE[1] (the Summarization Integrated Development Environment), which is an infrastructure that facilitates the construction of reporting interfaces that support group learning facilitators in the task of getting a quick sense of the quality and effectiveness of a collaborative learning interaction. The SIDE framework offers flexibility in the specification of which conversational behavior to take note of as well as how noted behavior should be reported to instructors, making it a valuable research tool.

It is said that a facilitator of effective collaborative learning should be a "guide on the side" rather than a "sage on the stage". In classrooms, instructors roam around the room, listen to conversations, and jump in at key moments to offer guidance. In on-line settings, group learning facilitators are responsible for a larger number of groups. Listening in on conversations involves reading a high volume of text based chat, often paired with data streams in other modalities such as a digital whiteboard. The challenge is to enable group learning facilitators to quickly get a sense of the collaborative interaction so his resources can be strategically invested. In this poster, we present SIDE (the Summarization Integrated Development Environment), an infrastructure that supports group learning facilitators in quickly getting a sense of the quality and effectiveness of a collaborative learning interaction so that the instructor is better equipped to carry out this challenging task.

It has been found that effective learning in collaborative groups is linked to the process by which learners work on the task together, how they construct arguments, and how they build on the contributions of their learning partners, otherwise known as transactivity. Earlier research suggests that it is more effective to judge the quality of an interaction for learning when transactivity based conversational contributions are flagged [2]. The goal of our proposed reporting interface is to track these qualities of collaborative discourse as is becoming more practical because of projects such as TagHelper tools [1] that are capable of automatic collaborative learning process analysis. However, investigating the patterns of conversational behavior that are most indicative of the quality of an interaction or indicative of trouble in an interaction, is still an active area of research. Open questions related to which types of information and how much of it is ideal to present to

[1] This research was supported by Office of Naval Research, Cognitive and Neural Sciences Division, grant N00014-05-1-0043.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 793–795, 2008.

group learning facilitators, and in what form, are still at the frontier of this important area of research. Thus, rather than present a specific reporting tool designed to do a single type of analysis and present reports in one way, we present a framework that facilitates rapid prototyping of such reporting interfaces so that research in this area can move forward at an accelerated rate.

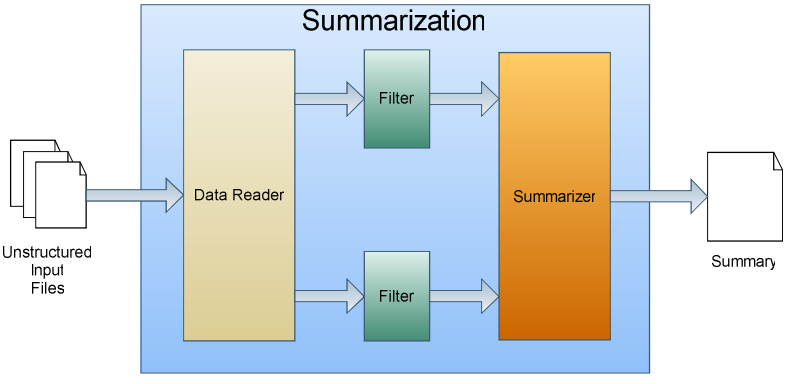


Fig. 1. SIDE Architecture

The architectural diagram in Figure 1 illustrates the two stage process of building a report from a discussion log. Consider the example of a report that charts how the consensus building style of different participants in an on-line discussion shifts over time [3]. The documents input to the process in Figure 1 are individual contributions to the discussion. A Data Reader reads in those texts and stores them in a flat representation. The Filters each encode a type of analysis that can be applied to the unstructured conversational data- e.g. one Filter might be responsible for applying a coding scheme that labels argumentation acts such as Claim, Data, Qualifier, or Warrant, while another flags segments at a higher level of argumentation structure such as Argument, Counter-Argument, or Integration. Thus, each Filter encapsulates the analysis functionality currently included in tool sets like TagHelper tools [1]. Once the Filters are applied to unstructured data, it then becomes structured data. The segmentation and labeling facilitate navigation through the data in a principled fashion. It is then possible to determine the most common type of contribution within a region of a conversation, or the last time someone contributed a Counter-Argument. The Summarizer in Figure 1 uses the structure applied by the Filters to select a subset of the conversation and presents either the texts themselves or some aggregation of the texts as the summary, for example by plotting the concentration of Counter-Arguments within 10 minute intervals over the course of a 2 hour discussion.

The interface for the Summarizer is displayed in Figure 2. A separate interface is provided for defining Filters, which is not shown. The Summarizer interface allows a summary to be defined as a sequence of Recipes. Each Recipe consists of a Selector, a Ranker, a Limiter, a Sequencer, and a Display. The Selector uses the analysis applied by the Filters to select a subset of segments of data that satisfy a Boolean expression- e.g. the expression might specify that all Counter Arguments and Integrations should be selected. The Ranker specifies criteria used to rate selected segments with respect

to its importance for the summary. The Limiter specifies what proportion of the segments that were selected by the Selector should be retained and presented in the report- if 50% is selected, then the segments will be sorted based on the Ranker criteria, and the top half will be retained. These segments are then ordered by the Sequencer in the order in which they were contributed to the conversation. Finally, the sequenced list of segments are passed to a Display module, which might simply present the text from each segment in order, or might present some summary of what was found, like the ratio of Counter Arguments to Integrations, etc.

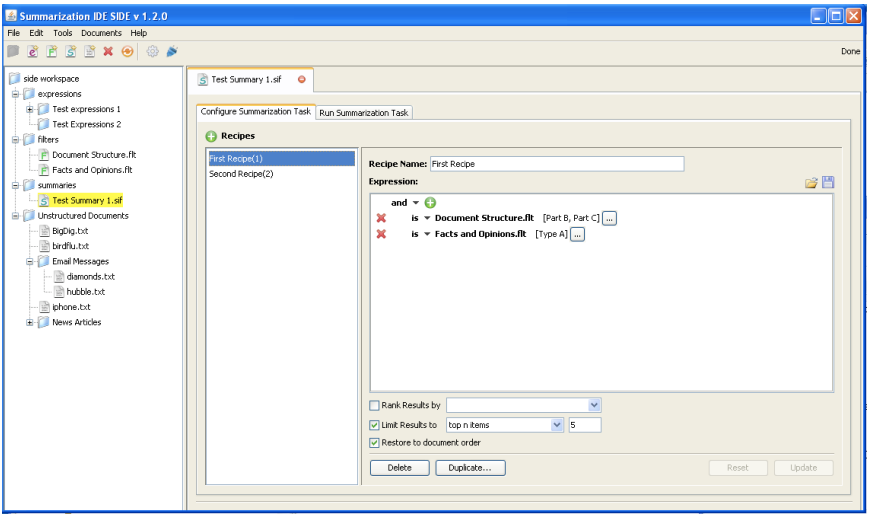


Fig. 2. SIDE Summarizer Interface

We are currently pilot testing SIDE in a graduate course on Summarization and Personal Information Management where students are using it to develop prototype reporting tools, so that we can determine what additional functionality is desirable.

References

1. Rosé, C.P., Wang, Y.C., Cui, Y., Arguello, J., Stegmann, K., Weinberger, A., Fischer, F.: Analyzing Collaborative Learning Processes Automatically: Exploiting the Advances of Computational Linguistics in Computer-Supported Collaborative Learning. International Journal of Computer Supported Collaborative Learning (submitted, 2008)
2. Joshi, M., Rosé, C.P.: Using Transactivity in Conversation Summarization in Educational Dialog. In: SLaTE Workshop on Speech and Language Technology in Education (2007)
3. Weinberger, A., Fischer, F.: A framework to analyze argumentative knowledge construction in computer-supported collaborative learning. Computers & Education (2006)

Comparing Two IRT Models for Conjunctive Skills

Hao Cen, Kenneth Koedinger, and Brian Junker

Carnegie Mellon University
5000 Forbes, Pittsburgh, PA, U.S.A.
hcen@andrew.cmu.edu, koedinger@cmu.edu, brian@stat.cmu.edu

Abstract. A step in ITS often involve multiple skills. Thus a step requiring a conjunction of skills is harder than steps that require requiring each individual skill only. We developed two Item-Response Models – Additive Factor Model (AFM) and Conjunctive Factor Model (CFM) – to model the conjunctive skills in the student data sets. Both models are compared on simulated data sets and a real assessment data set. We showed that CFM was as good as or better than AFM in the mean cross validation errors on the simulated data. In the real data set CFM is not clearly better. However, AFM is essentially performing as a conjunctive model.

1 Introduction

A step in ITS is "the smallest possible correct entry that a student can make. It connects the transaction-level representation to the theoretically-derived knowledge component level" [1]. Often times in ITS, students face steps with conjunctive skill requirements -- The student needs multiple skills to solve the whole step. Thus, a step requiring a conjunction of skills is harder than steps that require requiring each individual skill only. In this paper, we want to answer the following questions:

1. Are the skills in the student log data set conjunctive in character?
2. What is a better method to model this conjunctivity?

These questions are of importance to the ITS community who is interested building quality cognitive model. Two popular classes of ITS – the Model Tracing Tutor, which builds explicitly on the cognitive model, and the Constraint-based Tutor [2] – represents domain knowledge as state constraints The effectiveness of these tutors is largely determined by the quality of written skills and constraints. A conjunctive model can be useful in evaluating the cognitive model against empirical student data.

2 Two IRT Models

To test the skill conjunctivity in the data, we developed two parametric IRT models. The first model – the Additive Factor Model (AFM) –, originated from [3], is depicted by Equation (1). The term "Additive" comes from the fact that a linear combination of skill parameters determines $\text{logit}(p_{ij})$ in the equation

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 796–798, 2008.

$$p_{ij} = \Pr(Y_{ij} = 1 \mid \theta_i, \boldsymbol{\beta}, \boldsymbol{\gamma}) = \frac{\exp(\theta_i + \sum_{k=1}^{K} q_{jk}\beta_k + \sum_{k=1}^{K} q_{jk}\gamma_k T_{ik})}{1 + \exp(\theta_i + \sum_{k=1}^{K} q_{jk}\beta_k + \sum_{k=1}^{K} q_{jk}\gamma_k T_{ik})} \quad (1)$$

Where

Y_{ij} = the response of student i on item j

θ_i = coefficient for proficiency of student i

β_k = coefficient for difficulty of skill k

γ_k = coefficient for the learning rate of skill k

T_{ik} = the number of practice opportunities student i has had on the skill k

q_{jk} = 1 if item j uses skill k; 0 otherwise

K = the total number of skills in the Q-matrix

The intuition of this model is that the probability of a student getting a step correct is proportional to the amount of required knowledge the student knows, plus the "easiness" of that skill, plus the amount of learning gained for each practice opportunity.

The second model -- the Conjunctive Factor Model (CFM) -- depicted by Equation (2) can be thought as modeling the conjunctivity as a multiplication of skill parameters. CFM is a special case of Embretson's multicomponent latent trait model [4] and is customized for the high dimensional feature of ITS, as there are many more skills in a cognitive model than the number of cognitive attributes in a traditional assessment.

$$p_{ij} = \prod_{k=1}^{K} \left(\frac{e^{\theta_i + \beta_k + \gamma_k T_{ik}}}{1 + e^{\theta_i + \beta_k + \gamma_k T_{jk}}}\right)^{q_{jk}} \quad (2)$$

To fit the parameters from the data, we designed a Penalized Maximum Likelihood Estimation method (PMLE) depicted in equation (3) to overcome over fitting. This PMLE penalizes the oversized student parameters in the joint estimation of the student and the skill parameters. Maximizing Equation (3) is equivalent to finding a posterior mode for a Bayesian model, with a normal prior on θ and flat priors on β and γ. A higher value for λ below corresponds to lower prior variance. The BFGS optimization algorithm is used in computing PMLE.

$$ll_{PMLE} = ll_{MLE} - \frac{1}{2}\lambda \sum_{i=1}^{I} \theta_i^2, \ \lambda = 1 \text{ by default} \quad (3)$$

where I = the total number of students.

3 Simulation Results, Real Data Results and Discussion

To compare CFM with AFM, we used both a simulated data set and a real assessment data set. Since students in the assessment data set were not exposed to repetitive learning opportunities, we removed the learning term from both models. Cross validation errors and the interpretability of the actual parameter fits are used to evaluate the models.

The simulated data is used to answer the question "If the data is conjunctive, which model is better?" We simulated data drawn from a CFM model with 100 student parameters, 3 skill parameters, and 7 items. We explored for different sets of the three skill probability values (.1, .5, .9), (.1, .1, .1), (.4, .5, .6) and (.9, .9, .9). In nearly all cases, CFM-PMLE was as good as or better than AFM-PMLE in cross validation. The biggest difference was from the skill set (.9, .9, .9) because the skill parameter values are so high that AFM-P cannot behave in a conjunctive form (which it can if the logic values are negative). Table 1 shows the results from one of the above skill sets.

Table 1. Model comparison of the simulated data. β = (.1, .5, .9).in probability.

	CVMean	CVSd	$\hat{\beta}$ in probability	$\hat{\beta}$ in logit
AFM-P	0.120	0.281	(0.03, 0.34, 0.73)	(-3.4, -0.67, 0.97)
CFM-P	0.111	0.174	(0.07, 0.5, 0.89)	(-2.54, 0.02, 2.07)

The real data is used to answer the questions "Is the data conjunctive? If so, which model is better" The real data set EAPS is taken from a difficulty factor study of 247 U.S. algebra students [5]. There are 1976 observations and 96 distinctive items. A simplification of their skill coding involves 3 skills.

Table 2. Model comparison of the EAPS data

	CVMean	CVSd	$\hat{\beta}$ in probability	$\hat{\beta}$ in logit
AFM-P	0.202	0.142	(0.35, 0.47, 0.43)	(-0.63, -0.14, -0.3)
CFM-P	0.187	0.221	(0.61, 0.7, 0.67)	(0.43, 0.85, 0.7)

In the real data set, CFM-P is not clearly better. However, AFM-P is essentially performing as a conjunctive model because the estimates are all negative. We saw this in the simulated data set with lower probability skill estimates. A goal of future research is to test CFM-P on real data sets where AFM-P cannot act conjunctively as in Table 2.

References

1. Kurt VanLehn, K.K., Skogsholm, A., Nwaigwe, A., Hausmann, R.G.M., Weinstein, A., Billings, B.: Whats in a step? Toward general, abstract representations of tutoring system log data. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511. Springer, Heidelberg (2007)
2. Mitrovic, A., Koedinger, K.R., Martin, B.: A Comparative Analysis of Cognitive Tutoring and Constraint-Based Modeling. In: User Modeling 2003. Springer, Heidelberg (2003)
3. Cen, H., Koedinger, K., Junker, B.: Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement. In: 8th International Conference on Intelligent Tutoring Systems (2006)
4. Embretson, S.: Multicomponent Response Models. In: Linden, W.V.D., Hambleton, R.K. (eds.) Handbook of Modern Item Response Theory. Springer, Heidelberg (1997)
5. Koedinger, K.R., Nathan, M.J.: The real story behind story problems: Effects of representations on quantitative reasoning. The Journal of the Learning Sciences (2003)

The Effect of Providing Error-Flagging Support During Testing

Amruth Kumar

Ramapo College of New Jersey,
Mahwah, NJ 07430, USA

Abstract. We evaluated the effect of providing error-flagging, i.e., error detection but not error-correction support, during testing. We found that error-flagging is not a substitute for knowing the correct answer. It does not help students game the tutor. But, it does help students improve their test score.

1 Introduction

Delayed feedback, error-flagging and immediate feedback are three typical types of feedback provided by tutors. Error detection and error correction are treated differently by the three types of feedback: immediate feedback detects and suggests corrections for errors, whereas delayed feedback expects the learner to both detect and correct errors. Error-flagging is a via-media – it detects errors for the learner, but leaves it to the learner whether and how to correct the errors.

One of the studies with the ACT Programming Tutor [1] found that immediate feedback helped learn the fastest, followed by error-flagging and delayed feedback in that order. There was little difference among the types of feedback on tests. Another study with the SQL-Tutor [3] found one of the highest initial learning rates with error-flagging – measured in terms of the probability of violating a constraint after feedback had been provided on the constraint on prior occasions. In an earlier preliminary study, we had found that providing error-flagging was not effective when the tutor did not explicitly state that errors were being flagged. On the other hand, explaining and providing error-flagging during tests resulted in significantly better scores [4]. In this paper, we will analyze some of the data that we have collected since our earlier study to evaluate the effect of providing error-flagging support during tests.

In spring 2007, we conducted in-vivo controlled studies with the arithmetic tutor. The arithmetic tutor presents arithmetic expressions and asks the student to evaluate them step-by-step, i.e., one operator at a time. After the student has submitted his/her answer, the tutor provides delayed feedback – it lists how many steps the student solved correctly, and displays the correct evaluation of the expression. Normally, when the student is entering his/her answer, the tutor displays it in black. When the tutor is set to provide error-flagging feedback, it color-codes the student's answer as it is being entered – it displays the answer in red if it is incorrect and green if it is correct. So, it provides error-detection support. The student has the option to correct the error, since the undo option is always available to the student. But, the tutor does not provide any feedback as to why the answer is incorrect, so, it does not provide error-correction support. In a

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 799–802, 2008.

multiple-choice question, such support for error-detection but not correction could encourage the student to employ a trial-and-error approach to answering questions. But, in expression evaluation, if an expression includes n operators, there are n! different orders in which the expression can be evaluated, and infinite options for intermediate results. So, trial-and-error is not a gainful strategy for evaluating expressions when support is provided for error-detection but not error-correction.

2 Evaluation

The arithmetic tutor was set up to administer the pre-test-practice-post-test protocol, with both the practice and post-test being adaptive [2]. The tutor was used by 10 sections of *Computer Science I* course. We randomly assigned sections to either control or test group. Both control and test groups got delayed feedback. In addition, test group got error-flagging support. The tutoring session was limited to 30 minutes. Since our interest was in examining the effect of error-flagging on testing, we considered only the data from the pre-test. The pre-test consisted of 14 problems on 23 concepts. We considered data of only those students who attempted all 14 problems on the pretest.

Presumably, when a student is told that his answer is incorrect (but not why), he will attempt to revise his answer. So, we partitioned the problems solved by each student into those where the student revised his answer and those where he did not revise his answer. We did a 2 X 2 mixed factor ANOVA analysis with each student's average score on the problems on which the student did not versus did revise the answer as the repeated measure and the treatment (without versus with error-flagging) as the between-subjects factor. We found a significant main effect for revising the answer ($F[1,166] = 16.824$, $p = 0.000$) – students scored more without revision (0.879) than with revision (0.807). Students revise their answer when they are not sure of their answer (without error-flagging) or are told that their answer is incorrect (with error-flagging). In other words, students revise their answer on problems on which they would otherwise have scored fewer points. But, even after revising their answer, students scored fewer points than on the problems where they did not (need to) revise their answer, i.e., problems on which they already knew the correct answer. This difference was significant when error-flagging support was provided (See Table 1). So, *error-flagging, i.e., error-detection but not error-correction support is not a substitute for knowing the correct answer, and does not necessarily result in a student demonstrating the same performance as if the student knew the correct answer.*

We found a significant main effect for treatment ($F[1,166] = 37.434$, $p = 0.000$) – subjects without error-flagging averaged 0.763 whereas those with error-flagging averaged 0.922 points, indicating an inherent difference in the two groups, which is a confound of our study.

Finally, we did not find a significant interaction between revisions and treatment – both control and test groups scored less with revisions than without (See Table 1). But, the difference between the problems without and with revision was significantly smaller for the test group (average 0.15) than for the control group (average 0.538, $p = 0.000$), i.e., *test group students scored more points than control group students by revising their answers*. Table 2 lists the number and percentage of students who revised their answers, number and percentage of problems on which they revised their

Table 1. Both control and test groups scored less on problems with revision than on problems without revision. The difference was significant for the test group only. Test group scored significantly more than the control group with and without revision.

Arithmetic Tutor		Without Revisions	With Revisions	Within-subjects *p*
Control Group (Without Error-Flagging)	N	70	26	0.128
	Ave	0.803	0.722	
	Std-Dev	0.154	0.290	
Test Group (With Error-Flagging)	N	160	142	0.000
	Ave	0.942	0.892	
	Std-Dev	0.111	0.146	
Between-subjects *p*		0.000	0.007	

answers and the number and percentage of times they revised their answers, i.e., selected 'Undo last answer' or 'Clear entire answer' menu option. *Test subjects revised their answers far more often than control subjects.* Test subjects revised their answers on 30.43% of the 14 pre-test problems and 29.30% of the 33 possible steps. Curiously, 11% of the test subjects never revised their answers in spite of error-flagging support and 37.14% of the control subjects revised their answers even without error-flagging support. Without the benefit of error-detection support, however, control group students did not improve their scores as much as test group students. So, *error-flagging support does help students improve their score, even though this may not be on par with actually knowing the correct answer.*

Table 2. Number of students who revised their answers, number of problems on which they revised their answers, and number of times they revised their answers

Arithmetic	N	Revisers		Problems		Undos		Clears
		Total	% of N	Total	Per Student	Total	Per Student	
Control	70	26	35.7%	39	1.5	50	1.92	4
Test	160	142	88.75%	605	4.26	1373	9.67	47

Finally, we considered whether error-flagging was helping students game the system – randomly guess the correct answer with the aid of error-flagging. Since gaming involves random/uninformed guessing until the student finds the correct answer, the average for problems with revision should have been 1.0, which is clearly not the case. So, *error-flagging does not help students game the system.*

Error-flagging support may be helping students avoid inadvertent mistakes or re-consider incorrect "first-impression" answers. In either case, it behooves us to provide error-flagging support during online-testing to give students the benefit of the doubt, the best advantage.

Acknowledgements. Partial support for this work was provided by the National Science Foundation under grant CNS-0426021.

References

1. Corbett, A.T., Anderson, J.R.: Locus of feedback control in computer-based tutoring: impact on learning rate, achievement and attitudes. In: Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 245–252 (2001)
2. Kumar, A.N.: A Scalable Solution for Adaptive Problem Sequencing and its Evaluation. In: Wade, V.P., Ashman, H., Smyth, B. (eds.) AH 2006. LNCS, vol. 4018, pp. 161–171. Springer, Heidelberg (2006)
3. Mitrovic, A., Martin, B.: Evaluating Effectiveness of Feedback in SQL-Tutor. In: IWALT 2000, Palmerston North, December 4-6, pp. 143–144 (2000)
4. Kumar, A.N., Rutigliano, P.: The Effects of Error-Flagging in a Tutor on Expression Evaluation. In: 13th International Conference on Artificial Intelligence in Education (AI-ED 2007), Marina del Rey, CA, July 2007, pp. 599–601 (2007)

Cognitive Tutoring System with "Consciousness"

Daniel Dubois, Mohamed Gaha, Roger Nkambou, and Pierre Poirier

GDAC Laboratory, University of Quebec at Montreal
P.O. Box 8888, Centre-ville Station, Montreal, Quebec, Canada
{dubois.daniel,nkambou.roger,poirier.pierre}@uqam.ca,
{gaha.mohamed,faghihi.usef}@courrier.uqam.ca

Abstract. Developing a learning system with the intent of optimizing human knowledge acquisition is a complex endeavor. CTS (*Conscious Tutoring System*) is meant to help astronauts learn the manipulation of Canadarm2. Our proposition relies on a cognitive architecture involving "consciousness" mechanisms. We give a partial account of CTS' cognitive architecture in its current state of development, and how it compares to other cognitive architectures and agents.

Keywords: Conscious agent, Global Workspace, ITS, deliberation, personality.

1 Introduction

Developing an ITS able to evolve in an autonomous way to better adapt both to the learner and to the learning environment would be a great advantage for our educational system. One of the big challenges with such a system is how it may be capable to deal with a variety of knowledge sources to adapt, learn and act.

Manipulating Canadarm2, a complex robotic arm, requires serious training. As part of a Canadian Space Agency research project, a virtual simulator complemented with a non-cognitive tutoring system, RomanTutor (Nkambou, Belghith and Kabanza, 2006), has been developed in our lab. However, aside from showing possible manipulation paths, it mostly supports weak coaching thanks to a path-planner integrated in the system. The coaching mainly consists in providing the learner with feedbacks about his actions. This approach fails to support thorough diagnosis of learner's failures. RomanTutor tutoring capabilities are being augmented by those of our "conscious" cognitive tutoring agent. In this paper, we offer an updated quick overview of our solution on this issue.

2 CTS' Consciousness Capabilities

Computers and various cognitive architectures already incorporate many of consciousness features. Our research intends to take this trend closer to humanly structures by reproducing some of consciousness' mechanisms and phenomena, even though there is not a universal definition for these (Güzeldere, 1997). We chose to iteratively converge to biological plausibility from psychological plausibility. Baars' Global Workspace (GW) theory (Baars, 1997) offers a valuable framework to that end. A justification for our

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 803–806, 2008.

choice can be found in previous papers. CTS' fundamental aspects refer to the GW theory and on how Franklin's team translated it in computational terms (Franklin & Patterson, 2006). A description of CTS' conceptual architecture and the interactions between modules and various codelets is given in (Dubois *et al.*, 2007).

Consciousness is essential for the sense of self. It makes possible to refer to one's preferences and long-term goals, evaluate situations and solutions, and choose in which direction to improve. CTS currently has an embryo of personality (fundamental principles as motivators) that we are in the process of augmenting by a complete framework for principles, values and emotions. When, for instance, deliberating upon the poor success it meets in helping his student, CTS may come to associating aspects of the situation with traits of the personality it his displaying, and therefore decide to modulate somewhat those traits that may cause the difficulties. Consciousness is always involved in situations where "canned solutions" cannot play part.

As Figure 1 illustrates (in another scenario), a deliberation consists in examining inputs coming from various modules belonging to the architecture in loops of proposal-feedback. All CTS resources (Behavior Network, Learner Model, Domain Expert,

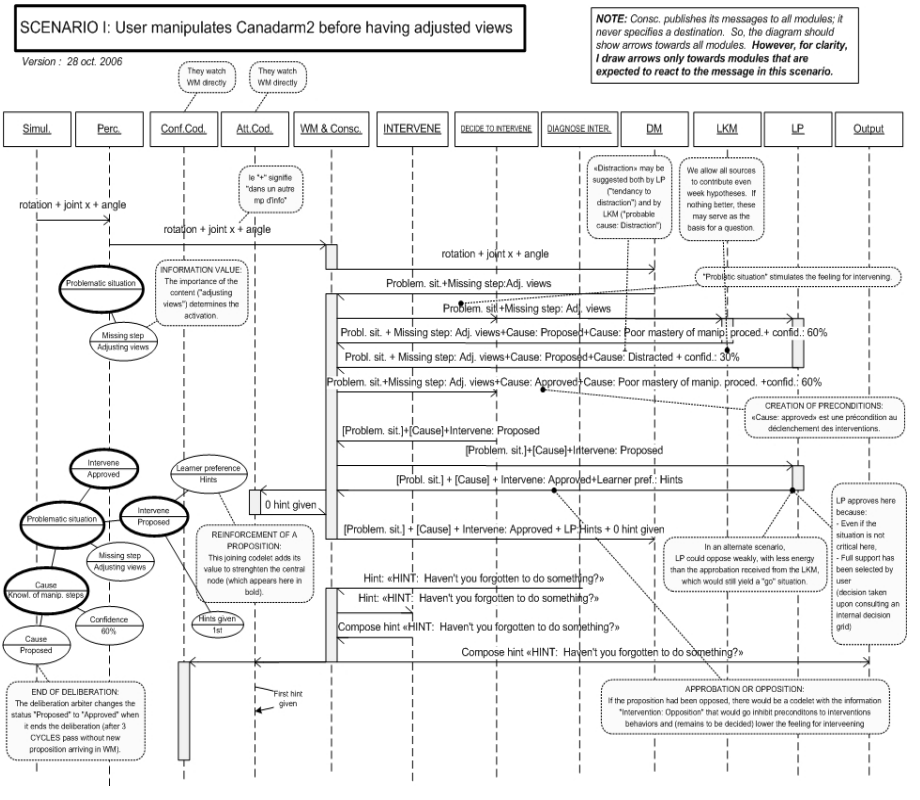


Fig. 1. Sequence diagram of a deliberation process. On the left, we show how a coalition of information codelets evolves along the way.

various memories, and Codelets) have to be made aware of the ongoing situation so they may contribute in achieving the most accurate assessment possible with the help of multiple angles of analysis. But no one knows in advance where the best contribution will come from; it is a contextual, emergent process.

Progressively, in the course of multiple cognitive cycles with "conscious" publications, the modules enrich or oppose certain information offered by other resources. An example of this happens when the astronaut appears inactive. Is he baffled and at a loss, or distracted, or annoyed and thinking of stopping, or is he analyzing and planning his move? The Learner Model may have many hypotheses to offer, one for each of its subcomponents (Learner Profile, Learner Knowledge, Learner Affect); the same is true of Episodic Memory, which may bring back parallels with other similar situations. After publication, an hypothesis suggesting that he is demotivated (Affect) may receive reinforcement from Episodic Memory with an implicit causal explanation of this demotivation (he previously had difficulties in similar situations) and from Learner Profile (this astronaut has a tendency to doubt himself and get discouraged). But this hypothesis may be opposed by the Domain Expert that states that this operation simply is difficult. With this opposition, the proposal looses some of its attractiveness. In effect, CTS may decide to wait a little more before attempting an intervention.

Sometimes there is no solution to the ongoing problem. In this case, the system renounces the deliberation and publishes the failure; the BN will then provide the most appropriate way to react to the situation in the current context, often soliciting input from the learner. More about deliberation can be found in (Dubois *et al.*, 2007).

3 Comparison with Other Cognitive Architectures

CTS' architecture parallels other cognitive agents and agent architectures: symbolic ones such as ACT-R, PRS (BDI), Blackboard, SOAR, CS/SAS, Icarus, etc., and, subsymbolic ones, such as CLARION. This comparison exceeds the available space, but we can draw some general conclusions about some of them. We have described many of CTS's capabilities put forth here in previous papers.

CTS incorporates PRS completely, which has been the first implementation of the BDI theory. What CTS adds is the ability to deal with emotions and their roles. It also allows the building of plans on-the-fly. CTS is also more faithful to the human model, with "consciousness", simple and specialized "unconscious processes", and learning.

The Blackboard architecture is also a foundation to CTS' functioning. CTS' architecture extends the original idea with the capability to learn and selectively keep successful sequences of action (solutions), automatization, creation of unexpected associations in WM (creativity), global and implicit analysis of situations (intuition), emotional influence on cognition.

In most its principles and mechanisms, CTS is also very close to ACT-R, with sub-symbolic (or rather, implicit) as well as symbolic levels, modularity, concurrent parallelism/serialism, etc. ACT-R clearly exceeds CTS in its predictive capability, as the equations driving its declarative memory strive to reproduce experimental results. Other differences exist in the way information is processed: ACT-R's rules do not readily lend themselves to a meta-level; is uses a simpler hypothesis about the cognitive cycle, and supports no clear separation between conscious and unconscious processing. With an

explicit central coordinator between the modules (the rules themselves), there's no room for unexpected solutions (intuition and creativity). Finally, adding other aspects such as emotions and multiple aspects regarding the learner might prove a challenge, at least if they are to be part of the rules as decision parameters.

4 Conclusion

In cases of incorrect maneuvers, the "consciousness" mechanism implemented in CTS allows diagnosing the situation by gathering opinions from multiple resources that look into the problem from their particular point of view. Although not described in this article, CTS' personality and fundamental principles also participate in decisions.

Our current works seek to enrich CTS' pedagogical capability. We are also designing and implementing emotional processing and some machine learning mechanism to help orient tutor's decisions, action, and support to the learner.

Acknowledgments. Our thanks go to the Canadian Space Agency and the Natural Sciences and Engineering Research Council (NSERC) for their logistic and financial support. Special thanks to Professor Franklin and to the University of Memphis, who graciously granted us access to their IDA technology. We also thank all the members of the GDAC Laboratory.

References

1. Baars, B.J.: In the theatre of consciousness: Global Workspace Theory, a rigorous scientific theory of consciousness. In: Consciousness Study, pp. 292–309 (1997)
2. Nkambou, R., Belghith, K., Kabanza, F.: An Approach to Intelligent Training on a Robotic Simulator using an Innovative Path-Planner. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 645–654. Springer, Heidelberg (2006)
3. Dubois, D., Poirier, P., Nkambou, R.: What Does Consciousness Bring to CTS? In: AI and Consciousness: Theoretical Foundations and Current Approaches: Papers from the 2007 AAAI Fall Symposium: Technical Report FS-07-01, pp. 55–60. AAAI Press, Menlo Park (2007)
4. Franklin, S., Kelemen, A., McCauley, L.: IDA: A Cognitive Agent Architecture. In: Systems, Man and Cybernetics. IEEE Press, Los Alamitos (1998)
5. Franklin, S., Patterson Jr., F.G.: The Lida Architecture: Adding New Modes of Learning to an Intelligent, Autonomous, Software Agent. In: Integrated Design and Process Technology, IDPT 2006. Society for Design and Process Science, San Diego, CA (2006)
6. Güzeldere, G.: The Many Faces of Consciousness: A Field Guide. In: Block, N., Flanagan, O., Güzeldere, G. (eds.) The nature of consciousness: Philosophical debates, pp. 357–373. MIT Press, London (1997)

It's Not Easy Being Green: Supporting Collaborative "Green Design" Learning

Sourish Chaudhuri[1], Rohit Kumar[1], Mahesh Joshi[1], Elon Terrell[3], Fred Higgs[3], Vincent Aleven[2], and Carolyn Penstein Rosé[1,2]

[1] Language Technologies Institute
[2]Human-Computer Interaction Institute
[3]Mechanical Engineering Department, Carnegie Mellon University,
5000 Forbes Avenue, Pittsburgh, PA 15213
{schaudhu,rohitk,maheshj,eterrell,higgs,va0e,cp3a}
@andrew.cmu.edu

Abstract. We present the results of a study[1] in which we contrast alternative forms of collaborative learning support in the midst of a collaborative design task in which students negotiate between increasing power and increasing environmental friendliness. In this context, we evaluated the instructional effectiveness of four alternative support conditions as well as a goal manipulation. Both manipulations yield surprising findings, which we are continuing to investigate.

1 Research Study

We are conducting our research on dynamic support for collaborative design learning in the domain of thermodynamics, using as a foundation the CyclePad articulate simulator [1,2] which allows students to implement design ideas using graphical interface widgets, and to explore the relationships between the settings of various parameters within the cycle design. In the study described in this paper, we specifically focus on issues related to "Green Design" with Rankine cycles. In the collaborative design exercise described below, students work in pairs to struggle with trade-offs between power output and environmental friendliness in the design of a Rankine cycle, which is a type of heat engine. We assigned each student within each pair to a different competing goal, with one student instructed to increase power output as much as possible and the other student instructed to make the design as environmentally friendly as possible. The trade-offs involved in this task offer students the opportunity to find one of the major motivations for seeking to increase the efficiency of a designed cycle.

84 students participated in the study by attending one of four lab sessions, which were structured into multiple phases during which we strictly controlled for time. At the beginning of each lab session that was part of the study, students were lead through formal training on the simulation software from an instructor using power point slides and the simulation environment. They then worked on Rankine cycles in Cyclepad using information from a booklet given to them, which was developed by a professor from the

[1] The research was supported by Office of Naval Research, Cognitive and Neural Sciences Division, grant number N00014-00-1-0600.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 807–809, 2008.

Mechanical Engineering Department. Subsequent to this, they took the pre-test, immediately before the experimental manipulation. The exploratory design exercise, which followed, was where the students worked in pairs using CyclePad and the ConcertChat collaboration environment [3] to optimize power and environmental friendliness. This was followed by the post-test and the questionnaire.

In our experiment presented below, we contrast 3 support conditions where dialogue agents offer different levels if interactive instruction within the chat environment, with a Control condition where no support was offered. Students use a collaboration software package called ConcertChat [3] to chat with each other as well as using the digital whiteboard associated with that environment to pass graphical information back and forth to one another.

In order to control for information presentation, the 3 support conditions offered students exactly the same information as was included in the booklet that all students had access to, although it was presented to them in 3 different ways. In the Knowledge Construction Dialogue (KCD) condition, a dialogue agent participated in the conversation at 3 pre-determined times in order to engage students in discussion related to a topic related to their design task. This form of support was evaluated successfully in a prior study [4]. In the second "Pointer" condition- rather than engage students in a dialogue, students were simply asked to refer to the page in the book that included the same information that was offered by the KCD. In the final "Minilesson" condition, the same information was inserted into the chat buffer by the dialogue agent, and students were not required to find the material in the book. While all three methods direct students to important information, they differ with respect to how intrusive they are and how much effort they require the student to make. Thus, by comparing learning effectiveness across all four conditions, we can determine whether the primary contribution of the KCDs used successfully in prior studies was the fact that it addresses the problem that students have trouble determining what information they need at key points, or if part of the value is in the interactive manner in which KCDs present information to students.

As outcome measures, we examined learning gains between Pre and Post test. 35 multiple choice and short answer questions were used to test analytical and conceptual knowledge of Rankine cycles. We also examined the quality of their design rationales contributed at the end of the collaborative design exercise. Finally, we compared answers to affective questionnaire items across conditions.

First we examined the effect on learning of the goal manipulation, i.e., whether students were assigned to the goal to achieve the greatest power (Power) or the condition to achieve the greatest environmental friendliness (Green). We were surprised at the result. With an ANCOVA analysis where we use total post-test score as the dependent variable, pre-test score as the covariate, and Goal as the independent variable, we see a marginal effect in favor of the Power condition $F(1,83) = 2.68$, $p = .1$, effect size .3 standard deviations. In terms of likelihood of pre to post test gain, we determined that students in the Power condition were marginally more likely to gain between pre and post test than students in the Green condition based on a binary logistic regression ($P < .1$). On the questionnaire, students in the Green condition also reported higher tendencies to feel influenced by their partner's point of view than their counterparts in the Power condition. Nevertheless, not too surprisingly, we see an effect on their design rationales. In particular, while there is no significant difference in likelihood of mentioning efficiency or an

efficiency related factor in the rationale between conditions, students in the Power condition were marginally more likely to mention a Power related factor ($P < .1$) whereas students in the Green condition were significantly more likely to mention an environmental friendliness related factor ($P < .05$). Although the negative effect of the Goal manipulation on students in the Green condition is only marginal, it is still surprising since students in the two conditions worked in pairs, where in each pair there was one student from each of these conditions. So they were exposed to all of the same material and worked through exactly the same solution. Considering also that it would have been more desirable for the Green condition to be the preferred condition, it is disappointing that the result came out the opposite, and to add insult to injury, the Green students were more swayed towards the Power side than the opposite.

The results with respect to the support variable were no less troubling. Comparing the frequency of students who gained between pre and post test across conditions using a binary logistic regression, all support conditions were significantly more likely to produce a pre to post test gain than the Control condition. However, to our surprise, the Pointer and Minilesson conditions were also significantly more likely to produce a pre to post test gain than the KCD condition. This shows that the potential gain for instructional support in the midst of collaborative learning is not restricted to the navigation problem we noticed in earlier studies, where students are not sure what material to attend to. It is disappointing, however, that the interactive support provided by the dialogue agents was worse than the non-interactive support. We suspect that this has to do with the interruptive affect of the agents that has been noted to decrease the level of transactivity in the discussion between students in collaborative pairs in our earlier studies. If we can address this problem, we believe we can dramatically improve the instructional effectiveness of tutorial dialogue agents for supporting collaborative learning, building on prior very encouraging results [4].

Moving forward from here we would like to do a much more in depth analysis of the collaborative learning discussion logs, especially with respect to transactive conversational behavior across the four conditions.

References

1. Forbus, K.D., Whalley, P., Everett, J., Ureel, L., Brokowski, M., Baher, J., Kuehne, S.: CyclePad: An articulate virtual laboratory for engineering thermodynamics. Artificial Intelligence 114, 297–347 (1999)
2. Wu, C.: Intelligent Computer-Based Engineering Thermodynamics and Cycle Analysis. Nova Science Publishers, New York (2002)
3. Concert Chat (2006), http://www.ipsi.fraunhofer.de/concert/
4. Kumar, R., Rosé, C.P., Wang, Y.C., Joshi, M., Robinson, A.: Tutorial Dialogue as Adaptive Collaborative Learning Support. In: Proceedings of Artificial Intelligence in Education (2007)

Cognitive and Technical Artefacts for Supporting Reusing Learning Scenario Patterns

Emmanuelle Villiot-Leclercq[1] and Aude Dufresne[2]

[1] LIG, Université Joseph Fourier, 110 av. de la Chimie - Domaine Universitaire - BP 53 - 38041 Grenoble - cedex 9, France
[2] Université de Montréal, Département de Communication, Pavillon Marie-Victorin 90 Vincent-d'Indy, Outremont QC H2V 2S9, Canada
{emmanuelle.villiot-leclercq}@imag.fr,
aude.dufresne@umontreal.ca

Abstract. We present an intelligent tutoring system for teachers, which was designed to support them while they learn and adapt a learning scenario for their teaching. The system is a learning environment where teachers can explore different learning scenarios and get advice on the criteria for choosing among them. Then they choose a "case study" scenario and learn to adapt it to their own context. Different kind of support are given during the interaction with the system, declarative and procedural help is presented as indexed help on each activity, but also as contextual help using a rule based advice system. The system was experimented with eight teachers (four academic and four from industry), who succeeded to adapt the 'case study" scenario to their own context. The evaluation and discussion present what help they found more useful and how the system could be improved.

Keywords: Learning system, web based learning, agent support to learning.

1 Introduction

Research on computer support for learning are trying to share not only ideas but learning objects and also to address the need to share and reuse learning objects, as well as the expertise on the pedagogy using computers for distant learning, in classrooms or when developing eportefolios systems. A scenario can be seen as "a set of organised tasks, ruled by actors, that use and define resources » [1].

A recent study across Candada [3] shows that Learning Objects are not being used that much. Learning scenarios are not easy to reuse and to adapt [4] as the content and context may vary, and each authors needs to control how it is integrated in his specific context. For Cromier [5] the reuse of learning scenarios must answer the need to save time, bring new ideas, discover new pedagogical methods, but scenarios must exist, the teacher must be open to reuse, and he must be trained to do it.

We proposed a model to support the design process, offering an environment where scenarios could be reused, and where the task to design learning scenarios could be explained. We present the experimentation of the system with teachers.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 810–812, 2008.

2 Designing a System to Support the Reuse and Adaptation of Scenarios

Villiot-Leclercq [6] proposes to define reuse as a process in three steps, which must be supported : the choice, the appropriation and the adaptation of the scenario. One of the challenges when supporting a task is to take into account the needs of users in the context of the task (complexity, time constraitns, learning styles, objectives, etc.) [7].

Many theories and research highlight the important parameters to adequately support learners: individual reactions to support, instructional design theories on the structures of information for learning [8], on the use of proximal development zone [9]. According to situated learning approach, learning can only occur in relation to a relevant context [10] : a real problem, with real activities, with access to expert knowledge, asking to explicit knowledge, in a collaborative context, with scaffolding and evaluation of learning [11]. Thus in the context of this research on supporting the design of scenarios, we tried to meet those requirements: an authentic problem, access to expertise and to explicit processes, scaffolding with multi-views perspective on the knowledge to be learned, co-construction with reflection and coaching. We designed a set of cognitive and technical artifacts to support the learning process activity:

- A scenario editor Explor@Graph [2].
- A formal model to express scenarios as Pleiades formalism [12].
- A graphic model of the scenario, on how to reuse, and adapt scenarios.
- Different models of typical scenarios also called Patterns
- Declarative, procedural and strategic knowledge on scenarios design .
- A real and a virtual coach to support the learning process.

The Explor@Graph Scenario Editor was used to present typical scenarios and to support teachers learning to adapt a « case study » scenario to their own context.

3 Experimentation of the Learning System

An evaluation of the system was done with eight users: four teachers which were collaborators in the CAUSA project on educational scenarisation practices and four insctructional designer at Symetrix an e-learning business designing online courses.

They are all expert designers who use technology on a regular basis, having experience with learning scenarios, so they could have a critical point of view on the proposed formalism, on the support environment and on the help system.

The assessment has identified some tracks on the advantages and limitations of each type of artifact : interest to use formalized scenario, benefits of the editor based on a graphical and flexible representation, interest for suggestions focused on pedagogical aspects, but also for their improvement (more procedural suggestions, more contextualized suggestions, in parallel but not interrupting the activity, which are also accessible as indexed help.To improve the support to different context, it would be interesting to adapt the system to different profiles of teachers and to different learning contexts (institutional / industrial).

Finally our work has focused on a device to support reuse of individual scenarios, however, reuse can be done in a collective way, and it is necessary to consider mechanisms for achieving support in such collective context of resuse, for example in team teaching or community of practices. In such sharing context, typical scenarios might evolve, as well as the pedagogical suggestions, which are linked to them.

References

1. Paquette, G.: Apprentissage sur Internet: des plateformes aux portails d'objets à base de connaissance. In: Pierre, S. (ed.) Innovations et tendances en technologies de formation et d'apprentissage. Presses de l'école polytechnique de Montréal, pp. 1–30 (2005)
2. Dufresne, A.: Conception d'une interface adaptée aux activités de l'éducation à distance - ExploraGraph. STE 8(3), 301–320 (2001a)
3. Robertson, A.: Rapport d'analyse sur les banques d'objets d'apprentissage francophones pancanadiennes. Colloque Les défis reliés à l'intégration pédagogique des ressources numériques, CRÉPUQ, Sherbrooke (2006)
4. Dufresne, A., Senteni, A., Richards, G.: La contextualisation des banques de ressources– barrières et clés. Can. Journal of Learning Technology 28(3), 27–42 (2002)
5. Cromier, L., Herandez, A.: L'intégration des REA, oui mais où sont-elles? Les défis reliés à l'intégration pédagogique des ressources numériques, CRÉPUQ, Sherbrooke, mars (2006)
6. Villiot-Leclercq, E.: Modèle de soutien pour l'élaboration et la réutilisation de scénarios pédagogiques, thèse de doctorat, Université Joseph Fourier/Université de Montréal (2007)
7. Arroyo, L.: A Concept-Based Approach to Support Learning in a Web-based Course Environment. In: Moore, J., Redfield, C.L., Johnson, W.L. (eds.) AIED. IOS Press, Amsterdam (2001)
8. Tricot, A., Pierre-Demarcy, C., El Bousssaghini, R.: Définitions d'aides en fonction des types d'apprentissages dans des environnements hypermédias. In: Hypermédias et Apprentissage 1998. Poitiers: LaCo-CNRS INRP, pp. 3–14 (1998)
9. Luckin, R., Underwood, J., Du Boulay, B., et al.: Designing Educational Systems fit for use: a case study in application of Human Centred Design for AIED. International Journal of Artificial Intelligence in Education 16, 353–380 (2006)
10. Herrington, J., Oliver, R.: An instructional design framework for authentic learning environments. ETR&D 48(3), 23–48 (2000)
11. Basque, J., Dao, K., Contamines, J.: L'apprentissage " situé " dans les cours en ligne: le cas du colloque scientifique virtuel (CSV). In: Tchounikine, P., Joab, M., Trouche, L. (eds.) Actes de la conférence EIAH 2005, INRP. Institut Montpellier II, pp. 177–188 (2005)
12. Villiot-Leclercq, E.: La méthode des Pléiades: un formalisme pour favoriser la transférabilité et l'instrumentation des scénarios pédagogiques. STICEF 14 (2007)

Integration of a Complex Learning Object in a Web-Based Interactive Learning System

Françoise Le Calvez and Hélène Giroire

LIP6 – MOCAH University Pierre and Marie Curie (Paris 6),
104 avenue du président Kennedy, 75016 Paris, France
{francoise.le-calvez,helene.giroire}@lip6.fr
http://www-desir.lip6.fr/

Abstract. Usually, learning objects are described and indexed in repositories using metadata so that users can select them to create new on line courses. What must the granularity level of these metadata be to make them fit the learners' needs? We describe here the problems encountered in integrating a complex learning object to train learners in solving problems in a web-based interactive learning system based on a learner model, and we give a possible solution.

Keywords: metadata, learning object, ITS, adaptive learning environment.

1 Context

To define learning sequences is a subject of research, which is foregrounded because indexing and describing learning objects are now progressing on the way of normalisation and some repositories begin to exist and be used. As Brooks and al. [2] said: "combining ITS and e-learning Technologies is a challenge". Our work comes within this context using two mature systems of different and complementary nature.

Our aim is to add a complex activity to a web-based interactive learning system ActiveMath [1] thanks to a learning object Combien? [3] [4], which is a training environment. We would like to take advantages of the wealth of the two environments. ActiveMath enables users to create courses from a set of learning objects relative to a concept to be learned. It uses a non domain specific learner model to choose the exercise that best fit the learner's level at a given moment [5]. The Combien? software enables learners to train themselves in solving combinatorics problems. It is a complex environment based on a meta-model to express the domain, the problems and the solutions. It proposes rich exercises (not MCQ) to learn a solving method. It contains a mechanism to detect errors incrementally, using error schema bases so that learners cannot continue their solving if it leads to a dead-end; in this case Combien? gives them hints to continue the solving rightly. It contains numerous varied exercises; each of them is in itself a learning object.

Let us take an example of Combien? exercise EX1: "How many five-digit numbers are there with exactly three occurrences of the digit 0?". The general form is: "given a set or sets (here, the set of positions P, and the set of digits D), count within some universe of configurations (here each configuration is a mapping from P to D and the

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 813–815, 2008.

universe is the set of all these mappings) the elements that satisfy some constraints (here the injective mappings whose range contains the digit 0 exactly three times) ". Solving a combinatorics exercise consists in counting the cardinal of the set SeC (set of the elements which satisfy the constraints). It forces learners to use the "constructive method" that consists in building step by step an element of the set to be counted SeC and then to reason about this construction to count the cardinal of SeC.

To make ActiveMath and Combien? interact, metadata have to be compatible. A "good" level of metadata is necessary to allow efficient search for finding the exercise most suited to the learner at a given learning stage. The main characteristic of the Combien? exercises is the class of the exercise according to the solving schema. Then for each class the solving difficulty depends on three types of skills: the modelling of the exercise statement, the finding of appropriate constraints, and the use of appropriate formulas to calculate.

2 Extra Metadata

ActiveMath uses metadata LOM compatible on the basis of DublinCore and OMDoc. Exercises are annotated by metadata relations *for*, *pre-requisites* or *counter* which link them to the concepts. Activemath has extended these metadata for pedagogical purpose. We are particularly interested in the relation *for* which links exercises and concepts, and in metadata *difficulty* which can be annotated by the learning context, in *competencylevel* and in *competency* (adopted from PISA). But we need more specialized metadata to distinguish between Combien? exercises.

To use the relation *for*, let us see the concepts which Combien? exercises are linked to. The first concepts concern the class of the exercise which is bound to the type of sets to be built (a set of sets, a set of functions, the union of sets of sets or of sets of functions). Other concepts are the two combinatorics principles, the multiplicative principle and the additive principle. For instance, EX1 is linked to the building of a set of functions and to the multiplicative principle by *for* relations.

Competency and *competencylevel* are very informative for the learner model. They respectively describe the mathematical competencies and the mathematical knowledge level an exercise trains. *Competencylevel* is a metadata which is associated not only to an exercise but also to the learner model. Thus, the estimated difficulty of an exercise for a learner is calculated on the one hand from the values of the metadata competency level and difficulty of the exercise and on the other hand from the value of the competency level of the learner. As we have seen in the previous section, the difficulty of a Combien? exercise depends on various characteristics. The difficulty metadata must be linked with these characteristics. We propose to extend annotations for *difficulty* using a new relation *for_skill*. For example, the *difficulty* linked to the characteristic *modelling* defined in the previous section, can be expressed by a level of difficulty for the skill *modelling*. Values of skills specify the values of ActiveMath competency which are too general for our purpose. Here, the skill *modelling* is focused on a deep understanding of the exercise statement and can be linked with the *model* competency. To ensure the possibility of describing all the occurrences of difficulty according to the characteristics we presented, we define a vocabulary for

learners' skills in the context of Combien?: *modelling* of the exercise statement, *determineConstraints* for the finding of appropriate constraints, and *calculate* for the use of appropriate formulas.

For EX1, the value of difficulty is given by the three elements:

<difficulty value= "easy" for_skill= "modelling"/>,
<difficulty value="medium" for_skill= "determineConstraints"/>,
<difficulty value="medium" for_skill="calculate"/>

To be efficient, these addings of metadata must come with addings in the learner model. This one must contain skills to allow the search procedure to link these skills and those of the exercise description. Skills are more precise than competencies, but they are domain dependent and thus the kind of learner model is modified regarding ActiveMath purpose. To update the learner model after the solving of an exercise, a report is sent to ActiveMath which would be enriched by the analysis of the solving tracks.

The propositions to add skills to the description of the exercises and to the learner model and to explicitly link skill and difficulty fit in with the position of the PISA group when they say that their *competencies* describe skills at a very high level and must be specified to describe precise skill in a restricted domain.

3 Conclusion

To make two separately designed systems interact is always difficult. Our aim was to take advantage of the two systems. On the one hand Activemath capacity to choose a "right" learning object according to the user model and the domain to learn and, on the other hand, of the wealth of the training Combien? system. Both systems use metadata to describe and choice learning objects but these are not of the same level because of the restricted domain of Combien?

We have been confronted with a problem of a difference of level between the two metadata systems and we have been able to join these systems with respect to the Activemath features. We think that this study can open new perspectives to incorporate specialized complex activities in a general web learning environment.

References

1. ActiveMath, `http://www.activemath.org`
2. Brooks, C., Greer, J., Melis, E., Ullrich, C.: Combining ITS and eLearning Technologies: Opportunities and Challenges. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 278–287. Springer, Heidelberg (2006)
3. Combien? `http://combien.lip6.fr`
4. Le Calvez, F., Giroire, H., Tisseau, G.: Design of a Learning Environment in Combinatorics based on Problem Solving: Modeling Activities, Problems and Errors. International Journal of Artificial Intelligence in Education 18(1), 59–94 (2008)
5. Morales, R., van Labeke, N., Brna, P.: Approximate Modelling of the Multi-dimensional Learner. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 555–564. Springer, Heidelberg (2006)

Semantic Web Reasoning Tutoring Agent

Christiana Panayiotou and Brandon Bennett

School of Computing,
University of Leeds,
UK
{brandon,cpaa}@comp.leeds.ac.uk
http://www.comp.leeds.ac.uk

Abstract. This paper proposes a proof theoretic approach to check conflicts between the arguments derived from the ontologies of learning resources. Two types of arguments that can arise in a situation where a learner encounters conflicting viewpoints about a topic are identified, namely syllogistic arguments and arguments about the set of necessary and sufficient conditions that represent a concept. A method based on set equations is applied to create Syllogistic arguments from ontologies. The taxonomic associations of concepts in Ontologies, can be converted to categorical statements giving rise to syllogisms. We also consider arguments about the necessary and sufficient features for the representation of concepts and show that they can be handled in a very similar way as syllogistic arguments. The approach can be applied by a pedagogical agent in an interactive learning environment in order to identify, discuss differences in conceptualizations and check the validity of claims of different resources.

1 Introduction

Ontologies enable the representation of the semantic content of learning resources. With the growing release of semantic web tools and technologies these resources are becoming increasingly available to the learners. However, the existence of multiple resources gives rise to multiple perspectives and viewpoints about the same topics. Empirical research [1] has recorded evidence of discrepancies in conceptualizations of experts attributed to different background knowledge and practices. The educational community [2,3,4] argues that the use of argumentation in order to identify discrepancies in conceptualizations enables reflection and articulation and can enrich the learning experience. The purpose of this paper is to discuss our approach for the derivation of arguments from ontologies and for testing the validity of arguments raised by a learner. We focus on two types of arguments, namely *syllogistic* arguments and *necessity and sufficiency* arguments, which are important for learning. The rest of the paper is organized as follows: Section 2 shows how syllogistic arguments and arguments about necessary and jointly sufficient features can be inferred from ontologies and validated via a theorem prover. Section 3 summarizes the issues raised in this paper and the future objectives.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 816–818, 2008.

2 Formalization

2.1 Terminology

We define an *ontology* O as a structure $O \equiv \langle \mathcal{C}, D, R \rangle$, where C is a set of concepts, D is a non-empty set called the domain of the ontology and R is a set of relations over $\mathcal{C}$. An *interpretation* I of O is an assignment associating to each concept $C_i \in \mathcal{C}$ a non-empty subset, ${C_i}^I$, of D, and to each relation in R a subset of $\mathcal{C} \times \mathcal{C}$. We define the *model*, $\mathcal{M}$ of an ontology O, as an interpretation that satisfies each relation in R. An Ontology may include one or more hierarchies of concepts. These are important for the derivation of arguments considered in this paper. We use the term *concept hierarchy* to denote the structure $\mathcal{H} =< C_{\mathcal{H}}, R_{\mathcal{H}} >$ where $C_{\mathcal{H}}$ is a set of concepts, st. $C_{\mathcal{H}} \subseteq C$ of the ontology O, and $R_{\mathcal{H}} = \{Disjoint, SubclassOf, InstanceOf, Union, Intersects\}$ and every concept in $C_{\mathcal{H}}$ is associated with another concept via a relation in $R_{\mathcal{H}}$. As above, we are interested in those interpretations of a hierarchy that satisfy all the taxonomic relations within the hierarchy. A model, $\mathcal{M}_{\mathcal{H}}$ of $\mathcal{H}$ is an interpretation I of $\mathcal{H}$ where all the taxonomic relations in $R_{\mathcal{H}}$ are satisfied. Obviously, $\mathcal{M}_{\mathcal{H}}$ is a sub-model of $\mathcal{M}$.

2.2 Syllogistic Arguments and Ontological Taxonomic Relations

Generalized statements of the form: *Every X is a Y* or *Every X has the property Y* are referred to as *categorical propositions* . A *syllogism* is a particular type of argument that always has two premises and a single conclusion and all three statements are categorical propositions [5]. There are four basic categorical propositions that can be combined to produce 64 patterns of Syllogistic Arguments. However, only 27 of them are valid. As a result, testing validity of syllogistic arguments is important in any interaction with the tutor. The four basic categorical propositions are shown below, together with their ontological representations and set equations. [1]

Categorical Statement	Ontological Primitive	Set Equation
Every S is a P	SubclassOf(S, P)	$S \subseteq P$
No S is a P	SubclassOf(S, ComplementOf(P))	$S \subseteq \overline{P}$
Some S is a P	Intersects(S, P)	$S \cap P$
Some S is not P	Intersects(S, ComplementOf(P))	$S \cap \overline{P}$

2.3 Necessary and Sufficiency Conditions Arguments

We use the classical view of the representation of concepts which states that the features representing a concept are *singly necessary* and *jointly sufficient* to define a concept. Intuitively, a feature ϕ is *singly necessary* for the definition of C if and only if existence of C implies existence of ϕ. More formally, let us assume a feature ϕ. We define a set Φ consisting of all items of the domain which

[1] For simplicity, we use the concepts themselves instead of the actual interpretation of concepts in set equations.

have feature ϕ. Then, ϕ is necessary for the representation of concept C if and only if $C^I \subseteq \Phi$. Now assume that $\{\Phi_1, \ldots, \Phi_n\}$ represent the set of concepts corresponding to features $\phi_1, ..., \phi_n$ respectively. Then if $\{\Phi_1 \cap, \ldots, \cap \Phi_n\} \subseteq C^I$ we say that $\phi_1, ..., \phi_n$ are jointly sufficient for the definition of C. So, now we can easily derive the universal set equations corresponding to above the notions of sufficient and necessary features for the representation of concepts.

2.4 Checking validity of Arguments

Bennett [6] proved that universal equations can be translated to equivalent propositional logic formulae that can be tested for their validity with a propositional theorem prover. By adapting the *classical entailment correspondence theorem* [6] we get an equivalent result for taxonomic relations as follows: $\mathcal{M}_{\mathcal{H}} \models \phi$ iff $M_{C+} \models \tau = \mathcal{U}$, where $\phi \ _{CF} \rightleftharpoons^{ST} \tau$ for each i and $C+$ is the universal set equations language.

3 Conclusion

In this paper we show how a tutoring agent can infer arguments from ontologies. Arguments can be used to identify discrepancies in the ontologies of learning resources and differences between ontologies and a learner's beliefs. We show that syllogistic arguments follow naturally from ontological primitives and we represent arguments about the necessary and sufficient properties of concepts. The latter help to differentiate between concepts. Concepts and their associations are given set-theoretic semantics and can be translated to universal set equations. Bennett's [6] theory is used to convert universal set equations to propositional logic statements that can be checked for their validity with a propositional logic theorem prover. The theorem prover can also be extended to infer arguments from an ontology automatically.

References

1. Hameed, A., Sleeman, D., Preece, A.: Detecting mismatches among experts ontologies acquired through knowledge elicitation. Knowledge-Based Systems 15, 265–273 (2002)
2. Ravenscroft, A.: Designing argumentation for conceptual development. Computers and Education 34, 241–255 (2000)
3. McAlister, S., Ravenscroft, A., Scalon, E.: Combining interaction and context design to support collaborative argumentation using a tool forsynchronouscmc. Journal of Computer Assisted Learning 20(3), 194–204 (2004)
4. Hartley, J.R.: Qualidative reasoning and conceptual change: Computer based support in understanding science. Interactive Learning Environments 5, 53–64 (1998)
5. Walton, D.: Foundamentals of Critical Argumentation. Cambridge University Press, Cambridge (2006)
6. Bennett, B.: Logical Representations for Automated Reasoning about Spatial Relationships. PhD thesis, School of Computer Studies, The University of Leeds (1997)

An Affective Behavior Model for Intelligent Tutors

Yasmín Hernández[1], Enrique Sucar[2], and Cristina Conati[3]

[1] Instituto de Investigaciones Eléctricas, Gerencia de Sistemas Informáticos
Cuernavaca, Morelos, México
myhp@iie.org.mx
[2] Instituto Nacional de Astrofísica, Óptica y Electrónica, Coord. Ciencias Computacionales
esucar@inaoep.mx
[3] University of British Columbia, Department of Computer Sciences
Vancouver, B.C., Canada
conati@cs.ubc.ca

Abstract. We describe an affective behavior model (ABM) for intelligent tutoring systems. The model is a Dynamic Decision Network that selects tutorial actions based on both the current affective and pedagogical state of a student, as well as on the assessment of the expected effect of each available action on the student. We integrated the ABM with an educational game to learn number factorization, and here we present the preliminary results of a user study to evaluate its effectiveness.

Keywords: affective student model, Bayesian networks, dynamic decision networks, intelligent tutoring systems.

1 The Affective Behavior Model (ABM)

The ABM is designed to enable intelligent tutoring systems to include affective responses in their pedagogical actions. A diagram of the ABM is shown in Fig. 1. The ABM relies on both a model of a student's current knowledge (*pedagogical model* in Fig. 1) and a model of student affect (*affective model* in Fig. 1) to select an affective and a pedagogical action to support student learning and morale in the current situation. The two actions are then integrated into the actual tutorial action delivered to the student through the interface module. The ABM is basically a model that allows an ITS to establish a mapping from a student's affective and pedagogical state to tutorial actions. The mapping cannot be deterministic because there is inherent uncertainty in the assessment of both the current relevant student states and the effects of tutor's actions on these states. To deal with this uncertainty, the ABM relies on a dynamic decision network (DDN), depicted in Fig. 2. The DDN generates a probabilistic assessment of how each available tutorial action influences the affective and pedagogical state of the student, given a probability distribution over his/her current state. This assessment is then used to establish the expected utility of each tutorial action for the current state. The DDN selects the tutorial action considering two utility measures, one on learning and one on affect, which are combined to obtain the global utility of

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 819–821, 2008.

each available action for the tutor's goals. The influence of each tutorial action on the pedagogical and affective states is based the teachers' expertise, as we describe next. For a more detailed description of the model, see [2].

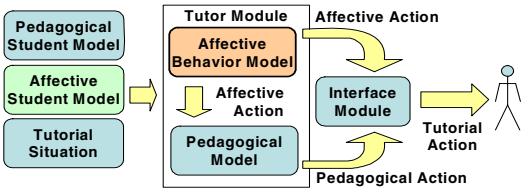


Fig. 1. General diagram of the affective behavior model

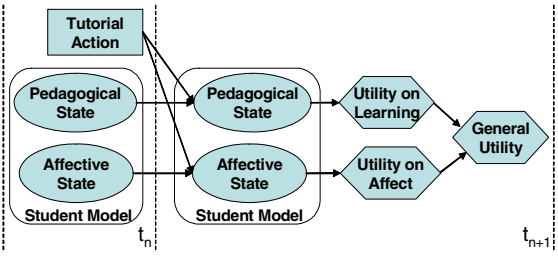


Fig. 2. High level dynamic decision network for the affective tutor model

2 Evaluation of the Model

We integrated the ABM with Prime Climb, an educational game to learn number factorization for grade 6 and 7 students. This game includes Merlin, a pedagogical agent implemented through Microsoft Agent [4], as well as Bayesian models for both student affect [1] and learning [3]. The agent in the original game does not explicitly consider affective factors in its decisions, i.e. it does not rely on the affective student model and only generate pedagogical actions in the form of verbal hints appearing in a speech bubble; for example: "Think about how to factorize the number you clicked on" or "You cannot click on a number which shares common factors with your partner's number". To deliver the pedagogical hints Merlin takes a fixed, neutral pose and facial expression. Therefore, we devised a new version of the game that includes the ABM and uses it to select affective actions in addition to select pedagogical actions. Affective actions are animations of the pedagogical agent, such as Merlin making a conciliating expression and extending his arms trying to explain and motivate the student. To decide which affective/pedagogical actions to include in the system and define their impact on the student state, we relied on teachers' expertise. Eleven teachers were shown the various animations and verbal hints available to Merlin, as well as a video of a student playing the game. Based on this information, they selected the animations and verbal hints they thought were most suitable for the Prime Climb agent. They also established a mapping between the various playing situations shown

Fig. 3. Two tutorial actions composed by an affective action (Merlin's animation) and a pedagogical action (verbal hint) selected by the affective behavior model for Prime Climb

in the video and what they thought were the most suitable agent actions for these circumstances. Fig. 3 shows two tutorial actions composed by an affective action, (Merlin's animation) and a pedagogical action (a verbal hint).

We conducted a user study in a school in Mexico with students from grades equivalent to grades 6, 7 and 8 in elementary school in the American system. The students in the lowest grade had just learned number factorization. Students in the higher grades were supposed to know this topic already, but their teachers thought it would still be useful for them to use Prime Climb as a review. For each grade, the students were divided into two groups; the control group played Prime Climb with the original pedagogical agent, the experimental group played with the ABM version. We gave each student a pre-test, then the students played for 40 minutes, and after that they took a post-test and a questionnaire. We found a significant effect of game version on post-pre test gain for the students in grade 6 (1-tailed t-test, $t = 6.95$, $p < 0.001$), with the experimental group learning more. This result shows that the ABM has great potential to improve an ITS's performance by including affective factors in its tutorial decisions. For the higher grades, neither groups showed significant improvements from pre-test to post-test. For the highest grade this is due to a ceiling effect in the pre-test, but for the intermediate grades, the pre-tests showed that students still needed help with number factorization. We speculate these students did not learn from Prime Climb as much as the younger students did because they did not put effort into it, believing that they had already mastered the topic. However, further studies are necessary to clarify this finding.

References

1. Conati, C., Maclaren, H.: Data-driven Refinement of a Probabilistic Model of User Affect. In: Ardissono, L., Brna, P., Mitrović, A. (eds.) UM 2005. LNCS (LNAI), vol. 3538, pp. 40–49. Springer, Heidelberg (2005)
2. Hernández, Y., Sucar, L.E.: User study to evaluate an affective behavior model in an educational game. In: 13th International Conference on Artificial Intelligence in Education, AIED 2007, Workshop on Modeling and scaffolding affective experiences to impact learning, Marina del Rey, California, EU, July 9-13, pp. 57–66 (2007)
3. Manske, M., Conati, C.: Modelling Learning in an Educational Game. In: Looi, Ch., McCalla, G., Bredeweg, B., Breuker, J. (eds.) Proceedings of 12th International Conference n Artificial Intelligence in Education, AIED 2005, Amsterdam, The Netherlands, July 19-23, pp. 411–418 (2005)
4. Microsoft Corporation, Microsoft Agent, Web site, `http://www.microsoft.com/msagent/default.asp` (Retrieved on November 2005)

Decision Tree for Tracking Learner's Emotional State Predicted from His Electrical Brain Activity

Alicia Heraz, Tariq Daouda, and Claude Frasson

HERON Lab, University of Montréal, CP 6128 succ. Centre Ville
Montréal, QC, H3T-1J4, Canada
{herazali,daoudata,frasson}@iro.umontreal.ca

Abstract. This paper proposes the use of machine learning techniques to build an efficient learner's emotional transition diagram transition. For information Extraction tasks, we led an experimentation in which we exposed a group of 17 learners to a series of pictures from the International Affective Picture System (IAPS). Decision tree classifier has demonstrated the best ability to learn model structure from data collected. Among the emotions involved in learning and according to the picture from IAPS and the current emotional state, we drew up the transition diagram. Our model aims to improve the task of predicting the emotional state in an Intelligent Tutoring System and achieve a prediction accuracy of 63.11%. These results suggest that the implementation of the decision tree algorithm in the intelligent tutoring system we are developing improves the ability for an ITS to track the learners emotional states.

1 Previous Work

Our previous work [2][3] indicated that an EEG is an efficient information source to detect emotions. Results show that the student's affect (Anger, Boredom, Confusion, Contempt, Curious, Disgust, Eureka, and Frustration) can be accurately detected (82%) from brainwaves [3]. We have also conducted an experimentation in which we explored the link between brainwaves and emotional assessment on the SAM scale (pleasure, arousal and domination). Results were promising, with 73.55%, 74.86% and 75.16% for pleasure, arousal and dominance respectively [2]. Those results support the claim that all rating classes for the three emotional dimensions (pleasure, arousal and domination) can be automatically predicted with good accuracy through the nearest neighbor algorithm. In this paper, we focus on the emotions that are exhibited during learning [1] and we identify them using classification methods. We use the International Affective Picture System to present to the learner different pictures able to trigger emotions; and we aim to estimate the average duration of each emotion and the effect of the visual emotional stimuli. Brainwaves are categorized into 4 different frequency bands, or types, known as delta, theta, alpha, and beta waves. Each of these wave types often correlates with different mental states. Table 1 lists the different frequency bands and their associated mental states.

The emotions that we will consider are those which appear during learning [1] and are: anger, boredom, confusion, contempt, curiosity, disgust, eureka and frustration.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 822–824, 2008.

Table 1. Brainwaves Categories

Wave Type	Frequency	Mental State
Delta (δ)	0-4 Hz	Deep sleep
Theta (θ)	4-8 Hz	Creativity, dream sleep, drifting thoughts
Alpha (α)	8-12 Hz	Relaxation, calmness, abstract thinking
Beta (β)	+12 Hz	Relaxed focus, high alertness, agitation, anxiety

2 Experimentation Description and Results

The experiment included 17 learners selected from the Computer Science Department of University of Montreal. In order to induce the emotions which occur during learning, we use IAPS. The participant is connected to an EEG. The purpose of the experimentation is to record any change related to the emotions or the brainwave amplitude. The resulting database was composed of 30551 tuples that contained the user id, the id of the picture displayed (from IAPS). the emotion of the user and the time (in $hh:mm:ss$) of the recording. The first treatment that was applied to the database was to extract a dataset of tuples that contains the picture id, the emotional transition from emotion e_t to emotion $e_{t+\Delta t}$, and the time, Δt (in sec) between each emotional or picture transition. Tuples with Δt values equal to 0 where removed as well as the few ones that had their Δt values scattered between 55 and 206. We eliminated categories of less than 6. Several classification algorithms were tested in order to provide the best accuracy (table 2). Classification accuracy varies from 60.85% (+/- 3.88%) to 63.12% (+/- 3.12%).

Table 2. Results of the algorithms that gave the best results

Algorithm	Accuracy	Kappa
Decision tree	63.11% +/- 3.12%	0.534 +/- 0.038
Random Forest	63.12% +/- 5.72%	0.532 +/- 0.079
CHAID	62.85% +/- 4.21%	0.529 +/- 0.057
Rule learner	60.85% +/- 3.88%	0.508 +/- 0.055

Figure 1 shows the results obtained from the Decision tree algorithm. Starting from emotion e1, for instance with emotion 2, we observe that we shift to emotion 6 if the duration (Δt) is higher than 7 seconds. We can observe that the average duration of feeling a particular emotion is estimated between 6 and 9 seconds. An interesting observation is that the picture category seems to be a relevant parameter only in the case of transitions from disgust. In the case of transitions from other emotions, the only parameter kept by the algorithm was Δt.

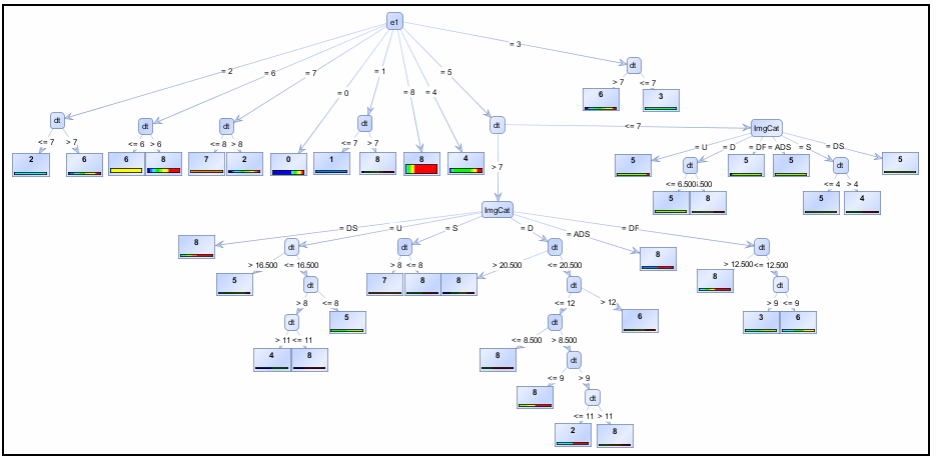


Fig. 1. The decision tree showing the transitions between emotions

3 Conclusion

It appears that there the learner's emotional transition depends always on the duration and, in the case of disgust, also on the picture category. Therefore, results show that we can follow the emotional transitions of the learner and that certain picture categories can influence emotions. The output of decision tree resulted in accurate predictions. If the method described above proves to be effective in tracking the learner's emotions, we can direct our focus to a second stage. Using this prediction, an ITS could select an adequate pedagogical strategy able to cope to certain learner's emotions in addition to cognitive states. This adaptation would increase the bandwidth of communication and allow an ITS to respond at a better level. If this hypothesis holds in future replication, then it would give indications on how to help those learners to induce positive emotions during learning.

References

1. D'Mello, S.K., Craig, S.D., Gholson, B., Franklin, S., Picard, R.W., Graesser, A.C.: Integrating Affect Sensors in an Intelligent Tutoring System. In: Affective Interactions: The Computer in the Affective Loop Workshop at 2005 International conference on Intelligent User Interfaces, pp. 7–13. AMC Press, New York (2005)
2. Heraz, A., Frasson, C.: Predicting the three major dimensions of the learner's emotions from brainwaves. In: 4th International Conference on Computational Intelligence and Cognitive Informatics: CICI 2007, Venise, Italy (2007)
3. Heraz, A., Razaki, R., Frasson, C.: Using machine learning to predict learner emotional state from brainwaves. In: 7th IEEE conference on Advanced Learning Technologies: ICALT 2007, Niigata, Japan (2007)

Toward Supporting Collaborative Discussion in an Ill-Defined Domain

Amy Ogan, Erin Walker, Vincent Aleven, and Chris Jones

Carnegie Mellon University, Pittsburgh, PA, USA
aeo@andrew.cmu.edu, erinwalk@andrew.cmu.edu, aleven@cs.cmu.edu, cjones@andrew.cmu.edu

Abstract. It is important to investigate what tutoring paradigms are appropriate for ill-defined domains. We focus on teaching intercultural competence, where asynchronous online discussion is a typical instructional task. We explore two methods of assessing student contributions and delivering feedback: an adaptive support system that provides individual feedback, and a peer moderator from the class supported by adaptive assistance.

1 Introduction

Intercultural competence, where students learn to interpret events in a foreign culture in terms of cultural differences, is often taught by presenting students with cultural video and then asking them to discuss the relevant issues. As more and more in-class discussion moves online to asynchronous discussion forums, teachers are not always available to support the discussion [1]. We are interested in applying the techniques of intelligent tutoring systems (ITSs) to address this problem. ITSs have made significant strides in well-defined domains such as math, but approaches for ill-defined domains, such as the learning of culture, are now just emerging. We leverage two resources in delivering discussion support: an intelligent autonomous system and fellow students from the class. Our work attempts to answer the following two questions. First, where can adaptive support be best applied in asynchronous discussion on intercultural competence? Second, how does providing feedback through these two resources affect discussion posts?

To teach intercultural competence, we use an e-learning environment developed by Ogan et al. [2], where students watch a video clip from a French film that includes cultural content (e.g., what immigration means to France), and follow exercises that encourage them to reflect deeply on the content. After this process, students engage in discussion, which we attempt to support in two ways. First, we develop an adaptive collaborative learning system *(ACLS)* that compares a given student post to a simple model of good intercultural discussion, and provides private feedback to students based on the model (*individual ACLS*). In the second approach, adaptive support is delivered to a peer moderator, a student from the class who moderates the discussion forum (*moderator ACLS*). In this work, we developed a prototype of each approach and piloted the prototypes with a small number of students.

B. Woolf et al. (Eds.): ITS 2008, LNCS 5091, pp. 825–827, 2008.

2 Design and Implementation

To form a basis for adaptive support, we used theoretical and empirical analyses to identify five dimensions of good cultural discussion. The first two are specific to cultural discussions, while the remaining three are general and we consider them prerequisite for cultural discussion. The first dimension is taken from Steglitz [3] who includes intercultural perspective taking as an important measure of intercultural learning (*D1: Does the post show awareness of multiple points of view?).* Next, our data revealed that students did not always post correct facts to the forum, thus suggesting *D2: Does the post reference correct cultural elements specific to the theme of the video?* Moving to general properties of good discussion, on-topic discussion should enable learning [4] (*D3: Is this post on-topic?)*. Next, students should post at a high level of "cognitive depth" (e.g., [5]), manifested by *D4: Does the post have good argumentation?* Finally, our existing forum data revealed that discussions benefited from having students introduce novel facts, leading to *D5*: *Does this post introduce relevant new facts?*

We designed the individual and moderator adaptive support based on the above dimensions. To implement the individual ACLS, we used automated key-word detection to achieve reasonable accuracy in identifying three of the five dimensions: *D1, D3,* and *D4*. After students created a post for the discussion board, they had to submit it to the system prior to posting. The system determined where the post fell along each dimension and randomly selected one feedback message from several with alternate wordings that were associated with the problematic dimensions. The system then added correct facts related to the content of the post from a collection of facts about the issues presented in the video. This feedback was presented privately to the student, who was required to make at least one modification to the post before submitting the final version to the discussion board. The following is an example of a final post:

> "Maybe Momo is a little arrogant, and M Ibrahim wants to "put him in his place" a little. If his father is so racist (seen by saying go to chez l'arabe) it's possible that Momo will imitate him. It is a little kind and nice of him to give a name a little funny or cute so that he learns that he is not superior to others. Also, the computer suggests that I add 'in the past, names given to infants born in France were required to be the names of Catholic saints, and often, Jewish families changed their names to better conform to the dominant culture.' So maybe Ibrahim wants Momo to follow his roots".

In our moderator intervention, two students from the class were chosen to moderate each assignment. They were asked to reply to threads with feedback on the posts that had been made so far and guide the discussion. Moderators were asked to rate posts on a binary scale ("yes/no") for each of the five model dimensions and submit them to the moderator support agent. The adaptive support gave the moderator a feedback template to fill in and suggested some facts that the moderator might want to incorporate into his or her post. The moderator then used the template to write and submit a post to the discussion forum. The following is an example of a feedback template:

> "You make a good argument when you say _____. But have you considered looking at the issue from a different perspective? Maybe what is happening is _____"

3 Evaluation

We conducted a pilot evaluation of the two types of support in two different upper level French classes. Both studies took place over a week, during which students watched a film clip and were asked to post to the discussion board three times. The individual ACLS forum had 8 participants, while the moderator ACLS forum had 3 participants and 2 moderators. We assessed both the quality and effect of feedback provided by each approach. In general, moderators deleted the feedback template and wrote their own feedback, often asking questions about specific issues in the video. Their posts contributed to the discussion with multiple perspectives, introduction of prior knowledge, and good argumentation. The feedback given by the individual ACLS offered suggestions for improvement along the dimensions and presented the students with relevant facts. A majority of the system ratings matched human ratings of the posts. When posts received less accurate ratings, we observed that the feedback was not out of place, perhaps because it was designed to apply in multiple situations. We then asked whether students used the feedback to improve the discussion. After feedback from the peer moderator, subsequent student posts tended to improve, although the small number of participants warrants further study. In the individual ACLS, students used the feedback they received to modify 15 out of the 25 posts. The majority of these posts improved, mostly on using correct facts. These facts were suggested in the feedback and may have also lead to the improvement seen in argumentation scores, because they provided better evidence to support students' conclusions.

In this work, we have created prototypes applying adaptive support technology to the ill-defined domain of intercultural competence. We see promise both in providing private adaptive feedback to posters in a discussion forum (through post improvement) and to a peer moderator of the discussion (through richer feedback). While further evaluation is necessary, these approaches could improve intercultural competence instruction and change our conception of asynchronous discussion support.

Acknowledgements. This research is supported by the Graduate Training Grant given to Carnegie Mellon University by the Department of Education (#R305B040063).

References

1. Beaudin, B.P.: Keeping online asynchronous discussions on topic. Journal of Asynchronous Learning Network 3(2), 41–53 (1999)
2. Ogan, A., Aleven, V., Jones, C.: Pause, predict, and ponder: Use of narrative videos to improve cultural discussion and learning. In: Proceedings of Computer Human Interaction conference (CHI 2008) (to appear, 2008)
3. Steglitz, I.: Intercultural perspective-taking: The impact of studying abroad. Unpublished dissertation. University of Minnesota (1993)
4. Guzdial, M., Turns, J.: Effective discussion through a computer-mediated anchored forum. Journal of the Learning Sciences 9, 437–469 (2000)
5. Hara, N., Bonk, C., Angeli, C.: Content analyses of on-line discussion in an applied educational psychology course. Instructional Science 28(2), 115–152 (2000)

Author Index

Abdessemed, Amir 312
Adam, Jean-Michel 480
Ahmad, Norasnita 722
Aïmeur, Esma 438
Aleahmad, Turadg 216
Aleven, Vincent 90, 216, 406, 693, 807, 825
Alimi, Adel. M 665
Almeida, Shane F. 766
Anwar, Mohd 681
Arromratana, Aniwat 343
Arroyo, Ivon 29
Arruarte, Ana 728
Ashley, Kevin 90
Atkinson, Robert 731
Azevedo, Roger 260, 690

Baker, Ryan S.J.d. 40, 406
Balla, Amar 740
Barnes, Tiffany 373
Barrón-Estrada, M.L. 746
Barrow, Devon 250
Bartlett, Marian 668
Bateman, Scott 563
Beck, Joseph E. 122, 353, 383, 766, 774
Ben Ammar, Mohamed 665
Bennett, Brandon 816
Biswas, Gautam 614
Blank, Glenn D. 291
Blessing, Stephen B. 204
Boicu, Mihai 228
Bolge, Eleanor 779
Bolster, Thomas 593
Bourdeau, Jacqueline 573
Bousbia, Nabila 740
Boyer, Kristy Elizabeth 239
Braun, Isabel 709
Britland, Mark 674
Brown, Christopher 80
Brown, Quincy 693
Bull, Susan 132, 674, 722
Burleson, Winslow 29

Cade, Whitney L. 470
Callan, Jamie 500
Cen, Hao 796
Cerri, Stefano A. 696
Chan, Tak-Wai 152
Chang, Kai-min 383
Chaudhuri, Sourish 793, 807
Chenevotot, Françoise 101
Chi, Min 603
Chou, Chih-Yueh 152
Cohen, William W. 111
Collins, Allan M. 1
Conati, Cristina 819
Copeland, Jessica L. 470
Corbett, Albert T. 383, 406
Courtemanche, François 312, 719
Cui, Yue 784, 790, 793

D'Mello, Sidney K. 9, 40, 50, 260, 470
Daouda, Tariq 822
Davidson, Kelly 9
De Bra, Paul 771
Delozanne, Élisabeth 101
de Menezes, Ilusca Lima Lopes 787
Di Eugenio, Barbara 80
Dong, Xiaoxi 674
Dragon, Toby 29
Dubois, Daniel 803
Dubois, Michel 480
Dufresne, Aude 551, 810
Dugénie, Pascal 696

Eberspächer, Henri 734
Eichelmann, Anja 416
el Kaliouby, Rana 29
Elorriaga, Jon A. 728
Eskenazi, Maxine 500, 656
Eydgahi, Hoda 29

Faulhaber, Arndt 416
Feeney, Christine M. 659
Forbes-Riley, Kate 60
Fortin, Mikaël 312
Fossati, Davide 80
Fournier-Viger, Philippe 395
Frasson, Claude 448, 787, 822

Gaha, Mohamed 803
Gal, Ya'akov 162
Gašević, Dragan 563
Gheorghiu, Roxana 749
Gilbert, Stephen 204
Giroire, Hélène 813
Goguadze, George 755
Gonzalez, Ma. Celeste T. 40
Goth, Julius 510
Gouaich, Abdelkader 696
Gouardères, Guy 665
Graesser, Art 9
Greer, Jim 681
Grimley, Michael 250, 281
Grosz, Barbara J. 162
Groß, Andreas 662
Grugeon, Brigitte 101
Guerdelli, Fethi 551
Guin, Nathalie 699
Guo, Yu 674, 774
Gupta, Amit 614

Ha, Eunyoung 510
Haddawy, Peter 583
Hage, Hicham 438
Harrer, Andreas 709, 715
Hatala, Marek 563
Hausmann, Robert G.M. 636
Hayashi, Yusuke 573
Heffernan, Neil T. 426, 766, 774
Heilman, Michael 500, 656, 659
Heraz, Alicia 822
Hernández, Yasmín 819
Higgs, Fred 807
Hirashima, Tsukasa 687
Hotte, Richard 702

Isotani, Seiji 646, 752

Jean-Daubias, Stéphanie 699
Jeon, Moongee 690
Jeong, Hogyeong 614
Jia, Jiyou 706
Joab, Michelle 734
Johnson, W. Lewis 270, 520
Johnston, Lucy 19
Jones, Chris 825
Jordan, Pamela 671
Joshi, Mahesh 807
Jovanović, Jelena 563
Jung, Sung-Young 758
Junker, Brian 796

Kang, Moonyoung 793
Kay, Judy 3
Kazi, Hameedullah 583
Kerly, Alice 132, 722
Kim, Jihie 343
Kloos, Carlos Delgado 540
Koedinger, Kenneth 111, 593, 626, 677, 790, 796
Kraut, Robert 216
Kulkarni, Anagha 500
Kumar, Amruth 799
Kumar, Rohit 790, 793, 807

Labat, Jean-Marc 702, 740
Lacerda, Gustavo 111
Lagud, Maria C.V. 40
Larrañaga, Mikel 728
Le Calvez, Françoise 813
Le, Vu 228
Lebeau, Jean-François 312
Lee, Frank J. 693
Lee, Sunyoung 530
Lefevre, Marie 699
Lehman, Blair 50
Lemoisson, Philippe 696
Lepp, Marina 70
Lesgold, Alan M. 6
Lester, James C. 239, 490, 510, 530
Lim, Sheryl A.L. 40
Lim, Sung-Joo 677
Lin, Chi-Jen 152
Litman, Diane 60, 459, 671
Looi, Chee-Kit 302
Lopez-Garate, Maite 737
Lozano-Rodero, Alberto 737
Lynch, Collin 90

Macapanpan, Alexis F. 40
MacWhinney, Brian 593
Madhour, Hend 725
Magoulas, George 142
Marino, Olga 702
Martin, Brent 194, 684
Martin, James H. 173
Matey, Luis 737
Mathews, Moffat 363
Matsuda, Noboru 111

Matthews, Melanie 50
Mayers, André 312, 719
McLaren, Bruce M. 323, 333, 677, 709, 715
McQuiggan, Scott W. 490, 510, 530
Meinel, Christoph 662
Melis, Erica 416, 755
Mendicino, Michael 426
Mephu Nguifo, Engelbert 395
Miksatko, Jan 333
Milik, Nancy 281
Mitrović, Tanja 363
Mitrovic, Antonija 19, 194, 250, 281
Mizoguchi, Riichiro 573, 646, 752
Moguel, Patrice 743
Mostow, Jack 122, 353, 383
Moulet, Lucie 702
Movellan, Javier 668
Muñoz Merino, Pedro J. 540

Najjar, Mehdi 719
Narciss, Susanne 416
Neji, Mahmoud 665
Nedji Milat, Iness 712
Nielsen, Rodney D. 173
Nkambou, Roger 395, 448, 803

Ogan, Amy 825
Ohlsson, Stellan 80, 250
Okamoto, Masahiko 687

Panayiotou, Christiana 816
Parvez, Shahida M. 291
Pascua, Sheila A.M.S. 40
Patvarczki, Jozsef 766
Pavlik Jr., Philip 593
Pechenizkiy, Mykola 771
Pelczer, Ildikó 763
Penstein Rosé, Carolyn 784, 790, 793, 807
Person, Natalie K. 50, 470
Phillips, Robert 239
Pinkwart, Niels 90, 709, 715
Poirier, Pierre 803
Poulovassilis, Alexandra 142
Prévit, Dominique 101

Ramachandran, Sowmya 731
Ravi, Sujith 343
Razzaq, Leena 426
Repp, Stephan 662
Reyes-García, Carlos A. 746
Robison, Jennifer L. 490
Rodrigo, Ma. Mercedes T. 40
Rodríguez, Fernando Gamboa 763
Rosé, Eric R. 793
Roscoe, Rod 614
Rotaru, Mihai 60
Rouatbi, Mohamed 551
Rowe, Jonathan P. 510, 530
Ruan, Meixian 706
Rubin, Andee 162
Rummel, Nikol 626, 709

Salvucci, Dario D. 693
Sandoval-Sánchez, Guillermo A. 746
Santillano, Jerry Q. 40
Sarda, Pankaj 343
Scheuer, Oliver 323, 709, 715
Schwartz, Daniel 614
Sellami, Mokhtar 712
Seridi, Hassina 712
Sewall, Jonathan 111
Shaw, Erin 343
Shieber, Stuart M. 162
Sison, Raymund 184
Stamper, John 373
Suarez, Merlin 184
Sucar, Enrique 819
Suebnukarn, Siriwan 583
Sugay, Jessica O. 40
Sukkarieh, Jana 779

Tajariol, Federico 480
Takeuchi, Akira 687
Tavano, Erin 343
Taylor, Roger 9
Tchounikine, Pierre 743
Tecuci, Gheorghe 228
Tep, Sinath 40
Terrell, Elon 807
Torniai, Carlo 563
Tricot, André 743
Tsovaltzi, Dimitra 709

van de Sande, Brett 636
Van Labeke, Nicolas 142
VanLehn, Kurt 7, 603, 636, 749, 758
Vasilyeva, Ekaterina 771
Vassileva, Julita 8

Viehland, Norma J.B. 40
Villiot-Leclercq, Emmanuelle 810
Vouk, Mladen 239

Wagster, John 614
Walker, Erin 626, 825
Wallis, Michael 239
Wang, Ning 270
Wang, Yi-Chia 793
Ward, Arthur 459
Ward, Wayne 173
Wentland Forte, Maia 725
Whitehill, Jacob 668
Witherspoon, Amy M. 260
Woolf, Beverly P. 29
Wu, Longkai 302
Wu, Shumin 520
Wu, Sue-mei 593

Yamangil, Elif 162
Yokoyama, Takuro 687

Zakharov, Konstantin 19
Zatarain, Ramón 746
Zhang, Xiaonan 122
Zouaq, Amal 448