LNAI 7466

Belén Díaz Agudo
Ian Watson (Eds.)

Case-Based Reasoning Research and Development

20th International Conference, ICCBR 2012
Lyon, France, September 2012
Proceedings

Springer

Belén Díaz Agudo Ian Watson (Eds.)

Case-Based Reasoning Research and Development

20th International Conference, ICCBR 2012
Lyon, France, September 3-6, 2012
Proceedings

Springer

Series Editors

Randy Goebel, University of Alberta, Edmonton, Canada
Jörg Siekmann, University of Saarland, Saarbrücken, Germany
Wolfgang Wahlster, DFKI and University of Saarland, Saarbrücken, Germany

Volume Editors

Belén Díaz Agudo
Universidad Complutense de Madrid
Facultad de Informática
C/ Profesor José García Santesmases, s/n.
28040 Madrid, Spain
E-mail: belend@sip.ucm.es

Ian Watson
University of Auckland
Department of Computer Science
1142 Auckland, New Zealand
E-mail: ian@cs.auckland.ac.nz

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-32985-2 e-ISBN 978-3-642-32986-9
DOI 10.1007/978-3-642-32986-9
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012944835

CR Subject Classification (1998):
I.2.1, I.2.3-4, I.2.6, I.2.8, I.1.11, H.3.3-5, H.2.8, H.4, H.5.3, J.1, J.3

LNCS Sublibrary: SL 7 – Artificial Intelligence

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume contains the papers presented at ICCBR 2012: the 20th International Conference on Case-Based Reasoning (http://www.iccbr.org/iccbr12/) held September 3–6, 2012, in Lyon, France. There were 51 submissions to the conference; each was reviewed by at least three Program Committee members. The committee decided to accept 19 papers for oral presentation at the conference, following a highly selective process. An additional 13 papers were accepted for poster presentation. The program also contained two invited talks, the abstracts of which are included in this volume.

The International Conference on Case-Based Reasoning (ICCBR) is the preeminent international meeting on case-based reasoning (CBR). Previous ICCBR conferences have been held in Sesimbra, Portugal (1995), Providence, USA (1997), Seeon Monastery, Germany (1999), Vancouver, Canada (2001), Trondheim, Norway (2003), Chicago, USA (2005), Belfast, UK (2007), Seattle, USA (2009), Alessandria, Italy (2010), and most recently in Greenwich, UK (2011).

The first day of ICCBR 2012 was given over to an Introduction to CBR Development Tools involving an extensive look at the state-of-the-art tools myCBR and jCOLIBRI and products from the company Empolis that use CBR. Running in parallel on the first day was the 4th Annual Doctoral Consortium (DC) that involved presentations from 16 students in association with 19 mentors. A highlight of the DC were talks from guest speakers Agnar Aamodt and Santiago Ontañón.

The second day featured workshops on CBR in the Health Sciences, Process-Oriented CBR, and finally a workshop called TRUE: Traces for Reusing Users' Experiences — Cases, Episodes, and Stories. We would like to thank all the Cochairs of these workshops for creating such a stimulating program. The workshops were complemented with the popular full-day live Computer Cooking Contest (CCC). We would like to thank all involved with the CCC, the international jury, and the event's sponsors.

Days three and four comprised presentations and posters on technical and applied CBR papers, as well as invited talks from two distinguished scholars: Yolanda Gil, of the University of Southern California, USA, and Klaus-Dieter Althoff, of the University of Hildesheim, Germany. Yolanda Gill argued that integrating case-based reasoning techniques with scientific workflow research would result in improved approaches to workflow sharing, retrieval, and adaptation. Klaus-Dieter Althoff analyzed the relationships between CBR and expert systems using different perspectives, including problem solving, learning, competence development, and knowledge types. The presentations and posters covered a wide range of CBR topics of interest to both practitioners and researchers, including foundational issues covering case representation, similarity, retrieval, and adaptation; conversational CBR recommender systems; multi-agent collaborative systems;

data mining; time series analysis; Web applications; knowledge management; legal reasoning; healthcare systems and planning and scheduling systems.

Many people participated in making ICCBR 2012 a great success. In particular Amélie Cordier and Marie Lefevre of the University Claude Bernard, France, who served as Conference Co-chairs, with Belén Díaz-Agudo, Complutense de Madrid, Spain, and Ian Watson, University of Auckland, New Zealand, as Program Co-chairs. We would especially like to thank Luc Lamontagne, University of Laval, Canada, and Juan A. Recio García, Universidad Complutense de Madrid, for serving as Workshop Coordinators, and Mehmet Göker of SalesForce and William Cheetham of General Electric for chairing the Introduction to CBR Development Tools meeting on the first day. We wish to thank David Aha and Thomas Roth-Berghofer for organizing the valuable Doctoral Consortium and Michel Manago for chairing the Computer Cooking Contest.

We thank the Program Committee and all our additional reviewers for their thoughtful and timely participation in the paper selection process. We acknowledge the time and effort put in by the members of the Local Organizing Committee at the Lyon 1 University including: Faty Berkai, Pierre-Antoine Champin, Béatrice Fuchs, Brigitte Guyader, Marie Lefevre, Alain Mille, Sylvie Oudot, and Raafat Zarka, plus all our student volunteers.

We are very grateful for the generous support of the ICCBR 2012 sponsors: the AIJ for student bursaries in support of the doctoral consortium, and Lyon 1 University, the Computer Science Department of Lyon 1 University, and CNRS for supporting the conference in general. Finally, we appreciate the support provided by EasyChair in the management of this conference and we thank Springer for its continuing support in publishing the proceedings of ICCBR.

Finally, we would like to dedicate the proceedings of this conference to the memory of Alan Mathison Turing (1912–1954), the father of computer science and of artificial intelligence.

September 2012

Belén Díaz-Agudo
Ian Watson

Organization

Program Chairs

Belén Díaz-Agudo	Universidad Complutense de Madrid, Spain
Ian Watson	University of Auckland, New Zealand

Conference Chairs

Amélie Cordier	University Claude Bernard Lyon 1, France
Marie Lefevre	University Claude Bernard Lyon 1, France

Industry Day Chairs

William Cheetham	General Electric, USA
Mehmet Göker	SalesForce, USA

Workshop Coordinators

Luc Lamontagne	Laval University, Canada
Juan Recio-Garcia	Universidad Complutense de Madrid, Spain

Doctoral Consortium Chairs

David Aha	Naval Research Laboratory, USA
Thomas Roth-Berghofer	University of West London, UK

Cooking Competition Chair

Michel Manago	Kiolis, France

Program Committee

Agnar Aamodt	Norwegian University of Science and Technology, Norway
David Aha	Naval Research Laboratory, USA
Josep Lluis Arcos	IIIA - CSIC, Artificial Intelligence Research Institute, Spain
Kevin Ashley	University of Pittsburgh, USA
Ralph Bergmann	University of Trier, Germany
Isabelle Bichindaritz	University of Washington, Tacoma, USA
Derek Bridge	University College Cork, Ireland

Table of Contents

Case-Based Reasoning and Expert Systems

Klaus-Dieter Althoff[1,2]

[1] Competence Center Case-Based Reasoning
German Research Center for Artificial Intelligence (DFKI) GmbH
Trippstadter Strasse 122, 67663 Kaiserslautern, Germany
klaus-dieter.althoff@dfki.de
[2] University of Hildesheim
Institute of Computer Science - Intelligent Information Systems Lab

Abstract. Case-based reasoning (CBR) and expert systems have a long tradition in artificial intelligence: CBR since the late 1970s and expert systems since the late 1960s. While expert systems are based on expertise and expert reasoning capabilities for a specific area of responsibility, CBR is an approach for problem solving and learning of humans and computers. Starting from different research activities, CBR and expert systems have become overlapping research fields. In this talk the relationships between CBR and expert systems are analyzed from different perspectives like problem solving, learning, competence development, and knowledge types. As human case-based reasoners are quite successful in integrating problem-solving and learning, combining different problem solving strategies, utilizing different kinds of knowledge, and becoming experts for specific areas of responsibility, computer based expert systems do not have the reputation to be successful at these tasks. Based on this, the potential of CBR succeeding as future expert systems is discussed.

Keywords: case-based reasoning, expert systems.

Reproducibility and Efficiency of Scientific Data Analysis: Scientific Workflows and Case-Based Reasoning

Yolanda Gil

Information Sciences Institute
University of Southern California
Marina del Rey, CA, USA
gil@isi.edu

Abstract. Scientists carry out complex scientific data analyses by managing and executing many related computational steps. Typically, scientists find a type of analysis relevant to their data, implement it step by step to try it out, and run many variants as they explore different datasets or method configurations. These processes are often done manually and are prone to error, slowing the pace of discoveries. Scientific workflows have emerged as a formalism to represent how the individual steps work and how they relate to the overall process. Workflows can be published, discovered, and reused to make data analysis processes more efficient through automation and assistance. In this talk, I will argue that integrating case-based reasoning techniques with workflows research would result in improved approaches to workflow sharing, retrieval, and adaptation. I will describe our initial work on semantic workflow matching using labeled graphs and knowledge intensive similarity measures. Furthermore, I will argue that if scientists followed a case-based approach more closely, scientific results would be more easily inspectable and reproducible. Through scientific workflows and case-based reasoning, scientific data analysis could be made more efficient and more rigorous.

Keywords: case-based reasoning, scientific data analysis.

A Computer Aided System for Post-operative Pain Treatment Combining Knowledge Discovery and Case-Based Reasoning

Mobyen Uddin Ahmed and Peter Funk

School of Innovation, Design and Engineering,
Mälardalen University, P.O. Box 883 SE-721 23, Västerås, Sweden
{mobyen.uddinahmed,peter.funk}@mdh.se

Abstract. The quality improvement for individual postoperative-pain treatment is an important issue. This paper presents a computer aided system for physicians in their decision making tasks in post-operative pain treatment. Here, the system combines a Case-Based Reasoning (CBR) approach with knowledge discovery. Knowledge discovery is applied in terms of clustering in order to identify the unusual cases. We applied a two layered case structure for case solutions i.e. the treatment is in the first layer and outcome after treatment (i.e. recovery of the patient) is in the second layer. Moreover, a 2nd order retrieval approach is applied in the CBR retrieval step in order to retrieve the most similar cases. The system enables physicians to make more informed decisions since they are able to explore similar both regular and rare cases of post-operative patients. The two layered case structure is moving the focus from diagnosis to outcome i.e. the recovery of the patient, something a physician is especially interested in, including the risk of complications and side effects.

1 Introduction

Approximately 40 million patients are undergoing minor to major surgical operations every year in Europe[1][33]. At least half of these patients from children to elderly suffered with moderate or severe post-operative pain. The degree of post-operative pain differs for various patients, operation site and the type of operation. For example, an operation on the thorax and upper abdomen are more painful than the lower abdomen [1]. Pain is considered to be an obstacle of recovery and also requires significant health care resources to manage. A number of factors such as clinical, local and patient-related questions are asked to the patient by the healthcare provider before and after the operation to decide on a proper treatment in pain relief. In practice, the clinician makes a pain treatment plan based on guidelines, standard protocols and evidence-based approaches before the operation, and monitors the recovery and pain levels afterwards and adjusts treatments when necessary. However, approximate 30% of the population does not conform within recommended procedures due to individual factors and unusual or exceptional clinical situations.

[1] http://pain-out.med.uni-jena.de/index.php/about-pain-out/research

Physicians might have experience with unusual or exceptional situations but may not remember them at the point of care due to large amounts of regular situations. Thus, the quality improvement of individual postoperative-pain management has become an important issue. A computer-aided intelligent Decision Support System (DSS) that generates alarms by presenting both regular and rare situations is seen by many physicians as a beneficial tool in their decision making tasks in post-operative pain treatment (informal discussion with physicians in the Pain-out project).

Physicians have experience which may have been collected during many years both from successful solutions as well as from very costly mistakes. This opens up new possibilities for experience reuse. DSS in experience reuse bears more similarities with human reasoning and is often easily accepted by physicians in the medical domain [2], [3], [4] and [5]. The aim of this paper is to develop a computer-based system that can act as an "intelligent" Clinical Decision Support System (CDSS) for experience reuse in post-operative pain management. The problems are addressed as below:

1. Traditional computer-based methods do not adequately enable experience reuse, dynamic performance improvement and efficient decision support in post-operative pain treatment.
2. Today, there is no system designed in post-operative pain management that identifies rare (i.e. exceptional and unusual) cases in order to enable experience reuse for the clinicians.

The research focuses on the development of a computer-based system that is able to address current problems faced by the healthcare industry by using CBR, knowledge discovery (i.e. clustering) and identification of rare cases. Therefore, the system combines case-based reasoning with knowledge discovery in terms of clustering and outlier detection in order to enable experience reuse by identifying rare cases. CBR is often suitable for applications with a weak domain theory, i.e. when the domain is difficult to formalize and empirical [4], which is a common scenario in medical domains. The advantages of CBR in the medical domain have been identified in several research works i.e. in [2], [3], [4], [9], [7], and [8]. CBR is inspired by the way humans reason e.g. solve a new problem by applying previous experiences adapted to the current situation. Thus, it will help to transfer experiences in the form of past cases similar to a current situation and assists in personalized treatment management. Aamodt and Plaza has introduced a CBR life cycle with four RE-s: Retrieve, Reuse, Revise and Retain [6] [30]. Fuzzy logic is applied in similarity matching for case retrieval to reduce sharpness and handle uncertainty/vagueness to estimate similarity between cases. The authors in [10] and [11] introduce Fuzzy logic with CBR in similarity measurement. A 2nd order clustering based approach has been applied where it combines Fuzzy C-Means (FCM) with Hierarchical clustering in order to identify rare cases. Using this approach the 18% of the cases are identified as rare (i.e., exceptional and unusual) and inputted into the case-library [12].

2 Related Work

A research effort has been carried out through a literature study, where recent advancement of CDSS in pain treatment has been investigated. A review paper on

DSS for chronic pain management in primary care is presented in [13], where 8 DSS have been studied by the authors. According to the paper, all 8 DSS were designed to assist clinicians in pain management. Most of them have applied artificial intelligent techniques such as CBR, rule-base reasoning (RBR), fuzzy logic and so on. The authors in [14] present a DSS in pain management for cancer patients. In their proposed system, the daily pain assessment has been conducted through a Numerical Visual Analog Scale (NVAS) and the DSS assists clinicians in counseling correct deviations from pain therapy guidelines. A recent DSS in the domain of palliative care addressed by Houeland and Aamodt [15] is an advice-giving system that assists clinicians to improve treatment of pain in lung cancer patients. The proposed system incorporates rule-based and model–based methods into the CBR approach. Elvidge in [16] also describes a DSS to help healthcare providers to improve pain and symptom management for advanced cancer patients. His web-based DSS incorporated CBR with evidence-based standards for end-of-life cancer care. To our knowledge CDSS in postoperative pain management is limited so far.

CBR receives much attention and an increasing number of research projects and numerous applications are addressed [7], [18], [19], [17] and [32] in the medical domain both in diagnosis and treatment in terms of experience reuse. One main reason for this interest and increasing number of research projects and applications is the relationship between reasoning with cases, evidence-based medicine and evidence-based practice (EBP). CBR and hybrid CBR is an emerging research area and not yet widely used but there are already medical applications using CBR such as in cancer diagnosis [20], and [21], oncology [22], Alzheimer patients [2], diabetes [23], obstructive sleep apnea [24], stress diagnosis and biofeedback treatment [25], [5], [26], and [27], breast cancer [28] and hemodialysis [29].

3 Overview of the System

A case-based system mainly depends on cases, their types and how they should be represented. The case comprises unique features to describe a problem. Cases can be presented in different ways; in the post-operative pain treatment application domain the case structure contains three parts: 1) problem, 2) solution and 3) outcome, which are slightly different from any other CBR system. Here, it has been constructed in traditional ways with symptoms and solution (treatment) and added outcome (recovery success) in an innovative way [31]. The cases could be defined differently on the basis of their use, manner and nature. Different DSSs might have different requirements on which type of cases are to be handled. For this domain, different types of cases namely regular cases and rare cases which is a more user-friendly format for physicians have been proposed. In this article cases that do not follow any standard protocol are regarded as outliers and classified in the “rare case” group by using cluster-based approach discussed in section 4. The “rare case” group contains exceptionally good cases and/or unusually bad cases. Further, these cases are also authorised and tagged by the case owner. A short description of the different types of cases used in the system is presented in [31].

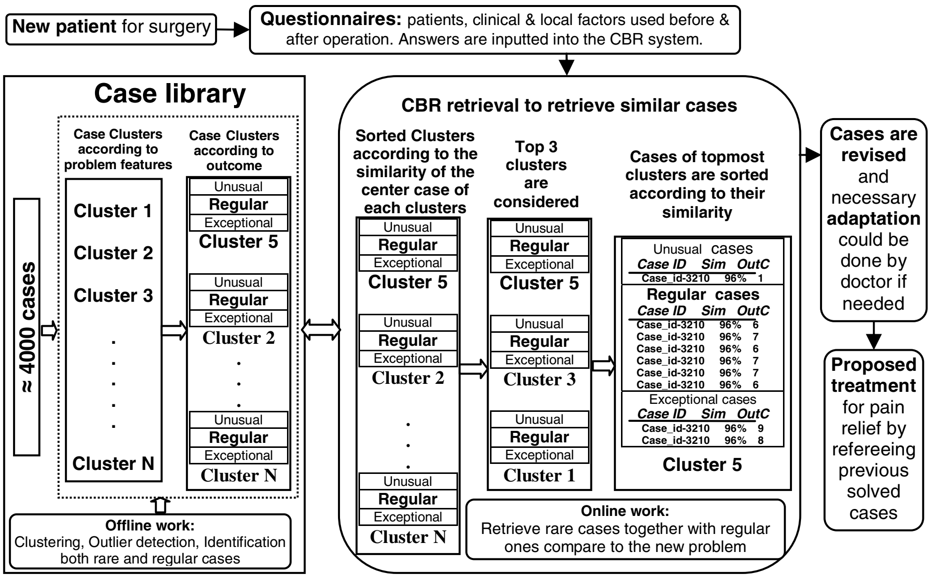

New patient for surgery
Questionnaires: patients, clinical & local factors used before & after operation. Answers are inputted into the CBR system.
Case library
≈ 4000 cases
Case Clusters according to problem features
Case Clusters according to outcome
Cluster 1
Cluster 2
Cluster 3
Cluster N
Unusual
Regular
Exceptional
Cluster 5
Cluster 2
Cluster N
Offline work: Clustering, Outlier detection, Identification both rare and regular cases
CBR retrieval to retrieve similar cases
Sorted Clusters according to the similarity of the center case of each clusters
Top 3 clusters are considered
Cases of topmost clusters are sorted according to their similarity
Cluster 3
Cluster 1
Unusual cases
Regular cases
Exceptional cases
Case ID Sim OutC
Case_id-3210 96%
Online work: Retrieve rare cases together with regular ones compare to the new problem
Cases are revised and necessary adaptation could be done by doctor if needed
Proposed treatment for pain relief by refereeing previous solved cases

Fig. 1. Schematic diagram of the system's work flow

A clinician confronted with an unknown case may be able to save the patient's life by remembering just one similar case and what was done to cure the patient. In this post-operative pain treatment context, it could be assumed that there may be some patients with rare conditions who have been treated successfully outside any standard protocol. The detailed information of such patients can be created as a case and these cases are then collected and stored to build a case library. A case is represented with a feature vector in the conditional part where the patient's contextual information, habit, site and nature of the surgery are considered. The solution part of the case consists of prescribed medication, and the outcome part of the case could contain pain assessment in a Numerical Visual Analog Scale (NVAS). The outcome explains the intention of using the solution and the potential of the solution to solve the problem. The retrieval step of the CBR cycle [6] can retrieve best matching cases (rare and regular cases) with solutions and outcomes from the case library. A schematic diagram of the proposed system is illustrated in Fig. 1. The three main components of the proposed decision support system are:

1. A library for knowledge and methods: a case library which contain ≈ 4000 records of previous cases where all the cases are clustered off-line. The clustering has been conducted in 2 steps; firstly cases are clustered in order to find rare cases, and secondly the rare-cluster cases are classified as exceptionally good and unusually bad according to case outcome.
2. An inference engine: CBR retrieval: an inference engine that employs the reasoning methodology to retrieves similar cases. In this system, the retrieval mechanism is conducted in two steps, firstly a new problem case will be compared with the center case of each cluster and the clusters are sorted accordingly, and then the top clusters are considered for final comparison.

Here, all the cases from the top most clusters are compared with the new one and sorted according to their similarity values.

3. An efficient and interactive user interface: provide a friendly ‘look and feel’ user interface for the doctors to revise, reuse and retain a case.

The design detail of the system starts during the admission of a surgical patient in a hospital, where the patient’s general, contextual, surgical, and habitual information are collected. This information is used in CBR systems to formulate a new problem case and to represent cases in the case library. In this CBR system, a new problem case is matched with selected cases from the case library and depends on the most similar clusters. The following tasks could be performed to implement the case-based system:

- Given a new problem case (new patient), the system will measure its similarity by comparing the feature vectors of a new problem case and center cases from each cluster, then sort the clusters accordingly. Finally, the system will retrieve similar cases from similar clusters in order to select the proper medication for a new patient.
- The physician could see regular and rare cases (i.e., exceptional and unusual cases) close to the new case to make a decision. The physician might reuse the same treatment or could revise the cases to adapt previous solutions to the new patient and advise on patient-specific medication for treatment.
- The physician could save the new problem case with the solution i.e. prescribed treatment and outcome as new knowledge for future use.

4 Identification of Rare Cases by Means of Clustering

In fact, finding rare cases is a difficult problem and especially true in the context of case mining, where one often wants to uncover subtle patterns that may be hidden in massive amounts of data. This is an important issue in knowledge discovery and there is a need to develop mechanisms to identify these rare cases. One common and easy way is to cluster all the cases in a case library and then identify the rare cases. In this system, the rare case identification is done offline in two steps: 1) all cases in the case library are clustered using a Fuzzy C-Means Clustering (FCM) algorithm and 2) the cases from each cluster are again grouped by applying the Hierarchical algorithm. Fig. 2. illustrates the steps that are taken into consideration while searching the rare cases.

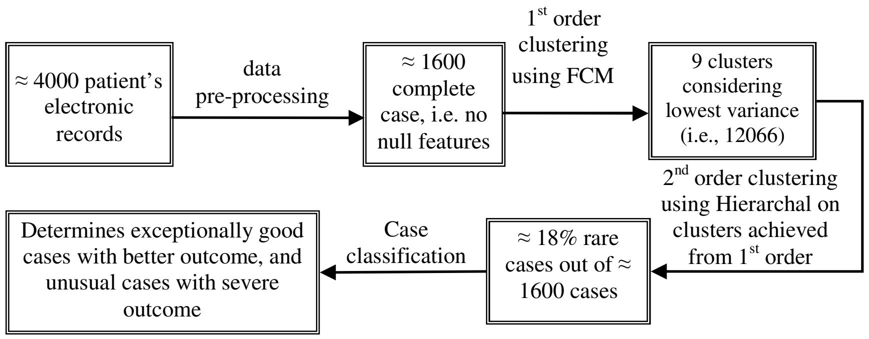

≈ 4000 patient’s electronic records
data pre-processing
≈ 1600 complete case, i.e. no null features
1st order clustering using FCM
9 clusters considering lowest variance (i.e., 12066)
2nd order clustering using Hierarchal on clusters achieved from 1st order
≈ 18% rare cases out of ≈ 1600 cases
Case classification
Determines exceptionally good cases with better outcome, and unusual cases with severe outcome

Fig. 2. Steps of the approach in order to identify rare cases, adapted from [12]

A data pre-processing step including a feature abstraction step is performed on the ≈ 4000 records of post-operative pain patients. In total ≈1600 cases with 17 features (1 for case ID, 15 for *problems* and 1 for *outcomes*) are obtained after the data preprocessing step and is discussed in [31]. However, only 15 features in the *problems* part of the cases were used in clustering. All the clustering algorithms and the user interface to identify rare cases are developed in MATLAB and applied MATLAB build-in clustering functions.

The 1st order clustering has been done using a FCM algorithm on the *problems* part of the cases. The detailed FCM method is presented in [12]. FCM is applied as a multi-variant clustering where 15 features are involved excluding ID. The main goal of this stage is partitioning, i.e. all the cases should be divided into several small groups with a similar frequency. Here, the percentage of average variance (i.e. the algorithm runs 10 times for each k) is used as a function to determine the number of clusters. The lowest percentage of average variance is achieved when the number of clusters is 9. The detailed information is presented in [12].

In the 2nd order, these 9 clusters are used and the Hierarchical clustering algorithm is applied in each cluster. The details of the algorithm are presented in [12]. In Hierarchical clustering, the distance between pairs of objects is calculated using Euclidean distance as a default parameter of the MATLAB function 'pdist'. The linkage function applies 'single' (i.e. shortage distance) as a default parameter which determines the objects in the data set that should be grouped into clusters. Finally, a cluster function is applied to group the sample data set into clusters by specifying the cluster's number. Here, the cluster's number is determined by observing the percentage of the case frequency. That is, the algorithm continues its iteration by increasing the number of clusters as long as at least two clusters obtained more than 10% of whole cases. Then, the clusters with small sizes (i.e. less than 10 %) are selected as the rare case cluster and thus the approach has achieved ≈ 18% as rare cases.

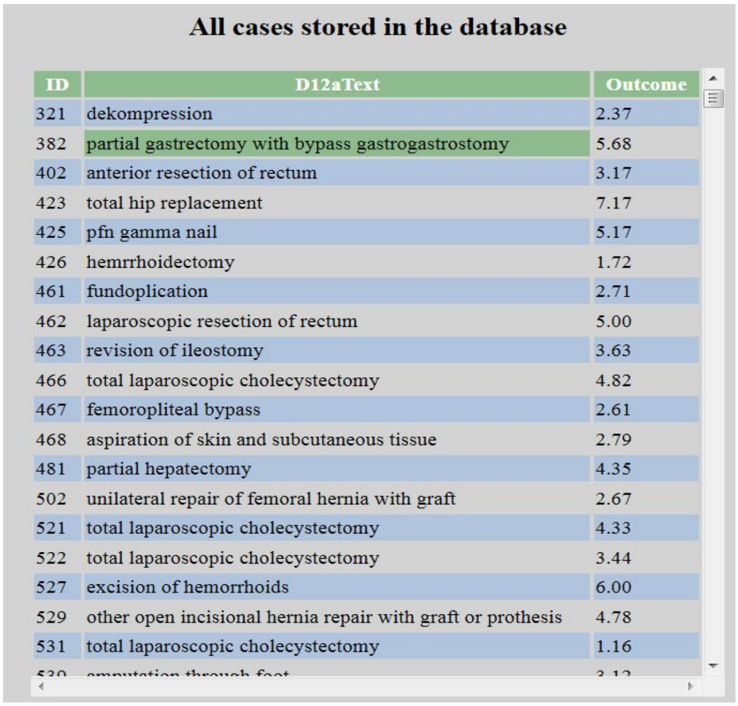
The last step in Fig. 2, determines the ≈ 18% cases whether they are *exceptionally* good (0-3.9) or *unusual* bad (6-10) according to the pain outcome (the threshold for good/bad may be changed). The attribute outcome is the average value of the pain measurement for each case. A clinician may be most interested in the extreme cases first (0/10) when looking for similar cases among the rare cases. Thus, the approach obtained 158 cases as unusually bad and 104 cases as exceptionally good. Only 14 cases with the outcome value between 4 and 5 exist among the set of rare cases.

5 Case-Based Decision Support System

As the cases in this project are formulated in three parts, the 'problem description' part contains around 278 attributes, and 'treatment' as a solution consists of 685 attributes, while 'outcome' as a recovery measure has 19 attributes. However, to formulate a case, feature abstraction has been done only considering the problem description and outcome information, which has been further mapped with the solution. So, out of 278 attributes only 15 features are extructed in the *problem* part and only 1 from 19 attributes were extracted from the *outcome* part. Deatailed information about feature abastruction is presented in [31]. Note that, the solution part of the cases remains unchanged since this data contains important medicine information which might modify during abstraction.

Retrieval is essential in medical applications since missed similar cases may lead to less informed decisions. Two of the factors which the reliability and accuracy of the system depends on are: 1) which cases are stored in the case library i.e. quality of the cases; 2) the retrieval of the relevant cases and their ranking. In this CBR system, similarity measurements are used to assess the degrees of matching by applying the standard Nearest Neighbour method as a global similarity algorithm. However, the similarity calculation has been done in two steps:

- Step 1: a new problem case is compared with the centre case of each cluster (note, all the clustering and outlier detection has been made on offline) and all the clusters are then sorted according to their similarity value.
- Step 2: the new problem case is compared again with all the cases belonging to each cluster. Hence, the system only considers the top nearest clusters define by the user threshold to the new problem case.

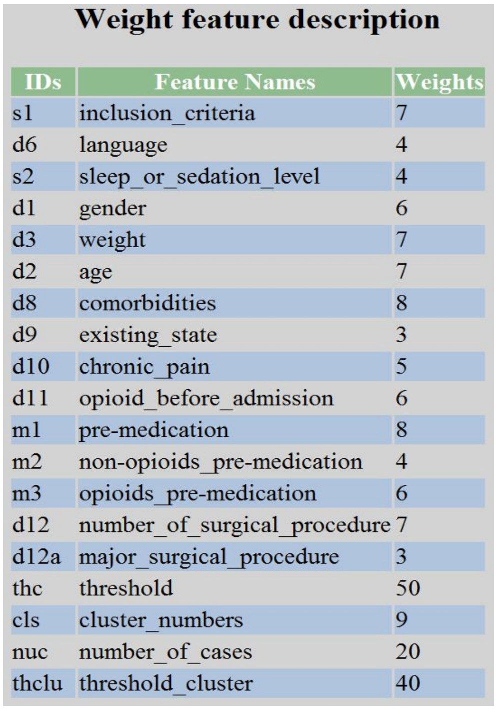


All cases stored in the database

ID	D12aText	Outcome
321	dekompression	2.37
382	partial gastrectomy with bypass gastrogastrostomy	5.68
402	anterior resection of rectum	3.17
423	total hip replacement	7.17
425	pfn gamma nail	5.17
426	hemrrhoidectomy	1.72
461	fundoplication	2.71
462	laparoscopic resection of rectum	5.00
463	revision of ileostomy	3.63
466	total laparoscopic cholecystectomy	4.82
467	femoropliteal bypass	2.61
468	aspiration of skin and subcutaneous tissue	2.79
481	partial hepatectomy	4.35
502	unilateral repair of femoral hernia with graft	2.67
521	total laparoscopic cholecystectomy	4.33
522	total laparoscopic cholecystectomy	3.44
527	excision of hemorrhoids	6.00
529	other open incisional hernia repair with graft or prothesis	4.78
531	total laparoscopic cholecystectomy	1.16
530	amputation through foot	3.12

Fig. 3. A screen shot of the DSS presents all stored cases with pain outcomes

The CBR retrieval approach was developed using the PHP programming language and the case library was built using a MySQL database. In Fig. 3, a screen shot of the DSS presents all stored cases from the case library along with their average outcome. Two cases are compared using different local similarity algorithms including *modified distance function*; *similarity matrix* and *fuzzy similarity matching* [5] [27]. The local weight defined by the case author or owner for each stored case, is assumed to be a quantity reflecting the importance of the corresponding feature individually. The reason to use individual case weighting is to combine several clinicians and experts knowledge into the system. The average weight of each feature and user defined threshold are presented in Fig. 4.

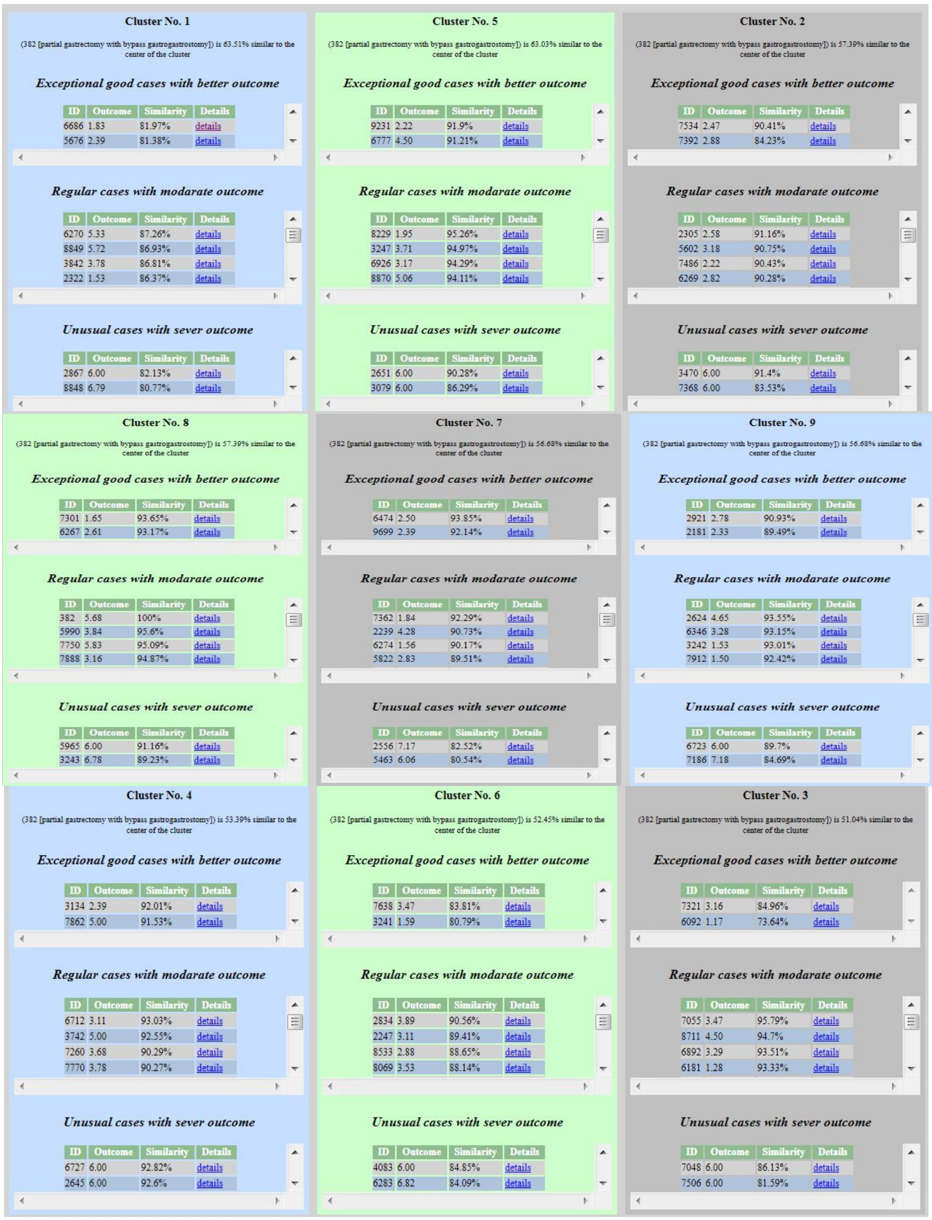


Weight feature description

IDs	Feature Names	Weights
s1	inclusion_criteria	7
d6	language	4
s2	sleep_or_sedation_level	4
d1	gender	6
d3	weight	7
d2	age	7
d8	comorbidities	8
d9	existing_state	3
d10	chronic_pain	5
d11	opioid_before_admission	6
m1	pre-medication	8
m2	non-opioids_pre-medication	4
m3	opioids_pre-medication	6
d12	number_of_surgical_procedure	7
d12a	major_surgical_procedure	3
thc	threshold	50
cls	cluster_numbers	9
nuc	number_of_cases	20
thclu	threshold_cluster	40

Fig. 4. DSS presenting features and average weight of the stored cases in the case library

The verification of the proposed approach performed in a prototypical system involves a close collaboration with doctors, nurses and clinics. The performance of the matching function in CBR is verified by whether the system can retrieve best similar cases with suitable solutions to provide proper treatment for a new case. The most similar cases are compared to a new problem case are retrieved and presented in 9 different clusters as shown in Fig. 5.



Cluster No. 1

(382 [partial gastrectomy with bypass gastrogastrostomy]) is 63.51% similar to the center of the cluster

Exceptional good cases with better outcome

ID	Outcome	Similarity	Details
6686	1.83	81.97%	details
5676	2.39	81.38%	details

Regular cases with modarate outcome

ID	Outcome	Similarity	Details
6270	5.33	87.26%	details
8849	5.72	86.93%	details
3842	3.78	86.81%	details
2322	1.53	86.37%	details

Unusual cases with sever outcome

ID	Outcome	Similarity	Details
2867	6.00	82.13%	details
8848	6.79	80.77%	details

Cluster No. 5

(382 [partial gastrectomy with bypass gastrogastrostomy]) is 63.03% similar to the center of the cluster

Exceptional good cases with better outcome

ID	Outcome	Similarity	Details
9231	2.22	91.9%	details
6777	4.50	91.21%	details

Regular cases with modarate outcome

ID	Outcome	Similarity	Details
8229	1.95	95.26%	details
3247	3.71	94.97%	details
6926	3.17	94.29%	details
8870	5.06	94.11%	details

Unusual cases with sever outcome

ID	Outcome	Similarity	Details
2651	6.00	90.28%	details
3079	6.00	86.29%	details

Cluster No. 2

(382 [partial gastrectomy with bypass gastrogastrostomy]) is 57.39% similar to the center of the cluster

Exceptional good cases with better outcome

ID	Outcome	Similarity	Details
7534	2.47	90.41%	details
7392	2.88	84.23%	details

Regular cases with modarate outcome

ID	Outcome	Similarity	Details
2305	2.58	91.16%	details
5602	3.18	90.75%	details
7486	2.22	90.43%	details
6269	2.82	90.28%	details

Unusual cases with sever outcome

ID	Outcome	Similarity	Details
3470	6.00	91.4%	details
7368	6.00	83.53%	details

Cluster No. 8

(382 [partial gastrectomy with bypass gastrogastrostomy]) is 57.39% similar to the center of the cluster

Exceptional good cases with better outcome

ID	Outcome	Similarity	Details
7301	1.65	93.65%	details
6267	2.61	93.17%	details

Regular cases with modarate outcome

ID	Outcome	Similarity	Details
382	5.68	100%	details
5990	3.84	95.6%	details
7750	5.83	95.09%	details
7888	3.16	94.87%	details

Unusual cases with sever outcome

ID	Outcome	Similarity	Details
5965	6.00	91.16%	details
3243	6.78	89.23%	details

Cluster No. 7

(382 [partial gastrectomy with bypass gastrogastrostomy]) is 56.68% similar to the center of the cluster

Exceptional good cases with better outcome

ID	Outcome	Similarity	Details
6474	2.50	93.85%	details
9699	2.39	92.14%	details

Regular cases with modarate outcome

ID	Outcome	Similarity	Details
7362	1.84	92.29%	details
2239	4.28	90.73%	details
6274	1.56	90.17%	details
5822	2.83	89.51%	details

Unusual cases with sever outcome

ID	Outcome	Similarity	Details
2556	7.17	82.52%	details
5463	6.06	80.54%	details

Cluster No. 9

(382 [partial gastrectomy with bypass gastrogastrostomy]) is 56.68% similar to the center of the cluster

Exceptional good cases with better outcome

ID	Outcome	Similarity	Details
2921	2.78	90.93%	details
2181	2.33	89.49%	details

Regular cases with modarate outcome

ID	Outcome	Similarity	Details
2624	4.65	93.55%	details
6346	3.28	93.15%	details
3242	1.53	93.01%	details
7912	1.50	92.42%	details

Unusual cases with sever outcome

ID	Outcome	Similarity	Details
6723	6.00	89.7%	details
7186	7.18	84.69%	details

Cluster No. 4

(382 [partial gastrectomy with bypass gastrogastrostomy]) is 53.39% similar to the center of the cluster

Exceptional good cases with better outcome

ID	Outcome	Similarity	Details
3134	2.39	92.01%	details
7862	5.00	91.53%	details

Regular cases with modarate outcome

ID	Outcome	Similarity	Details
6712	3.11	93.03%	details
3742	5.00	92.55%	details
7260	3.68	90.29%	details
7770	3.78	90.27%	details

Unusual cases with sever outcome

ID	Outcome	Similarity	Details
6727	6.00	92.82%	details
2645	6.00	92.6%	details

Cluster No. 6

(382 [partial gastrectomy with bypass gastrogastrostomy]) is 52.45% similar to the center of the cluster

Exceptional good cases with better outcome

ID	Outcome	Similarity	Details
7638	3.47	83.81%	details
3241	1.59	80.79%	details

Regular cases with modarate outcome

ID	Outcome	Similarity	Details
2834	3.89	90.56%	details
2247	3.11	89.41%	details
8533	2.88	88.65%	details
8069	3.53	88.14%	details

Unusual cases with sever outcome

ID	Outcome	Similarity	Details
4083	6.00	84.85%	details
6283	6.82	84.09%	details

Cluster No. 3

(382 [partial gastrectomy with bypass gastrogastrostomy]) is 51.04% similar to the center of the cluster

Exceptional good cases with better outcome

ID	Outcome	Similarity	Details
7321	3.16	84.96%	details
6092	1.17	73.64%	details

Regular cases with modarate outcome

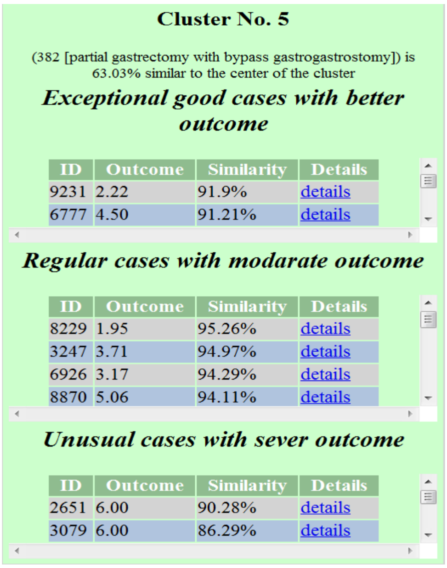
ID	Outcome	Similarity	Details
7055	3.47	95.79%	details
8711	4.50	94.7%	details
6892	3.29	93.51%	details
6181	1.28	93.33%	details

Unusual cases with sever outcome

ID	Outcome	Similarity	Details
7048	6.00	86.13%	details
7506	6.00	81.59%	details

Fig. 5. The system presenting the most similar cases both with rare cases (exceptional and/or unusual) and regular outcomes in different clusters

Here, cases are presented in different clusters based on the problem description and outcome. A case outcome has a value between 0 and 10, where "0" defines no pain at all and "10" defines severe pain. Note that, in the domain of post-operative pain treatment, similar solutions have different outcomes.



Cluster No. 5

(382 [partial gastrectomy with bypass gastrogastrostomy]) is 63.03% similar to the center of the cluster

Exceptional good cases with better outcome

ID	Outcome	Similarity	Details
9231	2.22	91.9%	details
6777	4.50	91.21%	details

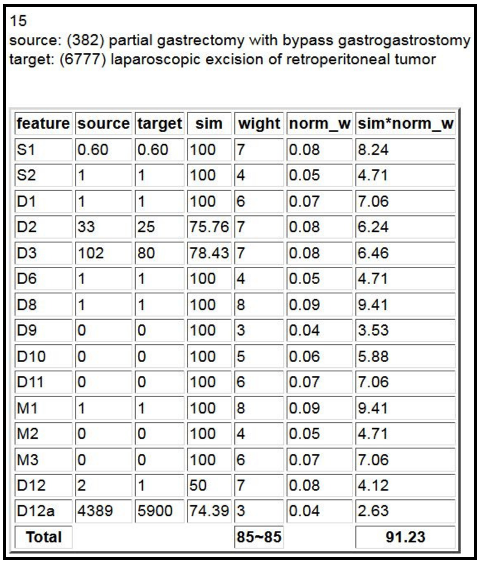
Regular cases with modarate outcome

ID	Outcome	Similarity	Details
8229	1.95	95.26%	details
3247	3.71	94.97%	details
6926	3.17	94.29%	details
8870	5.06	94.11%	details

Unusual cases with sever outcome

ID	Outcome	Similarity	Details
2651	6.00	90.28%	details
3079	6.00	86.29%	details

Fig. 6. A screen shot of Cluster 5, where most similar cases are presented both in rare and regular



15
source: (382) partial gastrectomy with bypass gastrogastrostomy
target: (6777) laparoscopic excision of retroperitoneal tumor

feature	source	target	sim	wight	norm_w	sim*norm_w
S1	0.60	0.60	100	7	0.08	8.24
S2	1	1	100	4	0.05	4.71
D1	1	1	100	6	0.07	7.06
D2	33	25	75.76	7	0.08	6.24
D3	102	80	78.43	7	0.08	6.46
D6	1	1	100	4	0.05	4.71
D8	1	1	100	8	0.09	9.41
D9	0	0	100	3	0.04	3.53
D10	0	0	100	5	0.06	5.88
D11	0	0	100	6	0.07	7.06
M1	1	1	100	8	0.09	9.41
M2	0	0	100	4	0.05	4.71
M3	0	0	100	6	0.07	7.06
D12	2	1	50	7	0.08	4.12
D12a	4389	5900	74.39	3	0.04	2.63
Total				**85~85**		**91.23**

Fig. 7. A screen shot of the DSS presents the overall similarity calculation between two cases

For example, cases in cluster 5 (in Fig. 6) are presented as rare (exceptional and unusual) and regular cases based on the case outcome. A doctor can look at the details and revise the case and thereafter make the decision. Thus, the doctor can reuse the solution from the previous case or may adjust a new solution for the pain treatment. This new problem case, the associated solution and the outcome in 24 hours could be stored in the case base for further use.

In Fig. 7, the overall calculations for similarity measurement of two cases are presented, where cases are 'source' i.e. stored case (id=382) from the case library and 'target' i.e. a new problem case that needs to be solved. Here, the weight of each feature is determined by calculating the average of all the weights of that feature and the weights are then normalised. The calculation shows the similarity between these two cases are 91.23, i.e. 91% the cases are similar.

6 System Verification

As we discussed earlier (in Fig. 2), the overall system works both offline and online, so the system has been tested considering 2 criterions: 1) clustering and identification of rare cases and 2) case-based retrieval. The cases are clustered using a 2nd order clustering-based approach, where around four thousand normally distributed data points are taken as input. However, only $\approx$ 1600 cases are used by considering complete cases i.e. cases with null features are excluded. Here, the execution time required for FCM 0.62 milliseconds and the Hierarchical 0.69 milliseconds. The number of clusters in the 1st order approach is determined by considering the average variance of 10 times iteration of the FCM algorithm. In 2nd order approach, the cluster's number is determined by observing the percentage of the case frequency. That is, the algorithm continues its iteration by increasing the number of clusters as long as at least two clusters obtained more than 10% of whole cases. Then, the clusters with small sizes (i.e. less than 10 %) are selected as the rare case cluster. Among the rare cases ≈ 57.25% of the cases are found as unusually bad and ≈ 37.68% of the cases are found as exceptionally good. However, about 5% of the rare cases contain the average outcomes value between 4 and 5. The details evaluations are presented in [12].

The CDSS using *case-based retrieval* is verified by implementation as a prototype where all the implemented methods are compared according to their outcome, that is, in terms of the technical point of view, they functioned properly. According to Watson [30], the trial has been conducted through the following 4 tests: 1) *retrieval accuracy* 2) *retrieval consistency* 3) *case duplication* and 4) *global test*. For the test *retrieval accuracy*, one case is taken from the case library as a query case and then the system retrieves the most similar cases. Among the retrieved cases, the query case is also retrieved as the top similar case with the similarity value 1.0. That is, the similarity value of two same cases is computed as 100% match. To test the *retrieval consistency*, the same query is used to perform more than one similar search and if it has been found that the same stored cases have been retrieved with the same similarity then the implemented retrieval function has consistency. It is also observed that no cases are identical during retrieval except the query case when it matches itself, thus it checks for *case duplication*. Regarding the *global test*, we have considered whether the most retrieved cases are useful to a take decision for treatment of pain relief. For the cases containing several solutions where each solution has several outcomes, we cannot consider "the percentages of correctly classification" [26]. However, considering the top 5 most similar cases (k=5), a clinician could find the appropriate case solution for treatment. Moreover, the proposed system also presents rare cases together with regular ones, this will generate an awareness alert, so that the care provider can focus resources on those who need it most and physicians are able to make a detailed analysis with references and advice from leading experts and statistics.

7 Conclusion

This paper presents a computer-based system in order to assist clinicians in their decision making task for post-operative pain treatment. The proposed approach combines a CBR approach with a 2nd order clustering approach in order to retrieve and present rare cases together with regular ones. Here, the retrieval step is conducted in two steps; 1st similarity values have been calculated between a new problem case with the center case of each cluster identified by applying Fuzzy C-Means and 2nd similarity values have been calculated between all the cases in the top clusters and the new problem case. Finally, the most similar cases in the clusters are structured in regular, exceptionally good and unusually bad cases are presented together. It was interestingly observed that the similarity values of the regular cases are always higher than the exceptionally good and unusually bad cases. This indicates that it is difficult to identify these cases using a traditional CBR approach alone and was possible to obtain by combining knowledge discovery with a CBR approach. This kind of discoveries and results became possible thanks to physicians' willingness to actively participate in the project, and their strong belief that such CDSS system would improve the quality of the pain treatment has been most encouraging. However, the value needs to be confirmed and validated by larger trails and measurements are ongoing.

Acknowledgments. This work has been supported by the PAIN OUT project, grant agreement number 223590, funded by the EC Seventh Framework Programme theme FP7-HEALTH-2007-3.

References

[1] Charlton, E.: The Management of Postoperative Pain. World Federation of Societies of Anaesthesiologists (7), article 2 (1997), `http://www.nda.ox.ac.uk/wfsa/html/u07/u07_003.html` (accessed March 2011)

[2] Corchado, J.M., Bajo, J., Abraham, A.: GERAmI: Improving the delivery of health care. Journal of IEEE Intelligent Systems, Special Issue on Ambient Intelligence 3(2), 19–25 (2008)

[3] Montani, S., Portinale, L., Leonardi, G., Bellazzi, R.: Case-based retrieval to support the treatment of end stage renal failure patients. Artificial Intelligence in Medicine 37, 31–42 (2006)

[4] O'Sullivan, D., Bertolotto, M., Wilson, D., McLoghlin, E.: Fusing Mobile Case-Based Decision Support with Intelligent Patient Knowledge Management. In: Workshop on CBR in the Health Sciences, pp. 151–160 (2006)

[5] Begum, S., Ahmed, M.U., Funk, P., Xiong, N., Von Schéele, B.: A Case-Based Decision Support System for Individual Stress Diagnosis Using Fuzzy Similarity Matching. In: Computational Intelligence (CI), vol. 25(3), pp. 180–195. Blackwell (2009)

[6] Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications, 39–59 (1994)

[7] Gierl, L., Bull, M., Schmidt, R.: 11. CBR in Medicine. In: Lenz, M., Bartsch-Spörl, B., Burkhard, H.-D., Wess, S. (eds.) Case-Based Reasoning Technology. LNCS (LNAI), vol. 1400, pp. 273–298. Springer, Heidelberg (1998)

[8] Holt, A., Bichindaritz, I., Schmidt, R., Perner, P.: Medical applications in case-based reasoning. The Knowledge Engineering Review 20(3), 289–292 (2005)

[9] Perner, P.: Introduction to Case-Based Reasoning for Signals and Images. In: Perner, P. (ed.) Case-Based Reasoning on Signals and Images. Springer (2007)

[10] Bonissone, P., Cheetham, W.: Fuzzy Case-Based Reasoning for Residential Property Valuation. In: Handbook on Fuzzy Computing (G 15.1). Oxford University Press (1998)

[11] Wang, W.J.: New similarity measures on fuzzy sets and on elements. Fuzzy Sets and Systems, 305–309 (1997)

[12] Ahmed, M.U., Funk, P.: Mining Rare Cases in Post-Operative Pain by Means of Outlier Detection. In: IEEE International Symposium on Signal Processing and Information Technology, pp. 035–041 (2011)

[13] Smith, Y.M., DePue, D.J., Rini, C.: Computerized Decision-Support Systems for Chronic Pain Management in Primary Care. American Academy of Pain Medicine 8(S3) (2007)

[14] Bertsche, T., Askoxylakis, V., Habl, G., Laidig, F., Kaltschmidt, J., Schmitt, S.P.W., Ghaderi, H., Bois, A.Z., Milker-Zabel, S., Debus, J., Bardenheuer, H.J., Haefeli, E.W.: Multidisciplinary pain management based on a computerized clinical decision support system in cancer pain patients. In: PAIN. Elsevier Inc. (2009)

[15] Houeland, G.T., Aamodt, A.: Towards an Introspective Architecture for Meta-level Reasoning in Clinical Decision Support System. In: The 7th Workshop on Case-Based Reasoning in the Health Sciences, Seattle, Washington, USA (July 2009)

[16] Elvidge, K.: Improving Pain & Symptom Management for Advanced Cancer Patients with a Clinical Decision Support System, eHealth Beyond the Horizon – Get IT There. In: Andersen, S.K., et al. (eds.) The Proceedings of the International Congress of the European Federation for Medical Informatics. IOS Press (2008)

[17] Begum, S., Ahmed, M.U., Funk, P., Xiong, N., Folke, M.: Case-Based Reasoning Systems in the Health Sciences: A Survey on Recent Trends and Developments. Accepted in IEEE Transactions on Systems, Man, and Cybernetics–Part C: Applications and Reviews (2010)

[18] Bichindaritz, I., Marling, C.: Case-based reasoning in the health sciences: What's next? Artificial Intelligence in Medicine 36(2), 127–135 (2006)

[19] Montani, S.: Exploring new roles for case-based reasoning in heterogeneous AI systems for medical decision support. In: Applied Intelligence, pp. 275–285 (2007)

[20] De Paz, F.J., Rodriguez, S., Bajo, J., Corchao, M.J.: Case-based reasoning as a decision support system for cancer diagnosis: A case study. International Journal of Hybrid Intelligent Systems (IJHIS) (2008)

[21] Glez-Peña, D., Díaz, F., Hernández, J.M., Corchado, J.M., Fdez-Riverola, F.: geneCBR: multiple-microarray analysis and Internet gathering information with application for aiding diagnosis in cancer research. Oxford Bioinformatics (2008) ISSN: 1367-4803

[22] Cordier, A., Fuchs, B., Lieber, J., Mille, A.: On-Line Domain Knowledge Management for Case-Based Medical Recommendation. In: Workshop on CBR in the Health Sciences, ICCBR 2007, pp. 285–294 (2007)

[23] Marling, C., Shubrook, J., Schwartz, F.: Case-Based Decision Support for Patients with Type 1 Diabetes on Insulin Pump Therapy. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 325–339. Springer, Heidelberg (2008)

[24] Kwiatkowska, M., Atkins, M.S.: Case Representation and Retrieval in the Diagnosis and Treatment of Obstructive Sleep Apnea: A Semio-fuzzy Approach. In: Proceedings of 7th European Conference on Case-Based Reasoning, pp.25-35 (2004)

[25] Nilsson, M., Funk, P., Olsson, E., Schéele, B.V., Xiong, N.: Clinical decision-support for diagnosing stress-related disorders by applying psychophysiological medical knowledge to an instance-based learning system. Journal of Artificial Intelligence in Medicine 36(2), 156–176 (2005)

[26] Ahmed, M.U., Begum, S., Funk, P., Xiong, N., Schéele, B.V.: A Multi-Module Case Based Biofeedback System for Stress Treatment. Artificial Intelligence in Medicine 52(2) (2011)

[27] Ahmed, M.U., Begum, S., Funk, P., Xiong, N., Schéele, B.V.: Case-based Reasoning for Diagnosis of Stress using Enhanced Cosine and Fuzzy Similarity. Transactions on Case-Based Reasoning on Multimedia Data 1(1) (October 2008) ISSN: 1864-9734

[28] D'Aquin, M., Lieber, J., Napoli, A.: Adaptation knowledge acquisition: a case study for case-based decision support in oncology. Computational Intelligence 22(3-4), 161–176 (2006)

[29] Montani, S., Portinale, L., Leonardi, G., Bellazzi, R., Bellazzi, R.: Case-based retrieval to support the treatment of end stage renal failure patients. Artificial Intelligence in Medicine 37, 31–42 (2006)

[30] Watson, I.: Applying Case-Based Reasoning: Techniques for Enterprise systems (1997)

[31] Ahmed, M.U., Funk, P.: A Case-Based Retrieval System for Post-operative Pain Treatment. In: Perner, P., Rub, G. (eds.) The Proceeding of International Workshop on Case-Based Reasoning (CBR 2011), pp. 30–41. IBaI, Germany (2011)

[32] Ahmed, M.U., Begum, S., Funk, P.: A Hybrid Case-Based System in Stress Diagnosis and Treatment. Accepted in the IEEEEMBS International Conference on Biomedical and Health Informatics (BHI 2012) (2012)

[33] Weiser, T.G., Regenbogen, E.S., Thompson, K.D., Haynes, A.B., Lipsitz, S.R., Berry, W.R., Gawande, A.: An estimation of the global volume of surgery: a modelling strategy based on available data. The Lancet 372(9644), 1149 (2008)

Developing Case-Based Reasoning Applications Using myCBR 3

Kerstin Bach[1,2] and Klaus-Dieter Althoff[1,2]

[1] Competence Center Case-Based Reasoning
German Research Center for Artificial Intelligence (DFKI) GmbH
Trippstadter Strasse 122, 67663 Kaiserslautern, Germany
{kerstin.bach,klaus-dieter.althoff}@dfki.de
[2] University of Hildesheim
Institute of Computer Science - Intelligent Information Systems Lab
Marienburger Platz 22, 31141 Hildesheim, Germany
{bach,althoff}@iis.uni-hildesheim.de
http://www.dfki.de/web/competence/cccbr/

Abstract. This paper presents the Open Source tool myCBR which has been re-implemented as standalone application with a designated application programming interface that can be used as plug-in for various applications. We will introduce how knowledge according to Richter's knowledge containers can be modeled and how myCBR has been successfully applied within various applications. Especially we introduce novel features of myCBR that support knowledge engineers developing more comprehensive applications making use of existing knowledge such as Linked Data or User Generated Content. The applications presented in this paper present the high variety how CBR can be applied for web-based and mobile technologies as well as configuration, diagnostic or decision support tasks.

Keywords: Case-Based Reasoning Tools, Knowledge Container Development, Case-Based Reasoning Applications, Open Source Software.

1 Introduction

Researching for novel Case-Based Reasoning approaches requires a test bed where users can easily create CBR applications using on the one hand an intuitive user interface and on the other hand including CBR as modules in more complex software systems. The re-implementation of myCBR[1] in Java is offering both to its user. For the re-implementation we aimed at keeping the user friendly interface, while directly accessing the XML-based knowledge representation.

The underlying idea of Case-Based Reasoning is reusing previous cases for solving future problems [1]. Following this principle we should also capture our experience when creating CBR systems and provide it for future use. This is one

[1] Throughout this paper, when referring to myCBR we are always speaking of myCBR 3 (http://mycbr-project.net/preview)

reason for developing the Open Source tool myCBR, which is available under the GNU General Public License. At the German Research Center for Artificial Intelligence we are aiming at using myCBR within projects and therewith further develop features that are given back to the research community.

In this paper we will on the one hand show how knowledge required to develop a CBR system can be built using the user interface, how applications have successfully been developed based on the Software Development Kit (SDK) and also how the existing myCBR implementation has been extended. The myCBR functionalities are explained along the knowledge containers introduced by Richter [22]. They differentiate between compiled (vocabulary, rules and similarity measures) and interpreted (cases) knowledge. For each knowledge container we describe the features that are supported by the core myCBR tool and possible extensions that might be useful for certain applications.

myCBR as a tool for creating CBR systems should be able to cover a high variety of tasks such as decision support, diagnosis, planning, etc. Especially when building knowledge-based applications two aspects have to be considered: during the definition phase the knowledge bases have to be created and discussed with experts. Therefore a user interface is crucial for making this knowledge transparent. On the other hand for running the CBR system we need an engine that makes use of previously created knowledge. myCBR offers both – the first one is referred as myCBR GUI and the latter one as myCBR SDK. The GUI makes use of the SDK and wraps a user interaction interface around it.

This paper is structured as follows: First, Section 2 will introduce how the four knowledge containers are addressed within myCBR and how novel features extend the tool for its applicability in a higher variation of application scenarios. The second part, Section 3, will showcase, based on seven myCBR applications, how the tool can serve in an industrial and scientific context. In the following related work section we will have a closer look at other, freely available, CBR tools such as FreeCBR, jCOLIBRI or eXiT*CBR. The last section gives a short summary and an outlook on further activities in this area.

2 Knowledge Container Development

Figure 1 shows the general architecture of myCBR. The left hand side describes the core components including the data import, the knowledge containers, the explanation component as well as the retrieval module. While the model component holds the vocabulary in various types of attribute descriptions, the similarity measures are described independently and connected to the model itself. The case base provides the case organization as well as case addition and deletion. The transformation component is only available in myCBR's OSGi release, but offers adaptation and completion capabilities as they are described in Section 2.2.

Additionally we developed extensions for making myCBR applicable to a broader range of applications. These extensions currently focus on Knowledge Engineering and Knowledge Acquisition tasks. The *myCBR GUI* is the user interface for creating myCBR applications by filling the knowledge containers.

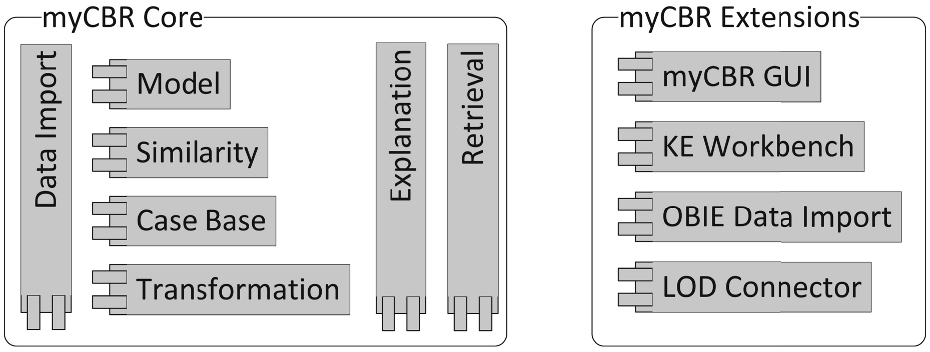


Fig. 1. myCBR Architecture

The other three extensions focus more on extracting relevant information from unstructured data and integrating them (semi-)automatically in the according knowledge containers.

2.1 Vocabulary

The vocabulary can be either built from scratch or by making use of existing data. The CBR applications we are targeting at with myCBR are structural CBR applications with a flat or object-oriented case representation (i.e. as described in [4]). Depending on the application and available data, Linked Data seems a good starting point building a proper, well-covering vocabulary. The myCBR SDK (myCBR Core in Figure 1) has an extension, called *LOD Connector*, which can access Linked Open Data sources such as DBPedia[2] or Freebase[3] for building taxonomies. When accessing linked data for building a vocabulary we target at concepts and their relations for eventually representing cases. While the concepts will serve as attribute values, the relations will fill the similarity measure container (see Section 2.3). For further use, the provenance of each value is stored in the concept explanations [24]. When building a taxonomy, the similarity between values is computed as follows: We start with a base similarity of $sim_0 = 0$ (by default), which is assigned to the root node. For each hierarchy level i, the children, grandchildren, etc., we compute the similarity to the root node as follows:

$$sim_i = 1 - \frac{1}{2^i} \tag{1}$$

The resulting taxonomy will then be organized comparable to the following skeleton, which contains for each parent node the similarity for all children in common [6]:

[2] http://dbpedia.org
[3] http://www.freebase.com/

```
+ root [$sim_0 = 0$]
  + Children_A [$sim_1 = 0.5$]
    – Grand_Children_AA
    – Grand_Children_AB
  + Children_B [$sim_1 = 0.5$]
    – Grand_Children_BA
    + Grand_Children_BB [$sim_2 = 0.75$]
      – Grand_Grand_Children_BBA
      – Grand_Grand_Children_BBB
  + Children_C [$sim_1 = 0.5$]
...
```

Another approach for creating the vocabulary is the fact that the myCBR user interface supports knowledge engineers defining the vocabulary. The myCBR graphical user interface (GUI) enables creating attributes with numeric and symbolic value ranges. From our experience, this user interface also supports the discussion with the domain experts, because they can see the knowledge model and provide insights [4].

2.2 Rules

So far, completion and adaptation rules are not deeply integrated in myCBR. However, we decided to create an integration of the industrial-strength rule engine Drools. In advance of this decision, we took a look at three different rule engines: JBoss Drools [7], jRete[4] and jRuleEngine[5]. We evaluated their suitability in the context of myCBR: on the one hand, according to the functionality they offer and, on the other hand, the integrability of the rule engine in the existing tool. Since myCBR is a freely available tool, only rule engines were examined, which are also freely available. Further, only Java-based rule engines are relevant allowing a straightforward integration into the Java based myCBR. All three use the Rete algorithm for optimization of the rule processing.

JBoss Drools Expert. Drools Expert is a rule engine of the JBoss community and part of the business logic platform. This platform consists of several modules and Drools Expert is the only relevant module in the myCBR rule engine context. The rules are defined in a proprietary rule language, the Drools Rule Language (DRL) and stored in a proprietary file format, which is based on DRL. Drools Expert supports many functions that are applicable for a Rule Engine in the area of CBR. Further development is necessary for handling symbolic attribute descriptions, which is required in a model based adaptation. Also other functionalities like set operations or similarity assignments have to be implemented additionally. Arbitrary objects can also be loaded into the Rule Engine and used by all rules as global objects. In addition, Drools Expert can be integrated into the myCBR GUI via OSGi and is offering most of the features

[4] http://sourceforge.net/projects/jrete/
[5] http://jruleengine.sourceforge.net/

required for completion and adaptation rules along with its open and modular framework.

None of the three rule engines can be recommended for the use without any limitations for myCBR. jRete and jRuleEngine by far do not offer the necessary functions and the customization would require an extremely high effort. Drools, however, offers a large part of the necessary functionality and can be connected with myCBR via the OSGi interface. Integrating Drools in myCBR meant having two types of rules to be implemented. On the one hand we have completion rules that run on the complete case base as a sub-process of loading the case base. They are built upon domain knowledge and enable the CBR system deducing attribute values from explicitly given features in a case or a query. Within this sub-process additional information is loaded, for example if you take cooking recipes as a case, completion rules can add the type of meal based on the ingredients and preparation methods [18]. Furthermore, each user query is also enriched with information using completion rules. The other type of rules, adaptation rules, are only applied to a subset of cases for performance reasons. Adaptation rules can become very complex and that is the reason why we only use them on the top 5/10/20 cases in order to adapt them to the user's preferences. Before Drools rules can be applied to a myCBR case base we first had to transform the myCBR SDK in an OSGi structure providing the required services.

Completion Rules. The process of applying Drools rules as completion rules is pictured in Figure 2. After the initial myCBR case base has been loaded the completion rules are loaded from a CSV file. Completion rules are simple if-then-rules that are applied to each case, where the if clause describes the condition and the then part the action. If an attribute value is already set it replaces the existing or adds another value.

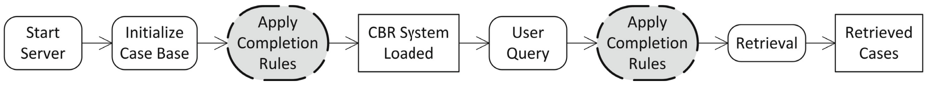


Fig. 2. Completion Rules for myCBR based on Drools

Adaptation Rules. Adaptation rules are more complex since the conditions as well as their according actions might cause time-consuming changes on a case, because depending on the underlying knowledge models, for instance similarity tables, taxonomies, calculations, adequate substitution candidates or value changes have to be retrieved along with the conditions that specify the circumstances when a rule can be applied. Afterwards the case itself gets adapted. The idea behind this way of adaptation is depicted in Figure 3.

One of the challenging aspects determining adequate adaptation candidates is making use of existing knowledge models. For instance when using taxonomies for similarity assessment we have sibling nodes and parent nodes. Depending on the knowledge engineering strategy, one has to define how substitution candidates are selected. Afterwards the rule also has to check whether the node is (a)

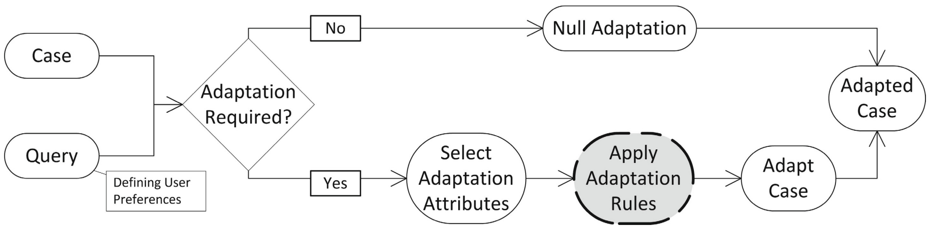


Fig. 3. Adaptation Rules for myCBR based on Drools

artificial and should not be used or (b) it fits the given user preferences. Artificial nodes are common in complex knowledge models. They represent connections between nodes that implicitly exist, but for which no instance is present (either caused by incompleteness or circumvented complexity/overfitting). Drools, does not support that feature in particular, however, it allows the access to knowledge models and the following steps describe how we currently compute substitution candidates (assuming that the values to be substituted are known):

1. Identify the taxonomy T in which the value n_{target} to be adapted is situated.
2. Identify the parent node n_{parent}.
3. Identify all children nodes $(n_{sibling_1}...n_{sibling_n})$.
4. Remove all artificial nodes of $n_{sibling_1}...n_{sibling_n}$ until only instantiable nodes exist.
5. Remove all n_x of $n_{sibling_1}...n_{sibling_n}$ that do not fit the user's preferences provided in the initial query.
6. Select the adaptation candidate with the highest similarity to n_{target}.

The identification of artificial nodes has been carried out by using myCBR's explanation capabilities, which allows the user to define so-called concept explanations for every entity in the model [24]. In this case we tagged artificial concepts in order to be accessible by the adaptation rule. Besides the adaptation based on knowledge within the knowledge containers, the rule engine based on Drools also allows more complex adaptation approaches that creates solution from case skeletons or even from scratch.

2.3 Similarity Measures

The similarity measures provided in myCBR do cover simple data types like Int, Float or Double for numeric attributes or String for textual attributes. myCBR also provides table similarities or taxonomies for symbolic value ranges. For numeric attributes myCBR additionally supplies also the similarity assessment based on the difference and quotient of values.

A very powerful similarity modeling approach provided by myCBR are similarity measures for set attributes. They allow more than one value per attribute and therefore the adequate handling is required. An example of the successful application is discussed in Section 3.1. A more detailed view on how similarity measures can be modeled in myCBR is given in [25].

2.4 Cases

Cases are given by the application domain and have to be transformed or indexed in order to fit the given case representation. Further, especially for e-Commerce or decision support scenarios on a common domain they can also be acquired from web sources. Especially when the source is User Generated Content some type of Information Extraction (IE) should be applied. Currently myCBR Core does not support any of these capabilities, however, we have created the *OBIE Data Import* extension (see Figure 1) for using IE tools when building the case base. Up to today we can use either SCOOBIE [2] or GATE's ANNIE component [9]. Both approaches use ontology-based IE, because this allows making use of Named Entities (NE) [12]. Cunningham [10] lists the requirements for ontology-based IE that we will also apply: Instead of plain gazetteers, we will use ontologies for which there is the main challenge populating the ontology. However, since we assume that the vocabulary has been defined as a knowledge model, for example a taxonomy, we initialize the IE process using that model. The result is a populated case base.

IE with SCOOBIE. SCOOBIE is an IE tool developed at DFKI that uses symbolic descriptions, especially RDFS[6] for describing, learning and further developing domain ontologies. SCOOBIE can be queried using SPARQL[7] and this is how the myCBR IE component makes use of SCOOBIE: we transform the myCBR knowledge model in an RDFS graph and during the process of loading cases we carry out IE that matches terms and populates cases.

IE with GATE. The GATE extension on the other hand makes also use of ontologies, but also applies more IE technologies. GATE's IE tool ANNIE [11] uses the so-called OntoRoot Gazetteer to create gazetteers based on ontologies, which are then applied to tokenized and NE-tagged data sources. IE based on ANNIE is more flexible because of its plug-in structure and depending on the provided data, additional NLP technologies can be applied. Furthermore, we tested the performance of both approaches measuring the computing time based on two knowledge models.

To compare these two IE approaches we carried out some performance tests on a MacBook with an Intel Core 2 Duo (2 GHz) processor and 4 GB RAM. Further we had two different knowledge models, model 1 contains 52 attribute values, model 2 has 1,482 attribute values, and we used three case bases that had to be indexed using model 1 and model 2. All models and case bases contain cooking recipes. Case base 1 (CB-1) contains only pasta recipes and model 1 has been created for that particular case base. CB-1 also contains reviewed and

[6] http://www.w3.org/TR/rdf-schema/
[7] http://www.w3.org/TR/rdf-sparql-query/

Table 1. IE benchmark for two knowledge models (seconds for ontology based IE and case population)

	Model 1		Model 2	
Case Base	SCOOBIE	ANNIE	SCOOBIE	ANNIE
CB-1 (108 cases)	203.87	25.83	323.72	21.64
CB-2 (100 cases)	69.33	10.54	87.23	24.77
CB-3 (1000 cases)	139.70	29.19	247.71	40.90

well-structured raw data. Model 2 contains more comprehensive cooking recipe knowledge, which targets at any type of recipe. Case bases CB-2 and CB-3 are arbitrary recipe collections of different sizes.

As Table 1 shows, the results integrating ANNIE outperform SCOOBIE. The quality of the extraction did not substantially differ. Another obvious effect is that the more matches we found, the more effort time the case population takes. This explains why the Model 1 – ANNIE population of CB-1 is much slower (more than twice the time) than the population of CB-2. Using the larger model, which contains almost 30 times more values, produces quite similar run times. Having this fact in mind and looking at the run times for populating CB-3 it shows that the number of cases populated influences the run time of case population most.

3 CBR Applications Based on myCBR

After introducing myCBR features as well as add-ons for an easier knowledge container population, this section will focus on applications we have recently built using myCBR. Each of the following applications uses core myCBR features, which are the vocabulary, multiple similarity measures, one – often more – case bases and a sequential search in a non organized case base. Depending on the applications' requirements we extend these existing features, for instance for populating cases for a case base, including Linked Data for building the vocabulary or creating cases, pre-processing data using IE, adding rule engines, or applying constraints. myCBR core is provided as a jar archive that provides CBR capabilities. The required processes and application logic has been developed independently and usually based on the examples given on the myCBR website.

Table 2 gives an overview of the applications we will discuss. The applications are grouped by their task and each row shows the amount of attributes (symbolic, numeric or textual), the used similarity measures, what type of rules is used, how many cases are included and which type of front end we provide to access the systems.

The core role within these applications is played by structured CBR and they show how manifold myCBR applications can turn out. About the half of the applications was created by Bachelor's and Master's students under supervision of a myCBR team member, the other part was implemented in research projects.

Table 2. Overview of myCBR-based applications

Name	Vocabulary	Similarity Measures	Rules	Cases	Front-End
Decision Support					
CookIIS	21 attributes, 1482 symbols	taxonomy, exact match, n-grams	178 completion rules, model-based adaptation	1489	mobile, web
EatSmart	13 attributes, 1323 symbols	taxonomy, exact match	none	272	web
myCamera	15 attributes, 38 symbols	taxonomy, polynomial, distance, exact match	none	624	web
FinancialDS	21 attributes, 140 symbols	taxonomy, table, cyclic, polynomial, distance, exact match, n-grams	4 simple completion rules	70	web
Configuration					
PC-Config	45 attributes, 116 symbols	taxonomy, polynomial, distance, exact match	hard coded constraints	488	web
Diagnosis					
Service Cases	31 attributes, 445 symbols	taxonomy, table, cyclic, polynomial, distance, exact match, n-grams	none	924	web
Decision Support & Information Composition					
docQuery	65 attributes, 1081 symbols	taxonomy, table, cyclic, polynomial, distance, exact match, n-grams	none	1061	proprietary

3.1 myCBR Applications

CookIIS is a CBR cooking recipe engine that showcases the strength of CBR in an easily understandable domain, which affects everyone once in a while: what to cook with the ingredients I love – while respecting allergies or dietary practices. CookIIS is developed since 2007 [13] and has been reimplemented recently using myCBR [5]. Further, it is the first time we used Drools as it is described in Section 2.2. This novel implementation also closes the loop of 4R [1] since it covers new features that collect feedback (Revise) and include new cases semi-automatically (Retain). From the user interaction point of view, CookIIS now made its way closer to the oven since it has an Android interface. The current version of the web interface as well as the mobile app can be found here: http://www.dfki.de/~bach/cookiis.html.

EatSmart is a CBR application that supports its users dealing with a *Metabolic Typing* conform nutrition. It picks up the idea of CookIIS, but is personalized and works with an individual training and eating plan. Usually only a menu for two or three weeks is given to *Metabolic Typing* customers. However, EatSmart increases the variety of recipes that fit in a certain nutrition plan and on the other hand points out if a user is looking for recipes or ingredients that are forbidden in the given nutrition plan. EatSmart is currently available in German only and can be found here: http://cbrdemo.kl.dfki.de/EatSmart. EatSmart showcases that CBR applications can make life easier by including intelligence into systems. The evaluation of this application with *Metabolic Typing* customers brought good results, because it was easier, especially for new users, to get familiar with ingredients and dishes that are allowed and those, which should be avoided.

myCamera differs from the before mentioned application since the user scenario – selecting an adequate digital camera – is less complex. However, myCamera showcases how different user interfaces can be implemented. On the one hand, in the simple search mode there are four questions to be answered and based on the given answers the underlying attributes are set. The answers are statements that describe how the camera will be used. On the other hand, the detailed search lets the user select eleven features that initialize a similarity-based retrieval. The novelty of myCamera are the cases that are retrieved from freebase. We query Linked Data and then populate the case base automatically. myCamera can be accessed via http://cbrdemo.kl.dfki.de/myCamera/.

PC-Config showcases how multiple case bases, in this particular application six: RAM, CPU, HDD, Graphic Card, Main board, Previous Configurations, can be integrated in order to configure valid PC systems. For the configuration either previous configurations can be recalled and revised to initialize a query or users can specify their preferences and the application starts configuring. The configuration is based on the selected main board, which sets the constraints for the subsequent components. Each case base is queried individually and the best matching cases are integrated. If the best match does not fit, PC-Config takes the second, third, etc. The application is in German only, it can be found here: http://cbrdemo.kl.dfki.de/PCKonfig/configpage.jsp. Besides addressing a configuration task, this application also makes use of Linked Data: the attribute values where obtained from Freebase and automatically populated in the myCBR knowledge model.

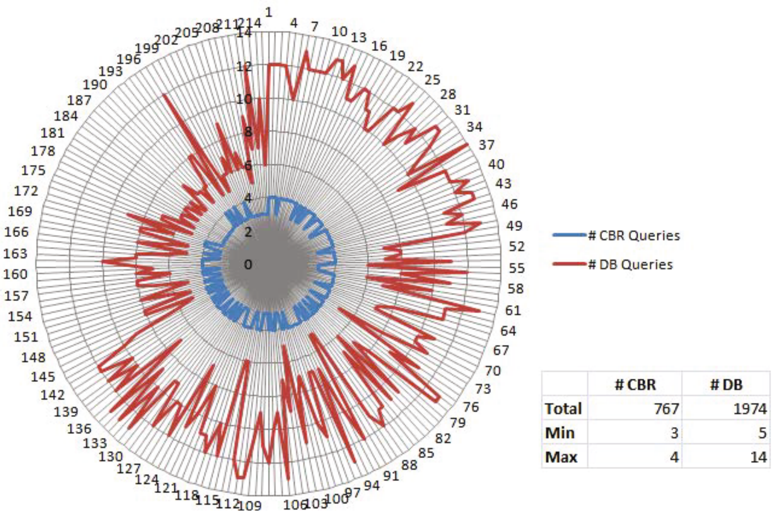
FinancialDS is applied in the domain of suggesting customers best fitting financing offers for certain goods. In this demonstrator application up-to-date financial plans and offerings have to be combined in order to create competitive offers for customers. This application partially makes use of Linked Data for populating the companies product names in the knowledge model and on the other hand basic, hard coded completion rules were used for Information

Extraction and attribute population while loading the case base. FinancialDS uses company internal guidelines for creating those cases that will then be presented to customers as product offers, so this approach uses general knowledge (guidelines) to produce an individual piece of information (product offer). Furthermore in the very beginning we only had a few guidelines available and only little expert knowledge on how to design similarity measures. The customization of guideline allowed us to create a proper case base and the high coverage of cases overcame the similarity measure shortcoming. According to Richter [22] this is one way how the knowledge container principle can adapt to various degrees of formalization.

Service Cases is a machine diagnosis application that was presented in detail in [4]. Summarized Service Cases make use of protocols between first and second level support of machines for providing faster, more efficient help in case a machine breaks down in the field. This application describes a classic scenario in which a well-organized collaborative knowledge base can be used in multiple areas – for maintaining warranty claims, improving the product by identifying weak points, and providing lessons learned by one participant to a broader audience. This type of diagnosis surely requires enough stakeholders, however, the scenario can be applied in many comparable domains. For instance, we have used a similar approach in a prototype for an intelligent office scenario where maintenance requests for ships were processed in a case-based way in order to save time from the invoice until an expert is sent out.

docQuery is a travel medicine application based on the SEASALT approach [21], which picks up the idea of collaborating multi-expert-systems [3]. The implementation does not have a web-based or mobile front end, which are currently developed. However, it integrates myCBR in JADE and currently only communicates via FIPA-ACL [8]. Within docQuery CBR is applied in two ways: First, as topic agents representing an expert in a particular area; second, as coordination agents that make use of previous query paths [17]. docQuery shows how myCBR can be applied in a distributed, agent-based architecture. The docQuery knowledge model is an object-oriented model, where each class has its own case base. When loading the topic agents, we initialize this set of case bases that is represented by the agent. The benefit of using CBR as underlying knowledge-based system is clearly the ranking of best matching results and the easy handling of multiple attributes. For example, when we are querying for diseases that occur in a certain region, we usually need information for one up to seven diseases. Querying a data base would mean sending seven requests – using myCBR's multiple value attribute means sending one request and receiving a list of diseases where the first seven hits have the same similarity and are relevant for further processing.

Figure 4 illustrates this effect for the queries required in docQuery to obtain information for each of the 214 countries (no 1-214 in the Figure) included in the system. In total CBR topic agents had been queried 767 times while the number of data base requests would have been 1,974. In this figure, the inner line represents the CBR queries that vary between 3 and 4 queries for each



	# CBR	# DB
Total	767	1974
Min	3	5
Max	4	14

Fig. 4. docQuery: CBR vs. DB queries for diseases

country – one query for each populated attribute. The outer line shows DB requests based on the same attributes. They require between 5 and 14 queries.

4 Related Work

Freely available CBR tools are for instance FreeCBR, jCOLIBRI or eXiT*CBR, which will be briefly discussed in this section. FreeCBR[8] is a rather simple CBR engine, which allows the realization of basic CBR features. However, it does not cover features like case revision or retention and more individualized knowledge models, or comprehensive global and local similarity measures, are not applicable either. Further, it still requires quite some effort to apply it to a high variety of tasks. jCOLIBRI started from a task oriented framework also covering distributed reasoning [19], recently jCOLIBRI Studio [20] for more comprehensive support of building CBR knowledge has been introduced. Up to today jCOLIBRI includes more machine learning and semantic web features while myCBR focused on the knowledge required in the knowledge containers. Furthermore, creating individualized case representations and especially flexible similarity measures is the strength of myCBR. eXiT*CBR has also its roots in machine learning applications and is specialized for medical diagnosis tasks [16]. It has recently been extended in order to cope with more than one case base. In comparison to myCBR, the ideas behind the methodology also differ, since we are focusing on the knowledge container model rather than the machine-learning-related tasks. The integration of Drools in an existing framework for executing rules on a given corpus has been introduced by Hanft et al. [14]. In this paper Drools has been

8 http://freecbr.sourceforge.net/

integrated in an existing OSGi environment. The approach presented here required a more comprehensive customization since myCBR was not embedded in OSGi and the requirements for the rules differed in terms of usable knowledge and modification of cases.

5 Summary

In this paper we presented myCBR 3 by introducing its core features as well as optional extensions for more complex applications. According to our Open Source strategy these extensions are also provided for myCBR users.

Our goal for the tool is to increase awareness of myCBR's capabilities and have more people use the tool. Further on, extensions such as the presented Information Extraction components are planned to be provided as add-ons. In these terms, we think of some kind of tool box or software product line [15] with the core myCBR framework that can be extended by, preferably Open Source, components on demand. The integration of Drools allows us to use completion and adaptation rules in applications, in addition we can also simulate constraints by the Drools rule mechanisms. However, currently myCBR does not support constraint-based problem solving.

The seven showcases presented have been implemented using myCBR GUI and SDK by different groups of users (students and myCBR developers). Currently we are collaborating with the University of West London, who are working on the explanation component of myCBR as introduced in SEASALT^{exp} [23].

The Information Extraction engine introduced in this paper enables an easier Knowledge Acquisition for CBR systems, because it supports a broad range of unstructured/weakly structured source data to well-formed heavily structured data and hence provides a scalable knowledge intensity. Because of the fact that many sources of experiential knowledge is User Generated Content, the presented approach makes this vast amount of unstructured data accessible for CBR applications more easily. The demonstrator applications introduced showcase the high variety of tasks myCBR can be applied to.

Acknowledgement. We would like to thank our students from the 2011 CBR lab course as well as Sara Marter, Julian Satzky, Fabian Schwahn and Irina Weizel for their great work and feedback on myCBR. Further, we thank the myCBR team members for their implementation efforts and valuable discussions.

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications 1(7) (March 1994)
2. Adrian, B., Hees, J., van Elst, L., Dengel, A.: iDocument: Using Ontologies for Extracting and Annotating Information from Unstructured Text. In: Mertsching, B., Hund, M., Aziz, Z. (eds.) KI 2009. LNCS (LNAI), vol. 5803, pp. 249–256. Springer, Heidelberg (2009)

3. Althoff, K.-D., Bach, K., Deutsch, J.-O., Hanft, A., Mänz, J., Müller, T., Newo, R., Reichle, M., Schaaf, M., Weis, K.-H.: Collaborative multi-expert-systems – realizing knowlegde-product-lines with case factories and distributed learning systems. In: Baumeister, J., Seipel, D. (eds.) Workshop Proceedings on the 3rd Workshop on Knowledge Engineering and Software Engineering (KESE 2007) (September 2007)
4. Bach, K., Althoff, K.-D., Newo, R., Stahl, A.: A Case-Based Reasoning Approach for Providing Machine Diagnosis from Service Reports. In: Ram, A., Wiratunga, N. (eds.) ICCBR 2011. LNCS, vol. 6880, pp. 363–377. Springer, Heidelberg (2011)
5. Bach, K., Sauer, C.S.: mycbr student demonstrators at the computer cooking contest. In: Agudo, B.D., Cordier, A. (eds.) ICCBR 2011 Workshop Proceedings: Computer Cooking Contest Workshop (2011)
6. Bach, K., Sauer, C.S., Althoff, K.-D.: Deriving case base vocabulary from web community data. In: Marling, C. (ed.) ICCBR 2010 Workshop. Proc.: Workshop on Reasoning From Experiences on the Web, pp. 111–120 (2010)
7. Bali, M.: Drools JBoss Rules 5.0 Developer's Guide. Packt Publishing (2009)
8. Bellifemine, F., Caire, G., Greenwood, D.: Developing multi-agent systems with JADE. Wiley Series in Agent Technology (2007)
9. Bontcheva, K., Tablan, V., Maynard, D., Cunningham, H.: Evolving gate to meet new challenges in language engineering. Natural Language Engineering 10(3-4), 349–373 (2004)
10. Cunningham, H.: Information extraction, automatic. Encyclopedia of Language and Linguistics, 665–677 (2005)
11. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In: Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL 2002) (2002)
12. Endres-Niggemeyer, B., Jauris-Heipke, S., Pinsky, M., Ulbrich, U.: Wissen gewinnen durch Wissen: Ontologiebasierte Informationsextraktion. Information Wissenschaft und Praxis 57(6/7), 301 (2006)
13. Hanft, A., Ihle, N., Bach, K., Newo, R.: CookIIS – competing in the first computer cooking contest. Künstliche Intelligenz 23(1), 30–33 (2009)
14. Hanft, A., Schäfer, O., Althoff, K.-D.: Integration of drools into an osgi-based bpm-platform for cbr. In: Díaz-Agudo, B., Cordier, A. (eds.) ICCBR 2011 Workshop Proceedings: Process-Oriented CBR (2011)
15. van der Linden, F., Schmid, K., Rommes, E.: Software Product Lines in Action - The Best Industrial Practice in Product Line Engineering. Springer, Heidelberg (2007)
16. López, B., Pous, C., Gay, P., Pla, A., Sanz, J., Brunet, J.: exit*cbr: A framework for case-based medical diagnosis development and experimentation. Artif. Intell. Med. 51(2), 81–91 (2011)
17. Marter, S.: Case based coordination agents - knowledge modularization and knowledge composition (Fallbasierte Koordinationsagenten – Wissensmodularisierung und Wissenskomposition für dezentrale, heterogene Fallbasen). Master's thesis, Institute of Computer Science, University of Hildesheim (2011)
18. Newo, R., Bach, K., Hanft, A., Althoff, K.-D.: On-demand recipe processing based on cbr. In: Marling, C. (ed.) ICCBR 2010 Workshop Proceedings: Computer Cooking Contest Workshop, Karlsruhe, Germany, pp. 209–218 (July 2010)
19. Recio-García, J.A., Díaz-Agudo, B., González-Calero, P.A.: A distributed cbr framework through semantic web services. In: Bramer, M., Coenen, F., Allen, T. (eds.) Research and Development in Intelligent Systems XXII (Proc. of AI 2005). pp. 88–101. Springer (December 2005)

20. Recio-García, J.A., Díaz-Agudo, B., González-Calero, P.A.: Template based design in colibri studio. In: Proceedings of the Process-oriented Case-Based Reasning Workshop at ICCBR 2011, pp. 101–110 (2011)
21. Reichle, M., Bach, K., Althoff, K.-D.: The SEASALT Architecture and Its Realization within the docQuery Project. In: Mertsching, B., Hund, M., Aziz, Z. (eds.) KI 2009. LNCS (LNAI), vol. 5803, pp. 556–563. Springer, Heidelberg (2009)
22. Richter, M.M.: 1. Introduction. In: Lenz, M., Bartsch-Spörl, B., Burkhard, H.-D., Wess, S. (eds.) Case-Based Reasoning Technology. LNCS (LNAI), vol. 1400, pp. 1–15. Springer, Heidelberg (1998)
23. Bergmann, R.: 2. Experience management. In: Bergmann, R. (ed.) Experience Management. LNCS (LNAI), vol. 2432, pp. 25–44. Springer, Heidelberg (2002)
24. Roth-Berghofer, T.R.: Explanations and Case-Based Reasoning: Foundational Issues. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 389–403. Springer, Heidelberg (2004)
25. Stahl, A., Roth-Berghofer, T.R.: Rapid Prototyping of CBR Applications with the Open Source Tool myCBR. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 615–629. Springer, Heidelberg (2008)

Diverse Plan Generation by Plan Adaptation and by First-Principles Planning: A Comparative Study

Alexandra Coman and Héctor Muñoz-Avila

Department of Computer Science and Engineering, 19 Memorial Drive West,
Lehigh University, Bethlehem, PA 18015
{alc308,hem4}@lehigh.edu

Abstract. Plan diversity has been explored in case-based planning, which relies on the availability of episodic knowledge, and in first-principles planning, which relies on the availability of a complete planning domain model. We present a first comparative study of these two approaches to obtaining diverse plans. We do so by developing a conceptual framework for plan diversity which subsumes both case-based and first-principles diverse plan generation, and using it to contrast two such systems, identifying their relative strengths and weaknesses. To corroborate our analysis, we perform a comparative experimental evaluation of these systems on a real-time strategy game domain.

Keywords: plan diversity, case-based planning, first-principles planning.

1 Introduction

There exists a large body of work comparing plan adaptation and first-principles planning. A number of studies have shown the benefits of planning informed by episodic knowledge over planning relying solely on complete planning models (e.g., collections of actions). Such studies have been conducted on total-order planning [21], partial-order planning [11], planning using planning graphs [8,9], and heuristic planning [20]. In addition, general frameworks have been developed to explain these benefits for both derivational analogy [1] and transformational analogy [12].

In this paper, we add a new step to this series of studies: a comparison of plan adaptation and first-principles planning with regard to solving the plan diversity problem. Plan diversity aims at obtaining multiple dissimilar solution plans for the same planning problem. Diverse plans can embody varied strategies and approaches to solving the problem, reflecting different priorities (such as caution versus willingness to explore in a military planning setting), thus catering to variation in circumstances, preferences and needs. Due to its practical relevance, plan diversity has been gaining attention over the years [15,19,4,5,6].

While previous work has shown separately how the plan diversity problem can be solved using case-based planning, which relies on the availability of episodic knowledge [4,6], and first-principles planning, which relies on the availability of a complete planning domain model [5,15,19], no work exists on comparing the two approaches. We address this gap by conducting a direct comparison between two systems for generating diverse plans, one using case-based planning, the other using first-principles planning techniques.

For the purposes of this study, we develop a conceptual framework for plan diversity which subsumes both case-based and first-principles diverse plan generation, and use it to contrast two such systems, identifying their relative strengths and weaknesses. We also perform a comparative experimental evaluation on a real-time strategy game planning domain, assessing the diversity of generated plans both by analyzing the plans themselves (as in [5,15,19]), and by analyzing the variation of the results obtained when running the plans in the game (as in [4,5,6]). While certain behavior will be ascribable to characteristics particular to the compared systems, we believe that our most significant conclusions can be extended to diverse plan generation systems using episodic knowledge/complete domain models in general.

2 A Framework for Diverse Plan Generation

In our previous work ([5] and [6]), we presented two systems for diverse planning: a first-principles planning system (based on the heuristic search planner FF [10]), and a case-based planning system. We will hereinafter be referring to the diverse first-principles planning system in [5] as Div*FF* and to the diverse case-based planning system in [6] as Div*CBP*. Div*CBP* conducts diverse case-based planning, using episodic knowledge (i.e. cases stored in a case base). Div*FF* conducts diverse heuristic planning, using complete models (i.e. state-transition models of planning domains). These models indicate, for every state, the applicable actions, and the states that result from applying these actions; the models are deterministic: for each *(state,action)* pair, they indicate only one resulting state.

In this paper, we propose a framework encompassing both Div*CBP* and Div*FF*. The crucial observation is that both systems are based on the same idea of repeatedly generating solutions based on a composite candidate solution plan evaluation criterion (Formula 1, where π is a plan and Π is a non-empty set of plans) balancing relative diversity with adequacy of the candidate solution plan.

$$EvalCrit(\pi, \Pi) = \alpha SolAdequacy(\pi) + (1-\alpha) RelDiv(\pi, \Pi) \quad (1)$$

Solution plan adequacy is a measure specific to each planning technique: estimated goal distance in heuristic search planning and case similarity in case-based planning. By modifying the α parameter in Formula 1, one can increase the emphasis on either solution adequacy or relative diversity.

Relative diversity (Formula 2) is the estimated diversity of the set of plans that will be obtained by completing the current partial candidate plan (if necessary) and adding it to the set of previously generated plans. In Formula 2 (a variant of the formula introduced by Smyth and McClave [18] in case-based reasoning for analysis tasks), π is a plan, Π is a non-empty plan set, and D: $\Pi \times \Pi \rightarrow [0,1]$ is a plan distance metric (a measure of the dissimilarity between two plans), which, as shown in [5] and [6], can be either quantitative (domain-independent) or qualitative (domain-specific).

$$RelDiv(\pi, \Pi) = \frac{\sum_{\pi' \in \Pi} D(\pi, \pi')}{|\Pi|} \quad (2)$$

In our framework, generation of a set of k diverse plans is conducted as indicated in the generalized pseudocode in Fig. 1. First, a plan is generated using the regular (non-diverse) variant of the planning system. This variant uses only a solution adequacy criterion (*SolAdequacy* in Formula 1) to assess candidate solutions. Then, (k-1) additional plans are generated using a modified, diversity-aware version of the planning system. This variant uses *EvalCrit* (Formula 1) to assess candidate solution plans. Relative diversity (Formula 2) is computed between the current candidate plan and the set of previously generated plans.

```
DiversePlanGeneration(PL,Prob,k)
//PL – planning system; Prob – problem; k – number of diverse plans to be
//generated; Π – set of plans; π – plan; $\pi_C$ – candidate solution plan;
//SolAdequacyPL – solution plan adequacy criterion specific to PL;
//RelDiv – Formula 2; generatePlan(Crit($\pi_C$), Prob) – generates a plan for
//problem Prob using criterion Crit($\pi_C$) to assess candidate solution plan $\pi_C$.

1. Π ← {}
2. π ← generatePlan(SolAdequacyPL($\pi_C$), Prob)
3. Add π to Π
4. Repeat
5.     π ← generatePlan(αSolAdequacyPL($\pi_C$)+(1-α)RelDiv($\pi_C$, Π), Prob)
6.     Add π to Π
7. Until |Π| = k plans have been generated
8. Return Π
```

Fig. 1. The general algorithm for generating diverse plan sets

In the next two subsections, we describe how the general pseudocode in Fig. 1 specifically encompasses both Div*CBP* and Div*FF*.

2.1 Diverse Plan Generation by Adapting Episodic Knowledge

Div*CBP* is subsumed by the algorithm in Fig. 1 in a case-based planning context. It introduces diversity at the retrieval stage of the CBR cycle. It assumes a transformational analogy adaptation method, in which the contents of a case are a problem (consisting of an initial and/or a final state) and a solution, consisting of a plan. The new problem is defined in terms of initial and/or final state. Line 2 (Fig. 1) corresponds to retrieving from the case base the case that is maximally similar to the new problem. Lines 4-7 correspond to the repeated retrieval, for *k-1* steps, of the case that maximizes a combined evaluation metric taking into account both the similarity to the new problem and the relative diversity to the set of solutions obtained so far. It follows that, for Div*CBP*, candidate solution adequacy is similarity to the new problem. Thus, Formula 1 can be written as follows:

$$EvalCritDivCBP(c,n,R) = \alpha Sim(c,n) + (1-\alpha)RelDiv(c.\pi,,R.\Pi) \qquad (3)$$

In Formula 3, c is the candidate case, n is the new problem, R is a set of previously retrieved cases, Sim is a case similarity measure used for traditional similarity-based retrieval, $c.\pi$ is the solution plan of case c, $R.\Pi$ is the set of all solution plans in R, and $RelDiv$ is defined as in Formula 2.

Retrieval of a diverse set of cases is followed by adaptation. Under the assumption that adaptation produces plans which are close to the source plan (as can be ensured using techniques such as adaptation-guided retrieval [17]), the resulting set of plans can be reasonably expected to be diverse, as is the case in [6]. In [6], we use Div*CBP* to generate plans for a real-time strategy game domain, with a simple domain-specific adaptation method. During adaptation, units of each kind (e.g. *soldier* units) in the new problem are matched to units of the same kind in the retrieved source plan. In the adapted plan, units in the new problem mimic the behavior of their corresponding units in the source plan.

The retrieval component of Div*CBP* is a variant of the Greedy retrieval algorithm proposed by Smyth and McClave [18]. The difference between the original Greedy method (used for analysis tasks) and our variant (used for planning, which is a synthesis task) arises from the fact that plan-diversity-aware retrieval needs to take the solution plan into account, in addition to the problem. During retrieval, the problem is considered for similarity purposes, while the solution is considered for diversity purposes. This criterion balancing between problem similarity and plan dissimilarity is captured in Formula 3.

2.2 Diverse Plan Generation Using First-Principles Planning

Div*FF* is a diverse plan generation version of the heuristic search planner FF [10]. Heuristic search planning is particularly well suited for modification according to the general algorithm in Fig. 1, as it obtains a solution plan by iterative refinement of partial solutions based on solution adequacy criteria. Planning is performed based on a given state-transition model.

With FF, search is conducted in the space of possible states, and candidate states are assessed using a goal distance heuristic, which is an estimate of the distance from the current candidate state to the goal state. The heuristic value of a candidate state is the length of the solution plan of a relaxation of the planning problem (we omit further discussion. For details, see [10]).

For the purposes of the relative diversity assessment in Formula 2, a candidate solution plan is obtained by merging the partial plan from the initial state to the current candidate state with the relaxed-problem plan generated by FF as mentioned before (hence, a candidate solution is only an estimate of the final solution).

In the context of Div*FF*, Line 2 in Fig. 1 corresponds to generating a plan using regular FF, while lines 4-7 correspond to generating *k-1* plans using the composite solution evaluation criterion described in Formula 4 below, which is a variant of Formula 1 (i.e., solution adequacy is computed using the FF heuristic).

$$EvalCritDivFF(\pi_c, \Pi) = \alpha h_{FF}(\pi_c) + (1-\alpha)RelDiv(\pi_c, \Pi) \tag{4}$$

In Formula 4, h_{FF} is the regular FF heuristic, π_c is a candidate plan, Π is the set of previously generated plans, and $RelDiv$ is defined as in Formula 2.

3 Comparative Analysis of Div*CBP* and Div*FF*

Div*FF* and Div*CBP* differ in terms of the knowledge they require as input, which also dictates a difference in the way planning is conducted. Div*FF* requires a complete state-transition model, and the search is conducted in the set of all states that can be constructed using this model. Div*CBP* uses the episodic knowledge provided by a case base, which can be incrementally updated; planning is conducted by following the case-based planning cycle.

Because Div*FF* uses complete domain information, it has the advantage of being able to generate any plan, which makes for higher potential plan diversity. A drawback is that the search space is exponential in the number of propositions in the do-main description. Therefore, actually finding diverse solutions may require considerable planning time. Planning effort may further increase when qualitative, rather than quantitative, distance metrics are used for diverse plan generation [5,6].

Quantitative distance metrics [5,6] are based on the number of plan elements (e.g., actions) which appear in strictly one of the compared plans. When quantitative metrics are used, it is possible for the resulting plans not to be meaningfully diverse. Indeed, in the real-time strategy game on which Div*FF* was previously tested, an informal inspection revealed that, frequently, plans in the generated set were encoding the same basic strategy, with the difference between them often consisting solely of a number of superfluous actions that had been added only in order to increase the diversity score. This was confirmed through experiments conducted in the game, in which the actual game scores resulting from running these plans had little variation, accounting for the fact that the plans were not meaningfully diverse. *Qualitative distance metrics* [5,6] can prevent this from happening, but may require Div*FF* to spend more time searching for a plan that meets the diversity criteria. Qualitative distance metrics incorporate domain-specific information in addition to the state transition model, and use it to compute plan dissimilarities. For example, in a real-time strategy game, an *archer* unit might be considered more similar to a *mage* than to a *soldier*, because the *mage* and *archer* are both capable of long-range attacks, while the *soldier* is not.

Another issue possibly affecting the ability of Div*FF* to reliably generate diverse plans is the fact that, during planning, it assesses relative diversity based on an ***estimate*** of the final plan (as explained in the previous section), as opposed to a completed plan.

The search space of Div*CBP* is limited to the contents of a case base. This may prove an impediment if the case base is not sufficiently diverse. More precisely, an ideal case base should include not only diverse plans, but diverse plans for diverse problems, so that whichever cases are preferred for problem-similarity purposes are also likely to include diverse plans. With such a case base, Div*CBP* should be able to reliably produce diverse plan sets. The primary reason for this is that diversity assessment is based on plans in the case base, which are complete.

(pa) Problem: 2 soldiers, 1 peasant, 1 mage //L1, L2: map locations; goal (as provided to Div*FF*): at least one unit has attacked
(πa) First generated plan: peasant patrols L1, *soldier1 attacks L2*

(b) Search Spaces (Examples)

Div*CBP*

CASE BASE *A* (3 cases)	**CASE BASE *B* (5 cases)**
(c1) Prob.: 2 sol., 2 pea., 1 mage **Plan:** pea1 patrols L1, *sol1 attacks L2*	**(c1) Prob.:** 2 sol., 2 pea., 1 mage **Plan:** pea1 patrols L1, *sol1 attacks L2*
(c2) Prob.: 2 sol., 2 pea., 1 mage **Plan:** *pea1 attacks L2*	**(c2) Prob.:** 2 sol., 2 pea., 1 mage **Plan:** *pea1 attacks L2*
(c3) Prob.: 2 sol., 3 pea., 3 mages **Plan:** pea1 patrols L1, *sol1 attacks L2*	**(c3) Prob.:** 2 sol., 3 pea., 3 mages **Plan:** pea1 patrols L1, *sol1 attacks L2*
▨	**(c4) Prob.:** 2 sol., 2 pea., 1 mage **Plan:** pea2 patrols L1, *mage attacks L2*
▨	**(c5) Prob.:** 6 sol., 10 pea., 6 mages **Plan:** pea3 patrols L1, *mage1 attacks L2*

Div*FF*

sol1 attacks L1	pea. attacks L1	sol2 attacks L1	mage attacks L1
sol1 patrols L1	pea. patrols L1	sol2 patrols L1	mage patrols L1
sol1 attacks L2	pea. attacks L2	sol2 attacks L2	mage attacks L2
sol1 patrols L2	pea. patrols L2	sol2 patrols L2	mage patrols L2

Example Candidate Plans for the Purposes of Computing Relative Diversity:
(πc) Div*CBP*: peasant patrols L1, *mage attacks L2*
(πd) Div*FF*: peasant patrols L1, [*mage attacks L2*]
$\searrow$
only an estimate, may not be part of final plan

Fig. 2. DivCBP and Div*FF* Comparison Example (problem and plans are simplified for brevity): (*p*a) problem (units represent entire armies, e.g. 1 *soldier* = 1 army of *soldiers*), (πa) the plan that has already been generated, (b) example search spaces (simplified), (πc) and (πd) example candidate solution plans

The simple adaptation criterion discussed before ensures that the adapted plans encode the same strategy as the source plan, and, consequently, that the set of plans generated by Div*CBP* is diverse. Even with smaller case bases (which may include diverse plans only for certain problems), it may be possible to increase generated plan diversity for Div*CBP* by lowering the α weight (Formula 3), so that the similarity factor is given less emphasis (or conversely, more emphasis is given to the diversity criterion).

Finally, while Div*FF* will often have a larger potential search space, it will not, in most cases, explore it exhaustively (at any step, it simply searches for the first state with a better heuristic evaluation than the previously selected one, stopping as soon as a solution is found). By contrast, Div*CBP* will always explore its entire search space, which, in the worst case, corresponds to the entire case base. If a maximally dissimilar solution exists in the case base, it may be retrieved (depending on whether it satisfies the similarity criterion as well). It also follows that the number of candidate plans considered by Div*CBP* will always be the same, irrespective of whether the distance metric is quantitative or qualitative.

In Fig. 2, we provide a comparative illustration of aspects of the Div*CBP* and Div*FF* planning processes. Consider problem pa for the real-time strategy game planning domain in [6]. It specifies the number of friendly armies of each type (*soldier* armies, *peasant* armies, etc.). Assume plan πa has already been generated, and we are now generating a second, dissimilar plan, using distance metric D = D_{Wargus} (see Formula 5 below), for computing relative diversity.

$$D_{Wargus}(\pi_1, \pi_2) = \begin{cases} 0, if\ attackUnitsType(\pi_1) = attackUnitsType(\pi_2) \\ d, 0 < d \leq 1, if\ attackUnitsType(\pi_1) \neq attackUnitsType(\pi_2) \end{cases} \tag{5}$$

In Formula 5, *attackUnitsType(π)* is the type of units in the attacking army of plan π, and *d* is the degree of difference between two types of units[1] [6].

The search space of Div*CBP* (Fig. 2(b)) is a case base. When using case base A, Div*CBP* retrieves case *c2*: even if *soldiers* are quite similar to *peasants*, this is the most dissimilar case available. When using case base B, case *c4* is retrieved, as its plan is maximally dissimilar to πa, based on D_{Wargus}, while its problem is very similar to pa. It should be noted that, if case *c4* were not available, case *c5* would be unlikely to be selected: even though it also uses *mages* to attack, its problem is not similar enough to problem pa. As a simplification[2], the search space of Div*FF* is represented in terms of the available actions in Fig. 2(b). The diversity of plans generated by Div*FF* will depend on the order in which actions are considered. If an attack action *a1* using a *peasant* army is considered before an attack action *a2* using a *mage* army, then the partial plan containing *a1* may be deemed sufficiently dissimilar to πa to be committed to, even if *a2* would be the choice creating maximal diversity. Candidate solutions for Div*CBP* (Fig. 2(πc)) are completed plans. Candidate solutions for

[1] In short, for the three types of units used in the example in Fig. 2, the degree of distance *d* specifies that *peasants* and *soldiers* are quite similar, and they are both very dissimilar to *mages*; and that any type of unit is maximally similar to itself.

[2] Strictly speaking, the search space contains all possible states. Unlike states, which are only visited once, actions can be selected multiple times.

Div*FF* (Fig. 2(πd)) are made up of two parts: the first has been committed to, while the second is only an estimate, computed as explained in the previous section. Even though candidate plan πd might be accurately assessed as being dissimilar to πa, the finalized plan might be different from the estimate πd, and very similar to πa.

Based on these considerations, in terms of generated plan set diversity, we expect Div*CBP* to generate less diverse sets of plans than Div*FF* when Div*CBP* can only access a small case base; and to outperform Div*FF* once the case base has been augmented sufficiently.

In terms of planning time, a significant bottleneck for Div*FF* is the preprocessing phase, which consists of parsing and *grounding*[3] the problem. Another time-consuming stage of the Div*FF* planning process is calculating the goal distance metric for each candidate state, which involves constructing a planning graph [2] and extracting a solution from it. Also, while Div*CBP* assesses candidate plans already provided in a case base, Div*FF* needs to actually generate these candidate plans. In the case of Div*CBP*, we expect planning time to increase as the size of the case base increases.

We believe that our results regarding plan diversity should, to some extent, be typical of diverse first-principles and case-based planners. However, results regarding planning time should probably not be used to draw generalizing conclusions, as other first-principles and case-based planning systems may have different efficiency-related strengths and weaknesses.

4 Experimental Evaluation

We use the real-time strategy game Wargus [4,5,6] for our evaluation. Div*FF* and Div*CBP* are used to generate battle plans for various game scenarios, varying in terms of number of armies of each type at our disposal.

4.1 Experimental Setup

In order not to introduce bias in favor of any of the compared algorithms, the systems themselves remain as described in [5] and [6]. Both systems are used with the qualitative distance metric D_{Wargus} (Formula 5), which was shown to produce significantly higher game diversity than the baseline quantitative distance metric in [6]. In each session, we tasked each planner to generate 4 diverse plans.

Test Problems. As test problems, we use the 5 problems in [6], indicating varying game configurations. For Div*CBP*, the game configurations specify the number of friendly armies of each type, as well as the number of units in each army. For Div*FF*, the configurations include the following additional information needed for the

[3] *Grounding* consists of instantiating operators into all possible actions, by associating them with all combinations of actual parameters out of the available objects in the problem description (e.g. operator <attack(*Soldier*, *Location*)>, given the available *Soldier* objects *sol1* and *sol2*, and the available *Locations loc1* and *loc2*, will be instantiated into 4 actions: <attack(*sol1*,*loc1*)>, <attack(*sol2*,*loc1*), <attack(*sol1*,*loc2*)>, and <attack(*sol2*,*loc2*)>.

complete model: (1) the coordinates of the units in the initial state, (2) the waypoints via which fighting units can move, and (3) a goal specifying that at least one army should have attacked.[4] This additional information does not affect plan diversity, as it is not used by the D_{Wargus} plan distance metric.

Game Configuration. The game configuration remains largely the same as in [6]. The only change involves replacing *archer* units with *gryphon rider* units, which make gameplay sessions more varied and compelling. Unlike land units, gryphon riders move by flying, which makes them impervious to many types of attack, and often lethal if their attacks succeed. However, they can be escaped rather easily by fleeing, as they move slowly and need time to power up between attacks. This creates more game variation for plans generated using both algorithms. Games are run on the two topologically different maps described in [6]. Basically, both maps consist of two areas divided by an obstacle. In the first map, there is a single pass through the obstacle, whereas, in the second map, there are two passes, enabling strategic decisions that would be irrelevant on the first map.

Div*CBP* Case Base. We use the case base introduced in [6], which contains 100 plans generated with the FF planner. However, in order to put to test our expectation that the performance of Div*CBP* will be lower with small case bases, and will improve as the size of the case base increases, we vary the size of the case base from 25 to 100 cases, thus simulating the incremental construction of a case base as more and more cases become available. The variant of Div*CBP* using a case base with *n* cases will be referred to as Div*CBP*n. The cases added to the case base at each incremental increase are chosen randomly. That is, we do not specifically tailor the case base towards diversity. However, as the number of cases increases (with new cases being chosen randomly), plan diversity in the case base is also likely to increase.

4.2 Performance Metrics

We use three evaluation metrics to assess the performance of Div*CBP* and Div*FF*. First, we assess how successful the two systems are at producing sets of plans that are diverse based on the specified distance metric. This is the standard type of plan-set diversity assessment conducted previously in generative planning [5,15,19]. Second, we report on planning time. Third, to show how plan-set diversity reflects on actual gameplay variation, we report the end-game score and game duration variation obtained when running the generated plans in Wargus (we conduct similar evaluation in [4,5,6]). We now discuss each of these evaluation metrics in detail.

Plan Set Diversity. As done in [5,15,18,19], we assess how successful the compared planning systems are at generating sets of plans which are diverse by using the diversity metric *Div* (see Formula 6 below) with distance metric D = $D_{\text{Wargus.}}$ Using this distance metric, the maximum diversity value for a set of plans is *0.65*, and

4 Conceptually, the goal of every game is to win, but this cannot be represented through a unique state configuration, especially since enemy information is not known.

corresponds to a set of plans in which each plan uses an army of a different type (out of the 4 available ones: *peasants*, *soldiers*, *gryphon riders*, and *mages*) to attack.

$$Div(\Pi) = \sum_{\pi,\pi' \in \Pi} \frac{D(\pi,\pi')}{\frac{|\Pi| \times (|\Pi|-1)}{2}} \qquad (6)$$

In Formula 6, π is a plan, Π is a non-empty plan set, and D: $\Pi \times \Pi \rightarrow [0,1]$ is a plan distance metric.

Our *hypothesis* with regard to this evaluation criterion is that, when equipped with a sufficiently large case base, Div*CBP* will be more successful than Div*FF* at consistently imposing the given distance metric on the set of generated plans. We expect that the sets of plans obtained using Div*CBP* with case bases of a certain size will have consistently greater *Div* scores than the sets of plans obtained using Div*FF*.

Planning Time. We measure plan generation time in seconds. For Div*FF*, we measure preprocessing time (preprocessing is conducted only once, before all planning sessions) and planning time for each of the four generated plans. For Div*CBP*, we measure retrieval and adaptation time for each of the four plans.

Game Diversity. To examine the connection between plan set diversity, assessed by analyzing the generated plans themselves, and in-game behavior, we run the plans in the game and evaluate the variation of gameplay sessions using the standard deviation of Wargus *score* and gameplay *time* (the duration, measured in game cycles, of gameplay sessions), as in [4] and [6].

Our *hypothesis* is that plan sets obtained using whichever of the two systems produces more highly diverse plan sets consistently will produce greater gameplay variation in most instances. We expect that, when run in the game, plans generated using the system in question will produce more variation (as measured using standard deviation and assessed using the F-test) of Wargus *score* and *time* than plans obtained using the other system. However, game variation is also likely to be affected by the inherent nondeterminism of the game, which can cause even repeated runs of the same plan to produce different results, particularly in terms of game duration.

4.3 Experimental Results

Plan Set Diversity. Plan set diversity values are shown in Fig. 3. It can be observed how Div*CBP* (with α = 0.5, as in [6]) already attains maximum plan set diversity, for all problems, with a case base containing only 30 (out of 100) cases. Furthermore, once this happens, Div*CBP always* generates maximally diverse plan sets (Div = 0.65), thus confirming our expectations.

On the other hand, it is also confirmed that Div*FF* has the advantage of being able to produce maximally diverse plan sets at any time, while Div*CBP* is only able to do so after the case base has become sufficiently large. However, the success of Div*FF* at generating maximally diverse plan sets depends largely on chance (considering the right candidate states first; while Div*FF* can generate all possible plans, this, in practice, is never the case because it would require prohibitively long running time). Out of the 25 plan sets it produces for all problems, only 2 are maximally diverse; the average values per problem attest to less successful diverse plan generation sessions.

In addition, we have ascertained that Div*CBP* can be encouraged to obtain greater plan set diversity by lowering the value of the α parameter in Formula 3 (thus promoting plan diversity over state similarity). When we lower the value of α from 0.5 to 0.35 for Div*CBP*25, the system consistently produces maximally diverse plan sets for all problems (which Div*FF* never achieves).

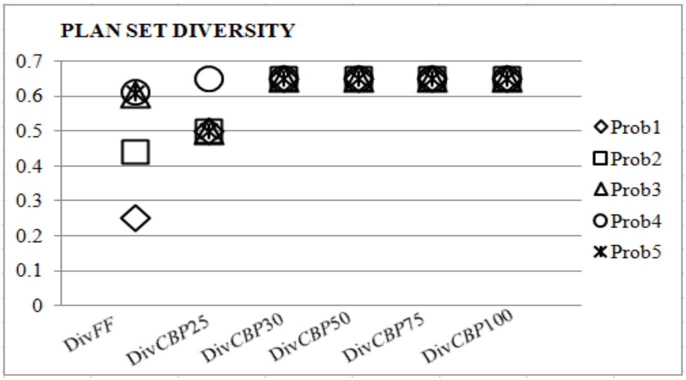


Fig. 3. Plan Set Diversity (Formula 6, with D = D_{Wargus}) for Div*FF* and Div*CBP* with case bases of different sizes

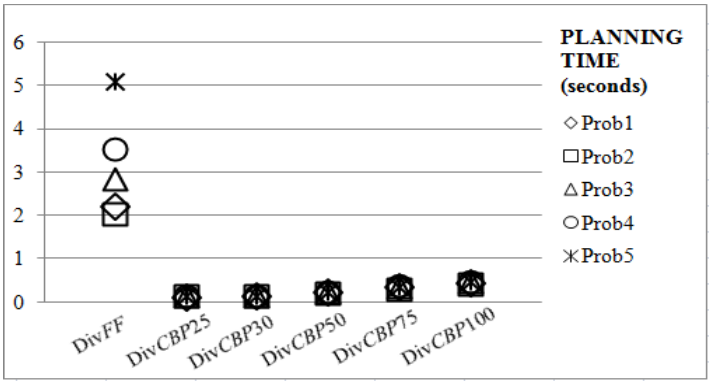


Fig. 4. Planning Time for Div*FF* and Div*CBP* with case bases of different sizes

Planning Time. Fig. 4 shows planning time for Div*FF* and Div*CBP*. All tested versions of Div*CBP* are faster than Div*FF*. Div*FF* planning time increases steeply with the size of the initial state (that is, the number of objects in the initial state, which in our case, consist mostly of unit armies and waypoints that the units can use for movement), due to the grounding bottleneck and to the increase in the number of states in the search space (which is a factor of the number of available objects).

In comparison, Div*CBP* planning time is, on the set of test problems, unaffected by the size of initial states. It increases with the size of the case base, but at a slow rate.

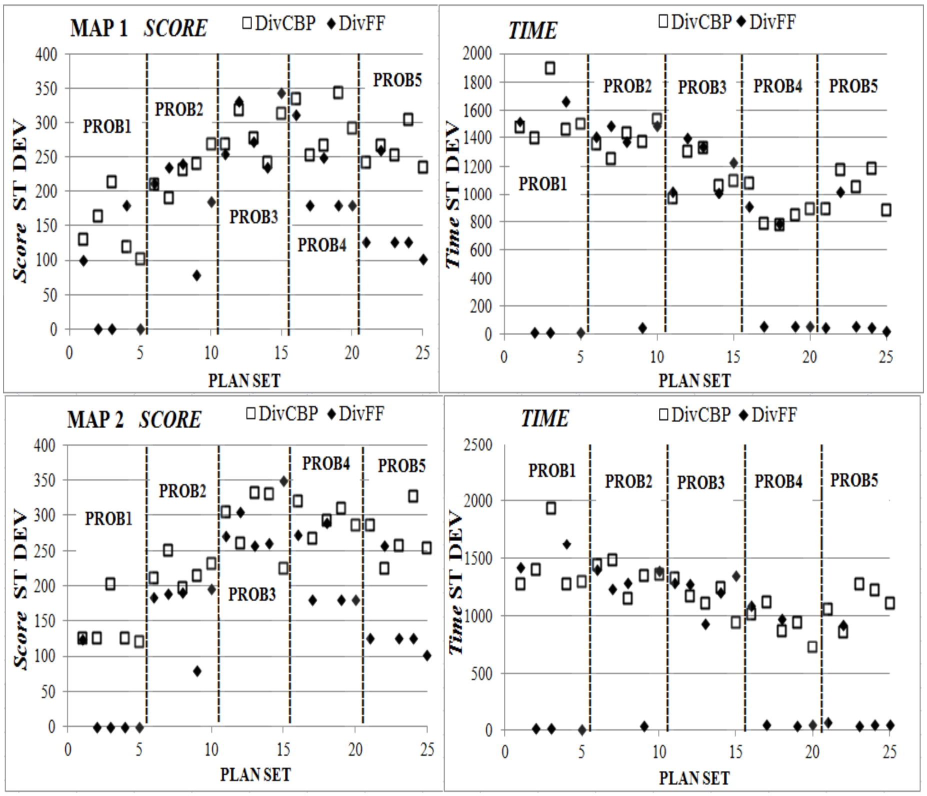


Fig. 5. Standard deviation of in-game results (score and time)

Game Diversity. Score and time (game duration) standard deviation results are shown in Fig. 5. Therein, each point represents the standard deviation of score or time (as indicated) for one plan set of 4 plans, where each plan is run in the game 5 times. There are 5 plan sets for each of the 5 problems, on each of the 2 maps (50 plan sets in all).

The two data sets in each chart correspond to results obtained using Div*FF* and Div*CBP*100 for plan generation. Div*CBP*100 is representative of all Div*CBP* variants which always produce maximally diverse plan sets on all test problems (starting from Div*CBP*30)[5].

For both *score* and *time*, the standard deviation per plan set of the Div*CBP* results is, on average, higher than that of the Div*FF* results on 3 out of 5 problems (Problems 1, 4 and 5) on each map. On the remaining two problems, standard deviation is

[5] As these variants of Div*CBP* produce very similar diverse plan sets, we did not test them comparatively in the game, as any difference in variation between them might more reasonably be attributed to the nondeterminism of the game, than to the particular merits of any of the variants. Beyond obtaining the maximum diversity based on the specified distance metric (which all these variants do), there is nothing more the compared planning systems can achieve in terms of diversity.

roughly the same, with no algorithm being clearly much better than the other in terms of diversity, though Div*FF* plans will occasionally produce results far below the problem average (which is never the case with Div*CBP* plans).

F-test results show that the difference between the variances of the Div*FF* and Div*CBP* *score* and *time* data sets is statistically significant, at the 95% confidence level, for the 3 problems on which the variation of Div*CBP* results is clearly higher, on both maps. For the remaining two problems, the difference in variation is not statistically significant on any of the maps.

5 Related Work

In terms of planning efficiency, advantages of planning using episodic knowledge over planning from scratch have been demonstrated by Veloso [21], Ihrig and Kambhampati [11], Gerevini and Serina [8][9], and van der Krogt and de Weerdt [20], and proved formally by Au, Muñoz-Avila, and Nau [1], and Kuchibatla and Muñoz-Avila [12]. Fox et al. [7] demonstrate advantages of adapting available plans (over replanning from scratch) in terms of plan stability, which is a measure of how many actions in the partial plan need to be revised; the fewer actions need to be revised, the more stable the planning algorithm. Unlike the authors of these previous studies, we are primarily concerned with benefits of plan adaptation over first-principles planning in terms of plan diversity. Plan diversity has been previously obtained in generative planning by Myers and Lee [15] and Srivastava et al. [19]. Solution diversity has also been extensively explored in case-based reasoning outside planning: such work includes that of Smyth and McClave [18], Shimazu [16], McSherry [14], McGinty and Smyth [13], and Bridge and Kelly [3].

6 Conclusions and Future Work

Combining ideas from diversity in case-based reasoning for analysis tasks [18] and plan diversity [15,19], we have developed a conceptual framework for obtaining plan diversity by evaluating candidate solution plans using a combination of solution adequacy and diversity criteria. We have shown how this framework subsumes two systems for diverse plan generation, the case-based planner Div*CBP* and the first-principles planner Div*FF*, and used it to compare these systems. Our analysis revealed that Div*FF* may not be as effective as Div*CBP* in consistently generating highly diverse plan sets, nor as fast. This is due to Div*FF* being characterized by (a) large search spaces, which can be cumbersome to navigate while searching for sufficiently diverse plans, (b) assessment of the diversity of plan sets based on estimates of the solution plans, rather than completed plans, and (c) performance bottlenecks arising from the processing of complete planning models. On the other hand, we expected Div*CBP* to be outperformed by Div*FF* when provided with a case base that is too small to include meaningful variation. Our analysis regarding the comparative strengths and weaknesses of Div*FF* and Div*CBP* was confirmed through experimental evaluation on a real-time strategy game. It was shown that plans obtained with Div*CBP* using case bases of sufficiently large sizes (with this size limit being low) are consistently maximally diverse (based on the qualitative distance metric used), while Div*FF* plan sets obtain lower diversity scores. This difference in variation was also reflected in game sessions obtained by executing the generated plans: on average, games based on Div*CBP* plans were at least as varied as games based on Div*FF* plans (even with inherent in-game variation influencing the results); in fact, for most problem/map combinations, Div*CBP* plans produced results ***significantly more diverse*** than those of Div*FF* plans, as confirmed through statistical significance evaluation.

We believe that an effective way of making use of the two systems complementarily would be to use Div*FF* to initially populate the case base to the point where Div*CBP* can obtain satisfactory diversity for a sufficient number of problems, then to rely largely on Div*CBP*.

In future work, we intend to demonstrate the practical value of plan diversity by showing how dissimilar plans can be used to explore and find the best approach in a (partially) unknown environment.

Acknowledgement. This work was supported in part by NSF grant 0642882 and NRL. We thank the anonymous reviewers for their comments.

References

1. Au, T.-C., Muñoz-Ávila, H., Nau, D.S.: On the Complexity of Plan Adaptation by Derivational Analogy in a Universal Classical Planning Framework. In: Craw, S., Preece, A.D. (eds.) ECCBR 2002. LNCS (LNAI), vol. 2416, pp. 13–27. Springer, Heidelberg (2002)
2. Blum, A., Furst, M.: Fast Planning Through Planning Graph Analysis. Artif. Intell. 90, 281–300 (1997)
3. Bridge, D.G., Kelly, J.P.: Ways of Computing Diverse Collaborative Recommendations. In: Wade, V.P., Ashman, H., Smyth, B. (eds.) AH 2006. LNCS, vol. 4018, pp. 41–50. Springer, Heidelberg (2006)
4. Coman, A., Muñoz-Avila, H.: Case-Based Plan Diversity. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010. LNCS, vol. 6176, pp. 66–80. Springer, Heidelberg (2010)
5. Coman, A., Muñoz-Avila, H.: Generating Diverse Plans Using Quantitative and Qualitative Plan Distance Metrics. In: Burgard, W., Roth, D. (eds.) Proc. of AAAI 2011, pp. 946–951. AAAI Press (2011)
6. Coman, A., Muñoz-Avila, H.: Qualitative vs. Quantitative Plan Diversity in Case-Based Planning. In: Ram, A., Wiratunga, N. (eds.) ICCBR 2011. LNCS, vol. 6880, pp. 32–46. Springer, Heidelberg (2011)
7. Fox, M., Gerevini, A., Long, D., Serina, I.: Plan Stability: Replanning Versus Plan Repair. In: Long, D., et al. (eds.) Proc. of ICAPS 2006, pp. 212–221. AAAI Press (2006)
8. Gerevini, A., Serina, I.: Fast Plan Adaptation through Planning Graphs: Local and Systematic Search Techniques. In: Chien, S., Kambhampati, S., Knoblock, C.A. (eds.) Proc. of AIPS 2000, pp. 112–121. AAAI Press (2000)
9. Gerevini, A., Serina, I.: Efficient Plan Adaptation through Replanning Windows and Heuristic Goals. Fundamenta Informaticae 2(3-4), 287–323 (2010)

10. Hoffmann, J., Nebel, B.: The FF Planning System: Fast Plan Generation Through Heuristic Search. JAIR 14, 253–302 (2001)
11. Ihrig, L., Kambhampati, S.: Storing and Indexing Plan Derivations through Explanation-based Analysis of Retrieval Failures. JAIR 7, 161–198 (1997)
12. Kuchibatla, V., Muñoz-Ávila, H.: An Analysis on Transformational Analogy: General Framework and Complexity. In: Roth-Berghofer, T.R., Göker, M.H., Güvenir, H.A. (eds.) ECCBR 2006. LNCS (LNAI), vol. 4106, pp. 458–473. Springer, Heidelberg (2006)
13. McGinty, L., Smyth, B.: On the Role of Diversity in Conversational Recommender Systems. In: Ashley, K.D., Bridge, D.G. (eds.) ICCBR 2003. LNCS (LNAI), vol. 2689, pp. 276–290. Springer, Heidelberg (2003)
14. McSherry, D.: Diversity-Conscious Retrieval. In: Craw, S., Preece, A.D. (eds.) ECCBR 2002. LNCS (LNAI), vol. 2416, pp. 219–233. Springer, Heidelberg (2002)
15. Myers, K.L., Lee, T.J.: Generating Qualitatively Different Plans through Metatheoretic Biases. In: Hendler, J., Subramanian, D. (eds.) Proc. of AAAI 1999, pp. 570–576. AAAI Press (1999)
16. Shimazu, H.: ExpertClerk: Navigating Shoppers' Buying Process with the Combination of Asking and Proposing. In: Nebel, B. (ed.) Proc. of IJCAI 2001, pp. 1443–1448. Morgan Kaufmann, Seattle (2001)
17. Smyth, B., Keane, M.T.: Adaptation-guided Retrieval: Questioning the Similarity Assumption in Reasoning. Artif. Intell. 102, 249–293 (1998)
18. Smyth, B., McClave, P.: Similarity vs. Diversity. In: Aha, D.W., Watson, I. (eds.) ICCBR 2001. LNCS (LNAI), vol. 2080, pp. 347–361. Springer, Heidelberg (2001)
19. Srivastava, B., Kambhampati, S.T., Nguyen, M., Do, B., Gerevini, A.: Domain Independent Approaches for Finding Diverse Plans. In: Veloso, M.M. (ed.) Proc. of IJCAI 2007, pp. 2016-2022. AAAI Press (2007)
20. van der Krogt, R., de Weerdt, M.: Plan Repair as an Extension of Planning. In: Biundo, S., Myers, K.L., Rajan, K. (eds.) Proc. of ICAPS 2005, pp. 161–170. AAAI Press (2005)
21. Veloso, M.: Planning and Learning by Analogical Reasoning. Springer, Berlin (1994)

Case-Based Appraisal of Internet Domains

Sebastian Dieterle and Ralph Bergmann

Business Information Systems II
University of Trier
54286 Trier, Germany
{diet5101,bergmann}@uni-trier.de
www.wi2.uni-trier.de

Abstract. The increased economic importance of the Internet in all branches of industry has turned Internet domain names into "virtual properties" for electronic commerce. However, the appropriate appraisal of a domain name is a great challenge for trading partners. To address this problem, we propose a new approach derived from the sales comparison approach largely applied in the real estate market, which is implemented using case-based reasoning. The required case base that contains the experience in previous domain trade transactions is automatically extracted from the Internet. Based on carefully selected features derived from the domain name, cases are retrieved using a knowledge-intensive similarity measure. Further, case adaptation is performed to adjust the sales value of a previous domain transaction with respect to the difference to the target domain being assessed. The proposed method is implemented as a prototype Internet Domain Name Appraisal Tool (IDNAT). The performed empirical evaluation using 4,231 cases from the .de domain clearly demonstrates the feasibility of the proposed approach.

Keywords: CBR Application, Appraisal, Case Representation and Mining, Similarity, Adaptation.

1 Introduction

When a new website is to be launched, the selection of the domain name is of crucial importance, particularly due to the increasing shortage in namespaces. Companies and organizations are faced with the difficult task of determining what an appropriate and easy to remember web address is worth. Hence, the appropriate appraisal of an Internet domain such as `hotel.de` is a very challenging task for the partners involved in domain trading transactions. Currently, several domain appraisal services in the Internet offer related services, but existing valuation formulas for domain names are not able to handle the complexity and ambiguity of the natural language used in a domain name. Overall, the valuation process is still not completely understood. Prices achieved in domain trade depend not only on the nature of the respective domain names, but also to some degree on the specific situation of the trading partners such as shortage of money, lack of market knowledge, personal significance of a domain name and the negotiation skills of the market players [1].

To address the problem of domain name appraisal, we propose a new approach derived from the sales comparison approach largely applied in the real estate market [2, 3, 4]. It compares recently sold similar properties (in the local neighborhood) to the target property and adjusts their sales price to compensate for the differences. This method is related to case-based reasoning (CBR) and hence, has already been implemented, thus using CBR for real estate appraisal [5, 6, 7]. However, to our knowledge, CBR has not yet been applied to the appraisal of Internet domains. Therefore, we adapt and elaborate this approach for the appraisal of Internet domains and demonstrate its feasibility. In this approach, the required case base that contains the experience in previous domain trade transactions is of core importance. Publicly available domain sales data, for example, from Namebio [8] or United-Domains AG [9] is an important source from which cases can be extracted. However, the pure strings representing the domain name are not immediately appropriate to assess the similarity of two domains.

It is important that more informative features are derived. This makes it possible to better characterize the domain name with respect to its sales value. We address this problem by deriving an elaborated case representation of domain name transactions containing relevant attributes that can either be calculated from the domain name by a generic function or which can be extracted by use of existing Internet services such as Google, TEOMA [10], or Open Directory Project (ODP) [11]. Hence, we follow the strategy to perform the domain name appraisal based on case-based reasoning on the Web.

The following section describes related work on domain name appraisal and on CBR for appraisal tasks. Section 3 then presents the details of the domain name appraisal approach and the implemented Internet Domain Name Appraisal Tool (IDNAT). The results of an empirical evaluation using an extracted case base of 4,231 domain name transactions from the .de domain are presented in Section 4 before final conclusions are drawn in Section 5.

2 Related Work

2.1 Internet Domain Appraisal

A domain name such as `domain.de` consists of two parts: the freely selectable Second Level Domain (SLD) before the dot and the respective Top Level Domain (TLD) after the dot. The sales value of a domain depends on various aspects. The so-called RICK Formula [12] was the first attempt to make a proposal on the value of a domain. It is well-known in literature and generally applicable, but it is very informal. It claims that a good domain should be free of legal *risks*, i.e., it should not contain terms that refer to other brands or label rights. Furthermore, it has to show a strong *image* TLD, e.g. `.com` for the international market or `.de` for the German-speaking region. The third aspect is the *commercial potential* of a domain, i.e., an assessment as to whether it is suitable for the operation of a profitable business model. The last and most important point during the selection of a domain is the “golden rule”, which states that the domain name should be *kept* as *short* as possible, i.e. the length is a crucial issue.

The fundamentally different SCHARF Model [13] is a factor model that allocates score points for 42 different characteristics of a domain. They are aggregated and with the help of a conversion table, the total number of points is turned into a concrete monetary value. In addition, certain scoring penalties are applied. For example, the value of the domain is decreased if it contains a number, a hyphen or a special character, because this can result in various ways of spelling and hence, spellings mistakes are more likely.

The recently proposed model by Sunyaev et al. [14] weights eleven numerical criteria and determines a precedence of similar domain names by means of a recursive tracking approach. For this purpose, the user initially enters search terms and by means of a Google search, further domain names are found, which in turn, are again used as search terms, etc. The domain names are placed in a precedence list, but a monetary valuation is not executed. This approach is not primarily concerned with assessing, but rather with finding existing domain names, as well as with the creation of valuable domain names with economic potential. In this context, the names that are found are supposed to thematically suit a planned project from a user's point of view. In addition, similar criteria are used as in the two previously described approaches.

Market-oriented domain appraisal approaches, such as Estibot Version 2.0 [15], have recently gained an increasing popularity. Estibot determines the sales value of a domain based on the grounds of similar sales, because these are supposed to best reflect the current market value. Transactions are only considered similar if they exactly contain one word as a substring that occurs in the domain being valued. For example, if the appraisal query is `airlinereservations.de`, all transactions which exactly have `airline` or `reservations` as a substring are displayed. With this, another important valuation criterion is added, i.e., the meaning of the domain name. This reflects the fact that a domain cannot be appraised in a manner independent of the profitability of a branch.

The approach described in this contribution consists of a combination and diversification of the existing appraisal approaches. In particular, it is also based upon earlier transaction cases, but applies a more elaborated similarity function on a larger set of comparison criteria, including structural and textual components.

2.2 Case-Based Appraisal in General

To our knowledge, CBR has not yet been applied to the appraisal of Internet domains, but it was recognized early as a technique capable of supplementing automated approaches for the appraisal of real estate with the help of market data. As an alternative to case-based appraisal, multiple linear regression [16], artificial neural networks [17] and traditional rule-based expert systems [18] have also been applied to appraisal. However, these are not discussed further here.

Gonzalez & Laureano-Ortiz [5] were the first to use CBR to value real estate in 1992. CBR corresponds most closely to the cognitive process of human appraisers. It solves problems by adjusting earlier experiences to a new situation. Since the solution consists only of one single attribute in the appraisal domain, it is very important to find the best possible solution. As a problem description, they used the size of the living space, the number of rooms, the age of the house, the date of sale and the type of heating. The adaptation rules for price adjustment are defined in advance by an expert and the weighted average of the most similar comparison object is computed [5]. Bonissone & Cheetham [6] propose an approach (called PROFIT) for valuing property ownership. In order to define the similarity between the subject entity and the comparison object, this approach combines case-based reasoning with fuzzy logic. McSherry [7] presented an approach implemented with CREST (Case-based Reasoning for ESTimation) that allows for the adjustment of comparison objects independent of the application domain, yet under the assumption of an additive value function.

The appraisal of Internet domains differs from that of real estates since linguistic aspects must be considered for Internet domains and since the geographic proximity plays no role at all. Hence, other criteria that can be automatically derived from the domain name must be determined.

3 CBR for Domain Appraisal

Our approach to case-based domain appraisal requires a case base of experience in previous domain trade transactions. The selected attributes contained in the case representation (described in Section 3.1) must enable an appropriate assessment of the similarity of two domain names and must also provide the information required for adaptation. While the case representation is usually static, the cases themselves must reflect the currently available experience in domain sales transactions on the Web and hence they will be extracted automatically from existing Internet services (see Section 3.2). The automatic extraction is essential in order to keep the case base up-to-date by integrating recent domain sales transactions.

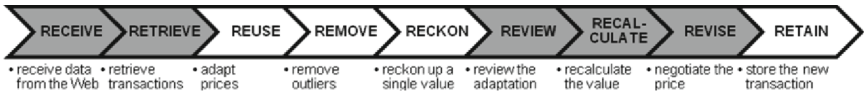


Fig. 1. Pricing a domain name using CBR

To appraise a new target domain, the user simply enters the desired domain name as a query and the appraisal process depicted in Fig. 1 is started. From the domain name, a new query case is generated by automatically deriving the relevant features from the domain name query string. We call this the *receive* step as additional data is received from the web to compute these features. In the subsequent *retrieval* phase, a set of similar cases is retrieved from the case base (currently we use the 11 most similar cases), using a knowledge-intensive similarity measure (see Section 3.3) based on the derived features. Then, the *reuse* phase performs the adaptation of each retrieved case, i.e., of the solution attribute containing the sales value of the sales transaction in the selected case (see Section 3.4). As a result, each retrieved case now has a new sales value assigned that reflects an assessment of the value of the target domain, based on the experience of the individual transaction of the case. In order to *remove* noise from the assessment results, outliers are removed and a weighted average (case similarity is used as weight) of the remaining assessment values is computed (*reckon* step). This value is the proposed appraisal value for the target domain. Professional users may then *review* the computed solution. If necessary, they can tweak the computation process by modifying the used cases and weights (see Section 3.6), which leads to a *recalculation* of the appraisal value. In the *revise* phase, the negotiation of the sales price among the trading partners takes place, based on the appraisal value as a starting point. In case the negotiation is successful, the final sales price is then used in the *retain* phase and leads to a new case that is stored in the case base. Selected details of this approach are presented in the rest of this section.

3.1 Case Representation

The case representation (identical for case and query) was developed such that it contains those characteristics of the domain name that are relevant to determine whether two domains have a similar sales value. A traditional attribute-value representation including symbolic, numeric, textual, and multi-valued attributes was sufficient for this purpose. Altogether, 27 attributes were identified (see Table 1 for the most important attributes). They serve the definition of similarity and are used to control adaptation of the solution, as well as to enable the traceability of the cases.

First, there are basic attributes that form the foundation of every case and from which all other attributes are derived. The basic attributes are the domain name (with SLD and TLD), the date of the transaction (with year and month) and the achieved sales price. The sales price is the only solution attribute in this representation, and hence a value is not available in the query. Neither a usable similarity measure nor useful adaptation rules can be defined directly on the SLD. For this reason, the spelling separated into words and the spelling separated into word components are considered. The difference between the two kinds of spellings is, for example, that the SLD "ArtificialIntelligence" consists of two words and also of two word components. However, SLD "bookworm" is only a single complex word, consisting of two word components ("book" and "worm"). This differentiation is of importance to better discern the semantics of a domain name. On the one hand, cases are retrieved in which the same or related words appear and on the other hand, the same or related word components determine which cases are selected. Although more results can be found with the consideration of individual word components, the quality of the results is higher when complete words are used. As an additional attribute, a classification of the domain name into a set of categories is used. The categories come from a hierarchical classification of the content domains of all web pages world-wide, as provided by the Open Directory Project (ODP) [11]. The category-value in the case representation is constructed in the form of a path in the hierarchy, for example: `World:German:Recreation:Travel:TravelReports:Europe:Spain`.

Table 1. Partial description of a query case and weights for retrieval

Attribute	Exemplary instance	Weight in retrieval
Domain name	`spanienurlaub.de` (German for "holiday in Spain")	-
SLD	spanienurlaub	-
TLD	de	-
Transaction year	2012	0.05
Length	13	0.1
Number of words	2	0.03
Number of word components	2	0.02
Category	World:German:Recreation:Travel: Travel Reports:Europe:Spain	0.15
Words	spanien urlaub	0.1
Word components	spanien urlaub	0.05
Search results	29,100	0.02
Contains hyphen	false	0.02
Number of hyphens	0	0.01
Contains special characters	false	0.01
Contains numbers	false	0.01
Domain age	1999.25	0.01
Global monthly searches	9,900	0.01
Local monthly searches	9,900	0.1
Avg. CPC in €	4.3	0.01
Daily clicks	23.79	0.01
Daily cost in €	102.33	0.05
Sales price in €	unknown	-

Furthermore, attributes that describe the length of a domain name are included. This relates to the number of individual characters, the number of words, as well as the number of word components. Other visible, linguistic characteristics of domains are differentiated by whether or not and how many hyphens or special characters such as ä, ö, ü, ß, á, à etc. or numbers exist in the SLD.

Other characteristics that have similar relevance must also be considered. This includes the domain age, i.e., the date of registration. In addition, the number of search results in search engines must be considered, i.e., the number of websites that contain the SLD in a separated spelling. Furthermore, the number of searches performed worldwide and regionally in countries related to the TLD is a characteristic that is also included, i.e., how often the search term is searched for per month. In addition, the average price per click on the advertisement for the corresponding search term, the number of daily clicks, and the total costs for advertising the search term are considered.

3.2 Data Extraction

The information regarding the selected characteristics must be calculated in an automated manner or extracted from the Internet in order to be able to automatically update the case base.

First, domain sales transactions were extracted from the Internet using existing services, such the world's largest public list from Namebio [8] or the domain sales list from United-Domains AG [9]. Double entries, obviously erroneous entries, syntactical incorrect values, as well as older values were filtered out. Then, the secondary attributes had to be derived from the extracted domain transactions. They can either result from the extraction of data from the Internet or from the combination of existing attributes. Here, a large number of specific algorithms were implemented (see Dieterle [19] for details). For example, based on a heuristic, the domain name is divided into the individual words and word components if these are not already separated by hyphens. This takes place through the assessment of how common the variant forms of spelling are by using Google queries.

Among other things, the SLD must be classified in the categories of the Open Directory Project (ODP) [11]. The ODP is the world's largest web directory, which is maintained by nearly 95,000 volunteer editors. It contains a hierarchy of more than 1,010,000 categories and may be used open source, for example, for the development of Artificial Intelligence (AI) applications. A large number of research projects [20] make use of this opportunity. During the data extraction, up to three different, compounded categories are extracted, to address the potential ambiguity of the words. Thus, a domain such as `crimefiction.de` fits in a category for crime novels, for crime series, and also into the category for computer games related to the genre of crime.

3.3 Similarity

This subsection presents the similarity measure used during the retrieval of similar cases. Please note that the goal is to define a knowledge-intensive similarity measure such that it approximates the *utility* of the case (see [23] chapter 4) for the current query. Given the fact that the sales value of a case is possibly adapted after retrieval, a case should also be considered useful to a query if its sales value can be adapted such that it is similar to the real sales value of the domain in the query.

Given the traditional attribute-value representation for cases, the well-established local-global approach for defining similarity measures is applied. As an aggregation function, the weighted sum of 18 local similarity functions is used. Table 1 illustrates how the local similarity functions are weighted. The weighting has an experimental character and is based on the following general considerations:

- attributes with a high relevance for the value of the domain should have a high weight,
- attributes whose differences can be well compensated by adaptation should have a low weight.

Different types of local similarity functions are defined, depending on the type of attribute: textual similarity functions define the similarity between two unstructured strings (e.g. the SLD), symbolic similarity functions define the similarity between symbolic values (e.g. categories), and numerical similarity functions compare numerical attributes (e.g. the length of a domain). These similarity measures are further specialized for the respective attribute (see Dieterle [19] for details).

In the following, two selected local similarity measures are presented. The first similarity measure $sim_{words}(Q,C)$ allows for the semantic comparison of words (or word components) of a domain.

$$sim_{words}(Q,C) = \frac{\sum_{q \in Q} \max(\{sim_{word}(q,c) | c \in C\}) + \sum_{c \in C} \max(\{sim_{word}(q,c) | q \in Q\})}{|Q| + |C|} \tag{1}$$

The arguments Q and C stand for a set of words contained in the query or case, respectively. The maximum similarity is derived for every individual pair of words using the similarity measure $sim_{word}(q,c)$. This measure compares two single words (or word components) q and c with each other.

$$sim_{word}(q,c) = \begin{cases} 1 & if\ q = c \\ \frac{\min(length(q), length(c))}{\max(length(q), length(c))} & if\ \begin{matrix} \exists s \in substring(q): s = c\ \vee \\ \exists s \in substring(c): s = q \end{matrix} \\ 0.8 & if\ \begin{matrix} \exists r \in relatedWord(q): r = c\ \vee \\ \exists r \in relatedWord(c): r = q \end{matrix} \\ 0 & otherwise \end{cases} \tag{2}$$

If q and c are identical, the similarity is the maximum. If one word is a sub-string of the other word, the similarity corresponds to the degree of overlap of both words. For example, "business" and "businesses" correspond with eight of ten letters (0.8); while "men" and "investment" correspond with only three of ten letters (0.3). In the case that the two words are related to one another, a reasonably high similarity value of 0.8 is assumed. The relatedness of two words is determined using the search engine TEOMA [10]. When entering a search term into TEOMA, a list of related search terms are also displayed. They are determined by TEOMA by mining frequent sequences of terms that users enter as queries. For example, the terms beach, ocean and palm trees are related to the term vacation. Alternatively, computational-linguistic approaches could have been used for semantic relatedness and similarity [21, 22], but they seem less useful for our purpose, as they are based on the occurrence of words in written documents or in dictionaries such as WordNet. Hence, they are not necessarily aligned with the natural search behavior of Internet users and the vocabulary used during a direct input of an Internet address.

The second local similarity measure that we now present is used to compare the category attribute. Here, a taxonomy is used to determine terminological similarity. Taxonomies are n-ary trees, whose nodes contain symbolic values. Unlike the taxonomy similarity measures proposed by Bergmann [23], we apply a graph-theoretical approach [24], which does not require assigning similarity values to each node of the taxonomy. Given the fact that the range of categories spans the whole world-wide spectrum of possible topics of a Web page, it is obvious that this taxonomy is extremely large (as we use the Open Directory Project it contains more than 1,010,000 categories), hence a manual assignment of similarity values is not feasible. Instead, the distance between the nodes in the taxonomy is used to define a similarity measure. The similarity between the category of a query and a case is computed by the following formula:

$$sim_{category}(q,c) = \begin{cases} 0 \; if \; countEdgesBetween(q,c) > 10 \\ 1 - 0.1 \cdot countEdgesBetween(q,c) \text{ otherwise} \end{cases} \tag{3}$$

The example in Fig. 2 plots the similarity between the node "women's soccer" and the node "Soccer World Cup 2006". Since there are four edges between them, the similarity according to (3) is 0.6. This form of similarity determination is efficient because it only considers the categories from the query and the case (and their paths in the taxonomy). The remaining 1,010,000 categories do not need to be considered.

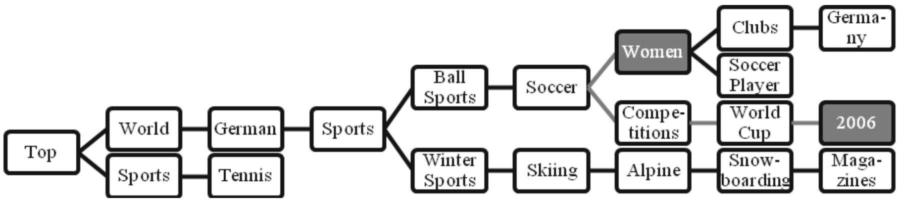


Fig. 2. Example of the distance between two cases within the taxonomy

3.4 Adaptation

The most similar cases found are not just reused unmodified, but their solutions, i.e., the sales value attribute is adapted by use of multipliers. In the area of company valuation (particularly when using the market value method), it is well-known that comparable objects (previously in the context of the branch, currently more in the context of factors such as sales growth, etc.) are adapted by means of multipliers representing factors such as profit, cash flow, or number of customers [25, 26]. Similar to the local-global principle of the similarity assessment, the adaptation multipliers used in our approach are decomposed into *local* and *global multipliers* (Eqn. 4).

$$c'_p := c_p \cdot \prod_{a \in A} \underbrace{\left(\frac{i_a(q_a)}{i_a(c_a)}\right)^{w_a}}_{local\ multiplier} \tag{4}$$

The global multiplier is computed by the weighted product of the individual local multipliers determined for selected attributes A relevant for adaptation. For each local multiplier, an attribute-specific index function i_a computes a valuation index based on the value of a single attribute $a \in A$. The ratio of the index values for the query attribute q_a and the case attribute c_a determines the local multiplier, additionally taking the attribute weight w_a into account. To adapt the sales price c_p of a case c (c'_p is the adapted price), we consider the attributes "time", "length of the domain name", "number of searches at Google", "occurrence of hyphens", as well as "occurrence of special characters".

Next, an example local multiplier for the attribute “time” is presented. It is based on the index values from the Internet Domain Name Index (IDNX) [1]. This is a price index ranging from 2006 until the current day, which indicates the development of the price level in the domain market. Methodically, neither the average price (vulnerable to outliers) nor the median (does not consider the quality of the traded domain names) is used, but instead the hedonic repeat-sales method [27] is applied. The very rare occurrence of a repeated sale of the same domain serves as the control variable. Otherwise, the method is based on the comparison of very similar domain transactions over time. These are domain transactions that only differ in the TLD, prior to compensating TLD-specific difference in the price [1]. A detailed description of the local multipliers for the other attributes is given by Dieterle [19].

After each of the k-most similar cases C_q is adapted, cases that are considered outliers O_q are detected. Outliers are defined as the cases whose adapted price c'_p deviates “too much” from the median. Here, “too much” means less than half or more than double as high as the median of the k-most similar cases. Outlier cases are not considered in the subsequent computation of the overall appraisal value q_p. It is determined according to (5) by a weighted average of the adapted prices c'_p of the most similar cases using the similarity values of the cases as weights w_c^q.

$$q_p := \frac{\sum_{c \in C_q - O_q} (c'_p \cdot w_c^q)}{\sum_{c \in C_q - O_q} w_c^q} \tag{5}$$

3.5 A Simple Example

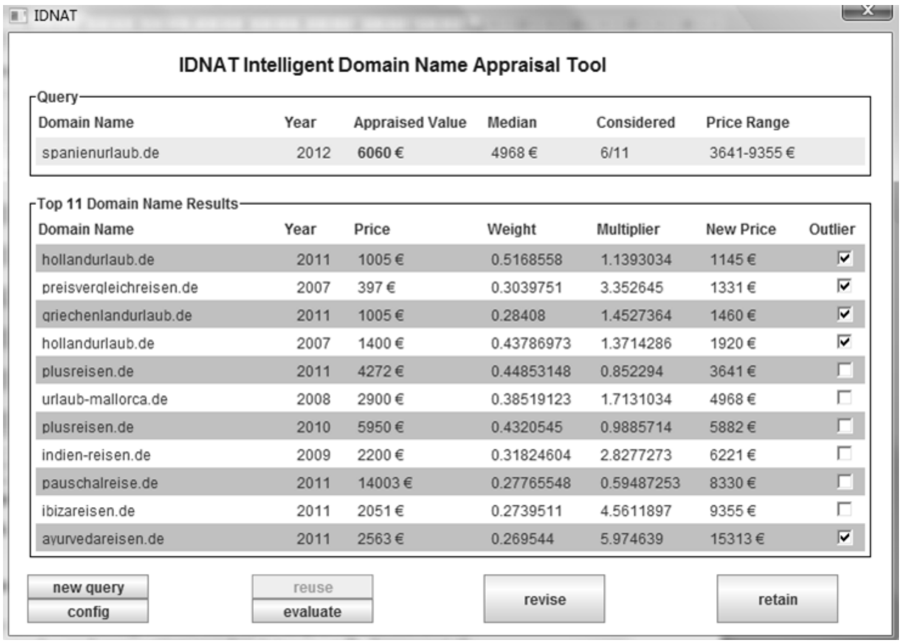
The appraisal calculation for the target domain `spanienurlaub.de` (German for “holiday in Spain”) in 2012 is shown as an example in Fig. 3. The eleven most similar domain transactions are illustrated on the left side as is the respective year and price. Each case contains one of the words “Urlaub”, “Reise” or “Reisen” (“vacation”, “trip” or “trips” in German), i.e., a word that belongs to the tourism branch. In each case, this word is combined with a second word, hence the length of the domain name approximately corresponds to the length of the query. Furthermore, most of the cases have a transaction year similar to the query. Six of the eleven cases also include a region name, from which three regions belong to the Mediterranean area. Hence, the similarity measure is able to identify cases from closely related web pages. The right side of Fig. 3 shows the weights, the global multipliers determined during adaptation, as well as the new prices resulting from the adaptation. The results are sorted by price in ascending order and the cases that deviate too much from the median are marked as outliers. They are not considered for the overall price calculation. In this example, the appraised value for the domain `spanienurlaub.de` is 6,060 €.

3.6 Tweaking Support for Professional Users

Aside from the automated appraisal that was just described, the developed approach is also capable of supporting experts in the appraisal of domain names by providing additional means of tweaking the results. This is particularly useful for commercial domain assessors and publicly appointed and sworn experts with strong knowledge in the field. Using a revision function, the expert user has three options to tweak the results: He/She can change the weighting of the most similar cases manually. If it does not correspond to his/her estimation, he/she can manually change the multipliers when he/she has reasonable grounds to believe that a different value relationship exists between the case and the query. Moreover, he/she can change whether certain cases are considered as outliers or not. The new appraisal value is automatically re-calculated on this basis.

4 Experimental Evaluation

The presented approach is fully implemented as a prototypical system, called Internet Domain Name Appraisal Tool (IDNAT). The implementation was done in JAVA using the object-oriented CBR framework jCOLIBRI2 [28]. In order to perform an experimental evaluation, a case base consisting of 4,231 cases describing domain sales transactions with the TLD .de was extracted from the Internet. For this purpose, the domain transaction list of the United-Domains AG [9] was used, since it is easily accessible and relatively comprehensive with over 1,000 .de entries. Furthermore, the world's largest public list from Namebio [8], with over 3,000 relevant entries, was also used.



Query

Domain Name	Year	Appraised Value	Median	Considered	Price Range
spanienurlaub.de	2012	6060 €	4968 €	6/11	3641-9355 €

Top 11 Domain Name Results

Domain Name	Year	Price	Weight	Multiplier	New Price	Outlier
hollandurlaub.de	2011	1005 €	0.5168558	1.1393034	1145 €	✓
preisvergleichreisen.de	2007	397 €	0.3039751	3.352645	1331 €	✓
griechenlandurlaub.de	2011	1005 €	0.28408	1.4527364	1460 €	✓
hollandurlaub.de	2007	1400 €	0.43786973	1.3714286	1920 €	✓
plusreisen.de	2011	4272 €	0.44853148	0.852294	3641 €	☐
urlaub-mallorca.de	2008	2900 €	0.38519123	1.7131034	4968 €	☐
plusreisen.de	2010	5950 €	0.4320545	0.9885714	5882 €	☐
indien-reisen.de	2009	2200 €	0.31824604	2.8277273	6221 €	☐
pauschalreise.de	2011	14003 €	0.27765548	0.59487253	8330 €	☐
ibizareisen.de	2011	2051 €	0.2739511	4.5611897	9355 €	☐
ayurvedareisen.de	2011	2563 €	0.269544	5.974639	15313 €	✓

Fig. 3. Price calculation for a query

The purpose of the experimental evaluation was to assess the appraisal quality, i.e., the correlation between proposed appraisal value by IDNAT and the real sales value of previous domain sales transactions. Additionally, we aim to evaluate the impact of the weights used during the similarity assessment. For this purpose, three weighting profiles were defined as shown in Table 2, which differ in the balance between semantic and numeric aspects.

Table 2. Weighting profiles for the evaluation

Experimental design	Basic assumption	Less semantic	More semantic
Semantic aspects	37.04 %	22.73 %	54.05 %
Numerical aspects	62.96 %	77.27 %	45.95 %

The semantic aspects refer to the meaning of the domain name and thus, contain the three attributes “words”, “word components” and “category”. The numerical aspects refer to the number of searches, the number of words, the length, etc., i.e., to all further attributes. The weighting profiles were selected for the evaluation because we observed that, quite often, only a small number of retrieved cases are highly similar with respect to both categories of features.

The evaluation was performed using a leave-one-out test. Hence, each case of the case base is used as query to the system – the case used as query is temporarily removed from the case base. Then, the percentage of correctly appraised queries is determined. An appraisal is defined to be correct if the value is not less than half and not more than double the real sales price. Such a wide range is necessary due to the large price differences that even occur for repeated domain transactions. Furthermore, the correlation coefficient (also called Pearson product-moment correlation coefficient) and the Spearman's rank correlation coefficient are calculated. The evaluation shows how close the system’s appraisal is to the real value and vice versa. Thus, it may also be possible to gain insight as to how fair the actually achieved market prices are. The results are illustrated in Table 3 and Fig. 4.

Table 3. Evaluation Results

Experimental design	Basic assumption	Less semantic	More semantic
Correctly valued	46.02 %	46.84 %	42.68 %
Pearson’s correlation coefficient	0.6729	0.5344	0.6452
Rank correlation coefficient	0.5742	0.5952	0.5214

The evaluation showed that using the profile basic assumption, 46 % of the 4,231 cases were correctly appraised. The average retrieval and adaptation time was 6.1 seconds, which is quite acceptable. The correctly appraised cases are also plotted in the inner interval of Fig. 4. Only 7% of the cases lie outside the outer interval, which are extreme outliers whose estimated value is less than one-tenth or more than ten times the achieved price. For these cases, we are speculating that most of them reflect incorrect sets of data or they are not pure domain transactions, but rather transactions involving entire projects or companies.

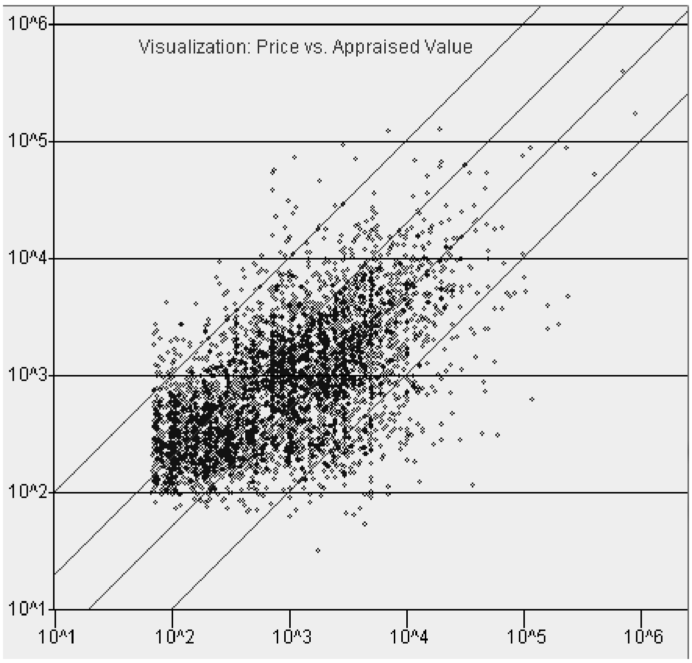


Fig. 4. Visualization of the real price vs. the estimated value (for the basic assumption profile)

The results also demonstrate that the real price and the estimated value have a clear positive correlation. The "More semantic" profile has a negative impact on the quality of the results. This relates both to the defined correctness criteria as well as to the correlation according to both correlation coefficients. On the other hand, the "Less semantic" profile does not deliver clear results. We believe that the reason for the negative impact of the more semantics profile is due to the fact that the prices of Internet addresses within a branch can be very different. If the numeric attributes differ too much and hence, very high or very low adaptation multipliers are used, the adaptation becomes less reliable.

While our evaluation was able to assess our approach in isolation, it does not allow for an assessment of it with respect to currently available assessment methods that follow an idea similar to the market-oriented domain appraisal approach used in Estibot. A comparison with existing approaches has two fundamental problems. First, most available services only allow for a very limited number of free queries per day. Secondly, a leave-one-out evaluation is not possible, since there is no means of explicitly excluding cases during the appraisal. Only a complete re-implementation could solve this problem, which however is difficult, since not all of the details of the appraisal algorithms are disclosed.

5 Conclusion and Future Work

This paper clearly demonstrates that domain names can be appraised using a CBR system. However, the evaluation also unveils the fact that a database of approximately

4,200 cases may be too small to detect a sufficient number of comparable transactions with adequate similarity for a complete TLD. Hence, an important starting point for future work is an enlargement of the case base. This involves including additional domain transaction lists as well as TLDs other than `.de`. In particular, special considerations are required to enable the adaptation across different TLDs. This is particularly dependent on the language and the popularity of a term in the regions covered by the TLDs. With regards to the similarity between categories, existing links between the same categories in different languages must be considered within the taxonomy.

The Internet administration ICANN intends to allow unlimited TLDs in the near future. It is expected that the amount will be in the four-digit range. It would be interesting not only to include a limited number of TLDs in the model, but to generalize it towards a completely generic approach, accepting arbitrary combinations of SLD and TLD.

The current computation time of a query increases linearly with the size of the case base because a simple sequential retrieval algorithm is used. With this approach, we estimate that the retrieval time would be almost five minutes for a case base of a realistic size of 200,000 transactions. Hence, more efficient retrieval methods are required, e.g., a two-step MACFAC approach.

References

1. Lindenthal, T.: Valuable Words: Pricing Internet Domain Names. Working Paper (2011), `http://www.idnx.com/working_paper_IDNX.pdf`
2. Lusht, K.M.: Real Estate Valuation: Principles & Applications. IRWIN, Chicago (1997)
3. Cortesi, G.R.: Mastering Real Estate Principles. Dearborn, Chicago (1996)
4. Mettling, S., Cusic, D.: Principles of Real Estate Practice. Performance Publishing Company, Bradenton (1996)
5. Gonzalez, A.J., Laureano-Ortiz, R.: A Case-Based Reasoning Approach to Real Estate Property Appraisal. Expert Systems With Applications 4, 229–246 (1992)
6. Bonissone, P.P., Cheetham, W.: Financial Applications of Fuzzy Case-Based Reasoning to Residential Property Valuation. In: Proceedings of the Sixth IEEE International Conference on Fuzzy Systems, pp. 37–44. IEEE (1997)
7. McSherry, D.: An Adaptation Heuristic for Case-Based Estimation. In: Smyth, B., Cunningham, P. (eds.) EWCBR 1998. LNCS (LNAI), vol. 1488, pp. 184--195. Springer, Heidelberg (1998)
8. Namebio, `http://www.namebio.com`
9. united-domains AG, `http://www.domain-spiegel.de`
10. IAC Search & Media, Inc., `http://www.teoma.com`
11. AOL Inc., `http://www.dmoz.org`
12. Huber, F., Dingeldey, D.: Ratgeber Domain-Namen, united-domains AG, Starnberg (2001)
13. Benken, B.: `http://www.bewertungsformel.de`
14. Sunyaev, A., Kaletsch, A., Krcmar, H.: Erfassung, Analyse und Bewertung von Domainnamen: Modellansatz und prototypische Umsetzung. Informatik-Spektrum 35(1) (2012)
15. estibot.com inc., `http://www.estibot.com`
16. Renshaw, E.F.: Scientific Appraisal. National Tax Journal 11, 314–322 (1958)
17. Do, A.Q., Grudnitiski, G.: A Neural Network Approach to Residential Property Appraisal. The Real Estate Appraiser 58, 38–45 (1992)

18. Czernowski, R.M.J.: Expert Systems in Real Estate Valuation. Journal of Property Valuation and Investment 8(4), 376–393 (1990)
19. Dieterle, S.: Marktorientierte Bewertung von Internet-Domainnamen – Konzeption eines fallbasierten Systems. Bachelor Thesis, University of Trier (2011)
20. Chirita, P.A., Nejdl, W., Paiu, R., Kohlschütter, C.: Using ODP Metadata to Personalize Search. In: SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 178–185. ACM, New York (2005)
21. Kolb, P.: Experiments on the difference between semantic similarity and relatedness. In: Proceedings of the 17th Nordic Conference of Computational Linguistics NODALIDA 2009, vol. 4, pp. 81–88. Northern European Association for Language Technology (2009)
22. Budanitsky, A., Hirst, G.: Evaluating WordNet-based Measures of Lexical Semantic Relatedness. Computational Linguistics 32, 13–47 (2006)
23. Bergmann, R.:Experience Management: Foundations, Development Methodology, and Internet-Based Applications. LNCS (LNAI), vol. 2432. Springer, Heidelberg (2002)
24. Bunke, H., Messmer, B.T.: Similarity Measures for Structured Representations. In: Wess, S., Richter, M., Althoff, K.-D. (eds.) EWCBR 1993. LNCS, vol. 837, pp. 106–118. Springer, Heidelberg (1994)
25. Boatsman, J.R., Baskin, E.F.: Asset Valuation with Incomplete Markets. The Accounting Review 56, 38–53 (1981)
26. Alford, A.: The Effect of the Set of Comparable Firms on the Accuracy of the Price-Earnings Valuation Method. Journal of Accounting Research 30, 94–108 (1992)
27. Shiller, R.J.: Measuring Asset Values for Cash Settlement in Derivative Markets: Hedonic Repeated Measures Indices and Perpetual Futures. The Journal of Finance 48(3), 911–931 (1993)
28. GAIA – Group of Artificial Intelligence Applications, `http://gaia.fdi.ucm.es`

Harnessing the Experience Web to Support User-Generated Product Reviews

Ruihai Dong, Markus Schaal, Michael P. O'Mahony, Kevin McCarthy, and Barry Smyth

CLARITY: Centre for Sensor Web Technologies
School of Computer Science and Informatics
University College Dublin, Ireland
{ruihai.dong,markus.schaal,michaelp.o'mahony,
kevin.mccarthy,barry.smyth}@ucd.ie
http://www.clarity-centre.org

Abstract. Today, online reviews for products and services have become an important class of user-generated content and they play a valuable role for countless online businesses by helping to convert casual browsers into informed and satisfied buyers. In many respects, the content of user reviews is every bit as important as the catalog content that describes a given product or service. As users gravitate towards sites that offer insightful and objective reviews, the ability to source helpful reviews from a community of users is increasingly important. In this work we describe the Reviewer's Assistant, a case-based reasoning inspired recommender system designed to help people to write more helpful reviews on sites such as Amazon and TripAdvisor. In particular, we describe two approaches to helping users during the review writing process and evaluate each as part of a blind live-user study. Our results point to high levels of user satisfaction and improved review quality compared to a control-set of Amazon reviews.

1 Introduction

Among the many varieties of user-generated content on the social web, user-generated product reviews have come to play a vital role in a wide range of product/service oriented settings. Today, leading online product and service providers, from Amazon to TripAdvisor and iTunes to Etzy, prioritise user-generated reviews alongside catalog content in a variety of ways. For example, Amazon prioritises such reviews by placing the average review score directly beneath a product title. Moreover, it provides interested users with one-click access to review content, which includes recommendations of the most helpful, favourable and critical reviews. The rationale for such access is that review information of this kind has been shown to play a valuable role when it comes to helping users to make purchasing decisions, increasing the conversion rate of casual browsers into informed, satisfied buyers. For example, Hu et al. [10] describe the results of one study on the value of online reviews, concluding that consumers do understand the value difference between favourable and unfavourable opinions and respond accordingly. Furthermore, when consumers read online reviews, they pay attention not only to review scores, but to other contextual information such as a reviewer's reputation and reviewer exposure. The market responds more favourably to reviews written by reviewers with better reputation and higher exposure. In related work, Zhu et al. [16] examine the influence of online reviews on video game sales, indicating that reviews are more influential for less popular games.

While many sites prioritise the promotion of user-generated review content, they do little to support users when it comes to generating reviews, beyond the provision of a simple text-input review-form. At best, this provides friction when it comes to attracting new reviews from first-time reviewers, but further it may negatively impact the quality of the reviews that are provided. For this reason researchers have begun to consider ways in which users can be better supported during the review writing process. For example, the seminal work of Bridge et al. [3] describes a system called GhostWriter, which is designed with this task in mind. Briefly, GhostWriter uses case-based reasoning techniques to harness a collection of past review experiences, which are then used as a source of suggestions for reviewers as they write. Essentially, GhostWriter suggests fragments of similar reviews as hints to the reviewer.

In this paper, we build on the GhostWriter idea and extend it in a number of important ways. Firstly, we use association rule mining techniques to extract correlated product features from raw review experiences. These features correspond to recurring review fragments across a collection of reviews; for example, we might notice that digital camera reviews which refer to *picture quality* and *color saturation* also refer to *white balance*; see also [8]. Secondly, we consider a new topic-oriented approach that allows us to map raw review fragments to more structured product topics, and so avoid recommendation redundancy and improve the breadth of coverage of suggestions. Thirdly, we describe how these ideas have been implemented in the *Reviewer's Assistant* (RA) in the form of a browser plugin, which facilitates direct integration with sites like Amazon, TripAdvisor, etc. Finally, this paper also includes a comprehensive live-user trial of RA, in which we evaluate the perceptions of end-users and the quality of the reviews they produce; ultimately demonstrating how use of the RA can lead to higher quality reviews than those currently found on Amazon.com, at least in terms of the evaluation domain of digital cameras.

2 Related Work

Recent research indicates that online product reviews have a significant influence on the purchasing behaviour of users; see [10,16]. For example, the effect of consumer reviews on book sales on Amazon.com and Barnesandnoble.com has been investigated by Chevalier & Mayzlin [5], concluding that the relative sales of books on a site correlates closely with positive review sentiment; although interestingly, there was insufficient evidence to conclude that retailers themselves benefit from making product reviews available to consumers. Similarly, Dhar & Chang [7] found a correlation between the volume of blog posts about a music album and future sales. Likewise, it has been shown that the early volume of online movie reviews can be used as a proxy for early sales [6].

Given the importance of product reviews it is not surprising that retailers and researchers have started to study different ways to help interested users find high quality reviews for products they are considering. As the volume of user-generated review content grows, it will become increasingly important to rank reviews based on some measure of relevance and/or helpfulness. For example, the work of [11–14] each consider different factors such as reviewer reputation, product genre familiarity, and review recency to automatically rank reviews based on their predicted helpfulness. Indeed, as the importance of online reviews has increased so has the temptation for interested parties to manipulate reviews to generate a bias for or against particular products. This in turn has motivated researchers to consider ways in which suspicious reviews can be identified and eliminated; see for example the work of Wu et al. [15].

In this work, we approach the problem of review quality from a different but complementary perspective: rather than attempt to rank reviews by helpfulness, or eliminate biased reviews, we focus on the process of writing a review in the first place. Our aim is to not only produce a better quality of user-generated reviews, but also to increase the number and diversity of reviews by attracting first-time reviewers who might be initially daunted at the prospect of writing a product review. Our work is inspired by the GhostWriter system, first introduced by Bridge et al. [4], as an approach to supporting users to create online adverts for personal goods and items they wish to dispose of, by using conversational CBR techniques [2] to reuse fragments of adverts of previously posted items. Later Healy and Bridge [9] adapted this approach to support users during the generation of product reviews, again using conversational CBR techniques, but this time recommending review fragments from past reviews that are similar to the user's current, incomplete review. Currently GhostWriter 2.0 [9] extracts noun phrases from these past product reviews (cases), and suggests these phrases directly to the reviewer/user.

Our Reviewer's Assistant is a close relative of the GhostWriter systems. It too harnesses past review experiences to proactively suggest review topics as the user writes their review. In this paper, we will describe how the RA combines association rule mining and topic extraction techniques with conversational CBR to generate these recommendations. In addition, the RA has been designed to fully integrate with existing online services such as Amazon and TripAdvisor, allowing users to use existing review tools while benefitting from suggestions made by the system.

3 The Reviewer's Assistant

The Reviewer's Assistant has been implemented as a browser plugin so that it can seamlessly integrate with pre-existing services like Amazon and TripAdvisor at the user interface level, providing support to users as they write their reviews, but without the need for backend integration with these underlying services. In

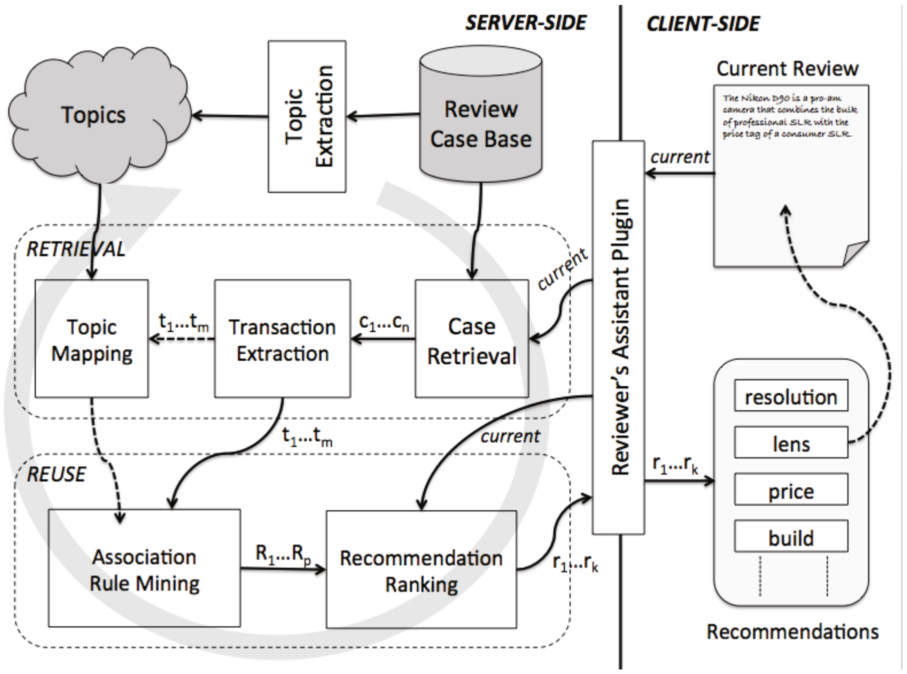


Fig. 1. The Reviewer's Assistance System Architecture

this section we will describe the basic system architecture and key technical features of the RA system and provide an example of the RA in operation.

Figure 1 provides a high-level overview of the Reviewer's Assistant system. The basic recommendation cycle is triggered on an ongoing basis as the user writes their review. During each recommendation cycle the user's current review text is used to identify a set of similar review cases. These cases are mined to extract a set of term-based association rules. These rules are then used to map the current review text to a ranked list of concrete topic recommendations for the user.

3.1 Case Discovery and Topic Extraction

Before describing the technical details of the basic RA recommendation cycle we will first look at the structure and source of case and domain knowledge.

The Review Case-Base. The RA is designed to operate over specific product domains and maintains a separate review case-base for each. A domain is assumed to be made up of a collection of products that share similar features, which ultimately will act as possible review targets for the reviewer. Each case corresponds to a previous review and includes the product id, the review text, and any meta information available, such as the overall review score or helpfulness. These reviews are automatically extracted from the underlying service by using any available API to extract relevant product and review data; typically the RA will extract only high-quality reviews (based on any helpfulness/quality meta data that is available).

Topic Extraction. In this paper we consider a version of the RA that maps review terms to product topics in order to improve recommendation quality. For the purpose of this feasibility study we adopt a very simple approach to topic modeling based around a hand-coded set of topics for the target domain, with each topic associated with a synonym set. As such the topic extraction component of Figure 1 is not currently automated but rather left as a matter for future work – assuming the value of our topic-based variation can be demonstrated in live usage.

3.2 Retrieving Similar Reviews

As the user writes their review, the review text is periodically (typically on the completion of a new sentence) used as query *current* against the relevant domain case-base to retrieve a set of similar reviews, from which term-based transactions are extracted as the basis for association rule mining.

Case Retrieval. In the current implementation we rely on a simple term-based Jaccard similarity metric to retrieve a set of the n review cases that are most similar to *current*. At present, this retrieval process is further restricted to only consider review cases that match the target review product id. For example, if the user is reviewing the latest "Nikon D90" camera, then only past reviews of this product will be considered for retrieval. Obviously this condition could be relaxed to facilitate the consideration of similar products but this is left as a matter for future work.

Extracting Transactions. At this point each of the n retrieved reviews is converted into a set of sentence-level transactions and a review-level transaction. This is a straightforward process that starts by identifying the nouns in a review text and then converts each sentence or review into a set of these nouns based on their order of appearance; see Figure 2. In the next subsection, we will describe how these transaction-based representations can be used by association rule mining techniques as a basis for recommendation.

Topic Mapping. In one version of the RA, *non-topic*, recommendation proceeds based on the mining of these noun-based transactional representations. However, in this paper we also consider a topic-based approach (we refer to this simply as *topic*) which first maps the raw transaction terms onto the topics extracted from a given review collection as mentioned above. The purpose of this approach is that it affords a level of abstraction (topics vs. nouns) that has the potential to provide a more intuitive set of recommendations based on more

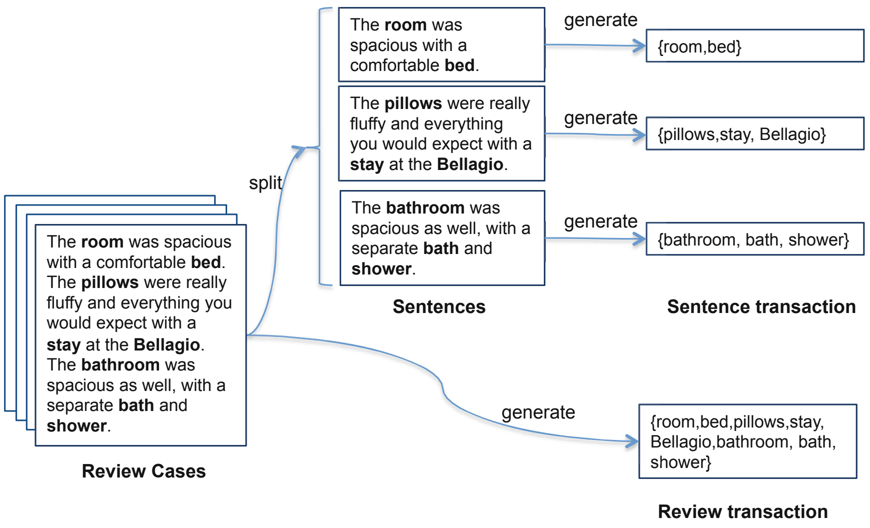

The room was spacious with a comfortable bed. The pillows were really fluffy and everything you would expect with a stay at the Bellagio. The bathroom was spacious as well, with a separate bath and shower.
Review Cases
split
The room was spacious with a comfortable bed.
The pillows were really fluffy and everything you would expect with a stay at the Bellagio.
The bathroom was spacious as well, with a separate bath and shower.
Sentences
generate
{room,bed}
{pillows,stay, Bellagio}
{bathroom, bath, shower}
Sentence transaction
generate
{room,bed,pillows,stay, Bellagio,bathroom, bath, shower}
Review transaction

Fig. 2. Preparing Transactions for Association Rule Mining (ARM)

meaningful product topics, rather than on the looser vocabulary of user generated reviews. As part of our evaluation in Section 4 we will consider whether this topic variation in fact translates into any meaningful evaluation benefit.

3.3 Generating Recommendations

The RA generates a set of ranked recommendations by using association rule mining techniques to discover patterns of nouns/topics that recur frequently across many reviews. In the following section we will describe how these rules can then be applied to the current review text in order to produce a ranked set of noun/topic recommendations to be returned to the user via the RA plugin.

Rule Mining. At this point we have a set of transactions (whether *non-topic* or *topic*), which reflect frequent collections of nouns/topics that occur at the sentence-level or review-level. For example, in the digital camera domain we might have transactions such as $\{image, lens, resolution\}$ and $\{size, price\}$ extracted at the sentence-level to indicate that review sentences discussed camera resolution, lens type and image quality or camera size and price. We can apply association rule mining [1] to identify frequently occurring transactions and to generate a set of association rules of the form $\{image, lens\} \rightarrow \{resolution\}$. Following the standard algorithm for association rule mining, we first filter-out rules that fall below a minimum *support* level; that is we keep subsets of transactions that have a pre-defined frequency of occurrence as candidates for rules and their antecedents, so-called itemsets.

Ranking Recommendations. The resulting rules are ranked in descending order of their confidence, which is basically an estimate of the probability of finding the topic/noun that forms the rule consequent given the occurrence of the antecedent. To generate a set of ranked recommendations we apply each of the extracted rules, in order of confidence, to the current review text. If the current review text triggers a rule of the form $LHS \rightarrow RHS$ then the noun/topic that is the RHS is added to the recommendation list. This process terminates when a set of k recommendations have been generated.

3.4 Recommendation Feedback and Case Retention

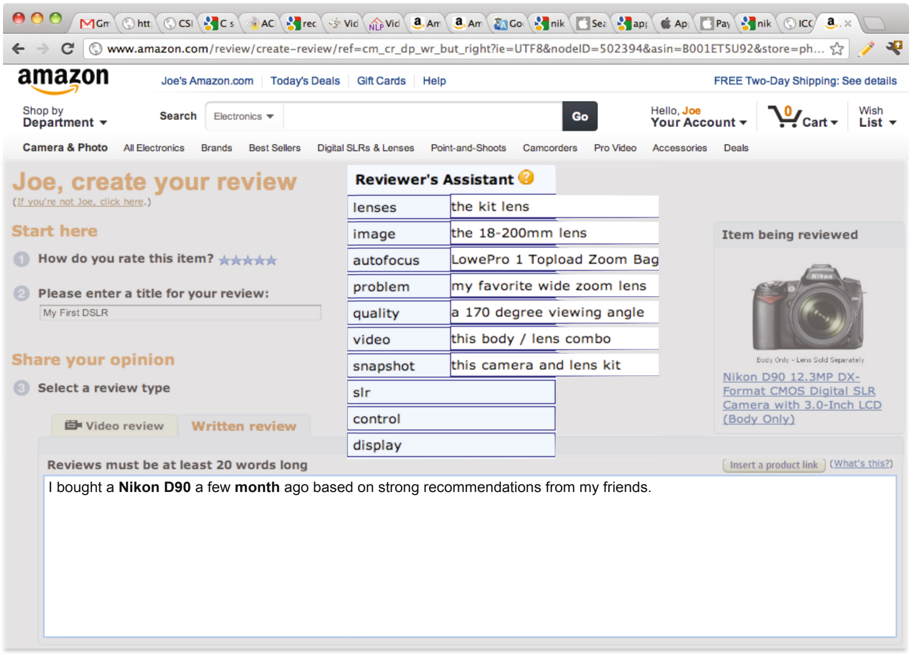
Each recommendation cycle refreshes the current set of suggestions for the user. Each of these recommendations can also be expanded to reveal the review fragments that formed the basis of the rule that led to the recommendation, thereby providing additional context for the user; for example, a recommendation for the topic *resolution* might show a review fragment such as *"... camera boasts an impressive 12 mega-pixel resolution..."*. By selecting a recommendation the user can directly add it to their review as a starting point for their own opinion.

There are a number of opportunities for learning to occur in the current system. Obviously, each completed review can be retained as a new case in the review case base. But in addition, it may also be possible to adapt the recommendation process by learning from direct user feedback; for example, as users select/ignore recommendations this can be used to reinforce/weaken future association rule patterns; this is left as a matter for future work.

3.5 The Reviewer's Assistant in Action

Figure 3 shows the Reviewer's Assistant in action for our user, Joe, who is reviewing a recently purchased Nikon D90 SLR camera on Amazon. Joe is presented with the usual Amazon review creation screen and the figure shows the Reviewer's Assistant overlay; the RA widget can be dragged to any suitable location on screen. The RA presents a dynamic set of updating review suggestions (in this case we show the topic-based version of the RA). Figure 3a shows some of the suggestions presented to Joe during the early stages of the review. In this case we see a number of suggestions for some common review topics for this product, including the *lens*, the *image* and *video* capability. As shown, Joe can view review fragments that relate to a particular topic by mousing-over the topic. For example, in this case the fragments *"the lens kit"*, *"the 18-200mm lens"* etc. are displayed for topic *"lens"*.

In Figure 3b we see a snapshot towards the end of the review writing. This time Joe is presented with additional topics, many of which are more specialised or not uniquely related to the specific product to provide the reviewer with an opportunity to broaden their review. Note also that as the reviewer writes their review, sentences that cover recommended topics are highlighted by emphasising the topic terms in the sentences. In Figure 3b we can see that the user's review

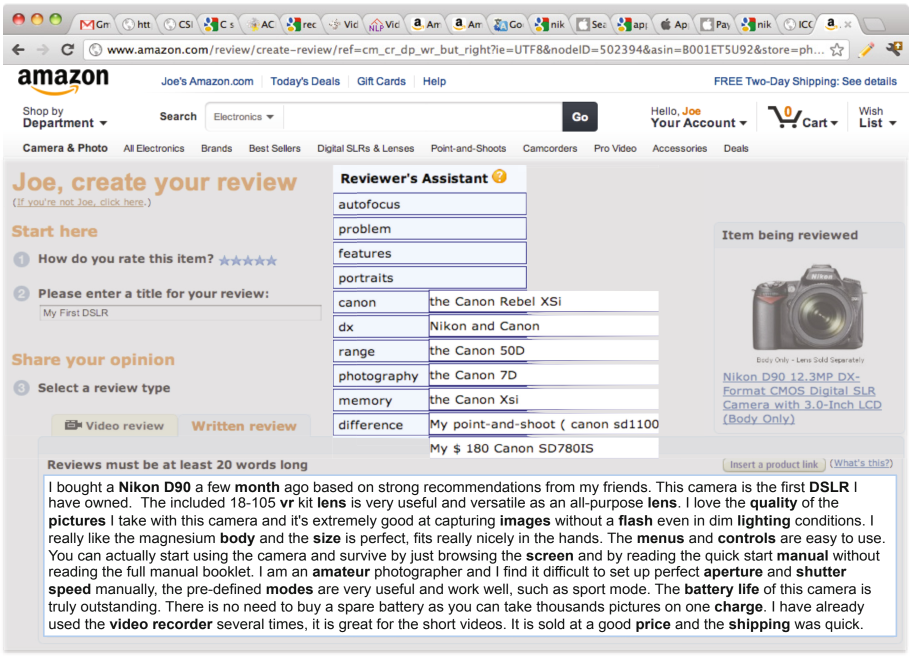

www.amazon.com/review/create-review/ref=cm_cr_dp_wr_but_right?ie=UTF8&nodeID=502394&asin=B001ET5U92&store=ph...
amazon
Joe's Amazon.com | Today's Deals | Gift Cards | Help
FREE Two-Day Shipping: See details
Shop by Department
Search Electronics Go
Hello, Joe Your Account
Cart
Wish List
Camera & Photo | All Electronics | Brands | Best Sellers | Digital SLRs & Lenses | Point-and-Shoots | Camcorders | Pro Video | Accessories | Deals
Joe, create your review
(If you're not Joe, click here.)
Start here
1 How do you rate this item?
2 Please enter a title for your review:
My First DSLR
Share your opinion
3 Select a review type
Video review
Written review
Reviews must be at least 20 words long
Insert a product link (What's this?)
Item being reviewed
Body Only - Lens Sold Separately
Nikon D90 12.3MP DX-Format CMOS Digital SLR Camera with 3.0-Inch LCD (Body Only)

Reviewer's Assistant

lenses	the kit lens
image	the 18-200mm lens
autofocus	LowePro 1 Topload Zoom Bag
problem	my favorite wide zoom lens
quality	a 170 degree viewing angle
video	this body / lens combo
snapshot	this camera and lens kit
slr	
control	
display	

I bought a **Nikon D90** a few **month** ago based on strong recommendations from my friends.

(a) Suggestions made at the beginning of the review writing process


www.amazon.com/review/create-review/ref=cm_cr_dp_wr_but_right?ie=UTF8&nodeID=502394&asin=B001ET5U92&store=ph...
amazon
Joe's Amazon.com | Today's Deals | Gift Cards | Help
FREE Two-Day Shipping: See details
Shop by Department
Search Electronics Go
Hello, Joe Your Account
Cart
Wish List
Camera & Photo | All Electronics | Brands | Best Sellers | Digital SLRs & Lenses | Point-and-Shoots | Camcorders | Pro Video | Accessories | Deals
Joe, create your review
(If you're not Joe, click here.)
Start here
1 How do you rate this item?
2 Please enter a title for your review:
My First DSLR
Share your opinion
3 Select a review type
Video review
Written review
Reviews must be at least 20 words long
Insert a product link (What's this?)
Item being reviewed
Body Only - Lens Sold Separately
Nikon D90 12.3MP DX-Format CMOS Digital SLR Camera with 3.0-Inch LCD (Body Only)

Reviewer's Assistant

autofocus	
problem	
features	
portraits	
canon	the Canon Rebel XSi
dx	Nikon and Canon
range	the Canon 50D
photography	the Canon 7D
memory	the Canon Xsi
difference	My point-and-shoot (canon sd1100
	My $ 180 Canon SD780IS

I bought a **Nikon D90** a few **month** ago based on strong recommendations from my friends. This camera is the first **DSLR** I have owned. The included 18-105 **vr** kit **lens** is very useful and versatile as an all-purpose **lens**. I love the **quality** of the **pictures** I take with this camera and it's extremely good at capturing **images** without a **flash** even in dim **lighting** conditions. I really like the magnesium **body** and the **size** is perfect, fits really nicely in the hands. The **menus** and **controls** are easy to use. You can actually start using the camera and survive by just browsing the **screen** and by reading the quick start **manual** without reading the full manual booklet. I am an **amateur** photographer and I find it difficult to set up perfect **aperture** and **shutter speed** manually, the pre-defined **modes** are very useful and work well, such as sport mode. The **battery life** of this camera is truly outstanding. There is no need to buy a spare battery as you can take thousands pictures on one **charge**. I have already used the **video recorder** several times, it is great for the short videos. It is sold at a good **price** and the **shipping** was quick.

(b) Suggestions made toward the end of the review

Fig. 3. The Reviewer's Assistant in action on Amazon

covers a range of topics that have been suggested, including *lens, aperture, shutter, battery life,* etc.

4 Live-User Evaluation

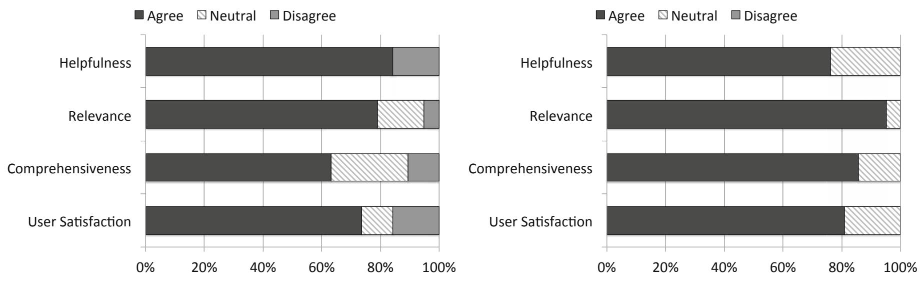
The Reviewer's Assistant, as described above, is designed to support users as they create user-generated reviews on sites like Amazon, TripAdvisor, and iTunes, by suggesting hints to users about the type of product features that they may wish to include in their reviews. We have presented two variations with respect to the type of recommendations made, one based on actual review fragments (*non-topic*) and one in which these fragments are first mapped to a set of well-defined topics, which are then recommended (*topic*). The following evaluation has three separate parts and in each, we pay particular attention to performance differences between the *non-topic* and *topic* variations of the RA, if any. In the first part, we describe the results of a live-user study focusing on how participants used the RA plugin and their feedback with respect to the utility of the recommendations and their overall satisfaction with the experience. In the second part of the analysis, we perform an objective analysis of the resulting reviews considering the depth and breadth of coverage offered by these reviews with respect to important product features. Finally, in part three, we perform a comparison of a subset of the above reviews and a set of comparable Amazon reviews, using a set of "expert" users to rate the helpfulness of these different reviews in order to better understand if the use of the RA leads to any improvements in review quality.

4.1 Usage Analysis

For the first part of this experiment we recruited 40 test users, 26 male and 14 female. 11 of the 40 participants had written at least one online review in the past and the majority had purchased products online through stores like Amazon and iTunes. We restricted our target product domain to that of digital cameras on Amazon and configured the RA plugins (*non-topic* and *topic*) accordingly; we chose this product domain because all users had at least some experience with this type of product.

The participants were randomly divided into *non-topic* and *topic* groups; exactly 21 participants (52.5%) had access to the *non-topic* version of the RA whilst the remaining 19 (47.5%) used the *topic* version. Each user was asked to select a product of interest and to write a review for this product; they were provided with a brief initial tutorial on the RA, the purpose of its suggestions, and how they might avail of them if appropriate.

During the trial user actions were logged as they completed their reviews and availed of the RA suggestions. At the end of the trial each user completed a short post-trial questionnaire in order to rate the RA under four key areas: 1) helpfulness – were the RA suggestions generally helpful? 2) relevance – were the suggestions relevant in the context of the review being written? 3) comprehensiveness – did the suggestions broadly cover the product being reviewed? 4)



(a) Topic suggestions (b) Non-topic suggestions

Fig. 4. User Feedback

overall satisfaction – was the participant satisfied with the overall experience provided by the RA?

The results of this questionnaire are shown in Figure 4 and are largely positive with respect to both the *topic* and and *non-topic* variations. For example, we can see that overall about 75% – 85% of users found the RA to be helpful, with the higher percentage pertaining to the *topic* variation. Interestingly, this advantage for *topic* is reversed when we look how relevant users found the recommendations to be. In this case we can see that, while 90% of the *non-topic* users rated the recommendations to be relevant, only 80% of the *topic* users rated their recommendations as relevant. Similarly, the feedback in terms of recommendation comprehensiveness also favours the *non-topic* variation, with scores of 85% versus 65% for *non-topic* and *topic* users, respectively.

The reason for this seems to be related to differences in how recommendations are managed after a user has covered a particular feature in their review. For example, in the case of the *non-topic* version, once a user writes about a specific feature, say *auto-focus*, this exact feature will be removed from the recommendation list and will not be suggested again in the future, but related features such as *zoom* or *focal-length* can be suggested. The same is true for the *topic* variation of RA, except that by removing the general topic, in this case *lens*, instead of the exact term *auto-focus*, lens-related features will not be suggested any more and *lens* will be replaced by another topic instead. In retrospect it appears that this particular recommendation filtering approach may have been overly restrictive and we will consider alternatives as a matter of future work.

Finally, in relation to the post-trial questionnaire we can see that overall there is strong user-support for the RA. Between 75% (*topic*) and 80% (*non-topic*) of users indicated that they were satisfied overall with the system.

4.2 Topic Coverage

We now consider the type of reviews that are produced. For example, is there any evidence that the *topic* and *non-topic* variations lead to quantitative differences in review quality? In this part of the evaluation we consider review quality in terms of the breadth and depth of topical coverage. In other words we can evaluate reviews based on the number of unique topics that they contain (*breadth*) and the average length of sentences on a given topic (*depth*). We can also measure the *redundancy* of a review as the average length of sentences that do not refer to well-defined topics.

More formally, the *breadth* of a review r with respect to topic set T is defined as the number of topics covered by that review, see Equation 1.

$$Breadth(r, T) = \mid \{t \in T \mid \exists s \in r : Cover(s, t)\} \mid \tag{1}$$

$Cover(s, t)$ is *true*, if topic t *is covered by* sentence s. A sentence *covers* a topic, if at least one synonym for (or member of) the topic is contained in it. In order to give proper semantics to the mathematical notation of $\in$ we assume that sentences are represented by a collection of all words, and reviews are represented by a collection of sentences. Note, one sentence might cover more than one topic.

The *Depth* of a review r with respect to topic set T is the average number of words to describe each topic covered by the review r, see Equation 2.

$$Depth(r, T) = \frac{\sum_{\{s \in r \mid \exists t \in T : Cover(s,t)\}} Length(s)}{Breadth(r, T)} \tag{2}$$

where the number of words in a sentence s is denoted by $Length(s)$.

Finally, the *Redundancy* of a review r with respect to topic set T is defined as the total length of sentences that do not cover any topic.

$$Redundancy(r, T) = \sum_{\{s \in r \mid \neg \exists t \in T : Cover(s,t)\}} Length(s) \tag{3}$$

Table 1. Breadth, Depth, and Redundancy; * indicates significance at 0.05

	topic	non-topic
Average Breadth*	10.42	7.62
Average Depth	10.69	10.53
Average Redundancy	9.68	10.24
Average Length	113.58	90.43

Thus we analyse the review texts of the 40 reviews produced during the above trial and compute their breadth, depth, and redundancy characteristics with reference to the defined set of product topics used for the digital camera domain. The results are presented in Table 1. We can see that while both techniques perform similarly in terms of review depth (10.69 for *topic* versus 10.53 for *non-topic*), the reviews produced with the *topic* version of RA tend to offer significantly broader coverage (a breadth of 10.42 for *topic* versus only 7.62 for *non-topic*) with less redundancy (9.68 for *topic* versus 10.24 for *non-topic*).

Of course this approach provides only a superficial analysis of review quality and it is not clear whether these depth, breadth and redundancy characteristics have any significant bearing on the ultimate perception of review quality or helpfulness. Thus, we analysis performance along these dimensions in the next section.

4.3 Review Quality

Ultimately the best test of the RA approach is to consider the quality of the resulting reviews in order to understand whether users find them to be helpful, for example. Even better is if we can compare our test reviews to a benchmark in terms of quality. This is the aim of this final evaluation section.

We collected 2 sets of reviews with similar lengths. The first set were chosen at random from the reviews written by participants of the RA trial above. We collected 10 random reviews written using the help of the RA with *topic* and another 10 written using the help of the RA with *non-topic*. For our second set, we selected two groups of Amazon reviews to serve as a benchmark, against which to judge the quality of the RA reviews. One group was chosen at random from among the most helpful Amazon camera reviews. We picked 10 reviews (*Amazon+*) that had a helpfulness score of at least 0.7 (meaning 70% of raters considered them helpful); in fact, the average helpfulness score for these reviews was 0.9 and thus we can view these as examples of very high quality product reviews written without the aid of RA. Next we chose another group of 10 random Amazon reviews (*Amazon-*), but this time we picked reviews that had a helpfulness score of less than 0.7; the average helpfulness score for reviews in this group was 0.41 and thus represent examples of lower quality reviews written without the help of RA.

Next, we recruited 15 reviewer "experts" (with a good understanding of the digital camera space) and asked them to perform a blind review of a random sample of reviews from the four sets above (*topic*, *non-topic*, *Amazon+*, and *Amazon-*). In each case we asked the experts to rate the reviews on a 5-point scale in terms of 1) helpfulness – how helpful did they think the review would be to others? 2) completeness – did the review provide a reasonably complete account of the product in question? 3) readability – was the review well written and readable? In total each test review was reviewed by 3 different experts. Finally, we calculated the average helpfulness, completeness, and readability ratings across each of the 4 review groups and also calculated their average breadth, depth and redundancy scores based on the approach taken previously.

The results are presented in Table 2 and show a significant positive benefit accruing to the RA in a number of important respects. For instance the average helpfulness rating of RA reviews (3.90 for both RA versions) is greater than the helpfulness rating for *Amazon+* (3.33) and *Amazon-* (3.07). Similarly, we can see clear benefits for the RA variations in terms of review completeness (3.67 and 3.57), when compared to *Amazon+* (2.67) and *Amazon-* (2.53). Both of these helpfulness and completeness benefits (RA versus Amazon) are statistically

Table 2. User Evaluation; * indicates significant difference between topic/ non-topic and Amazon+/ Amazon-; ** indicates significant difference between topic only and Amazon+/ Amazon-

	topic	non-topic	Amazon+	Amazon-
Helpfulness*	3.90	3.90	3.33	3.07
Completeness*	3.67	3.57	2.67	2.53
Readability	3.60	3.60	3.80	3.33
Average Breadth**	11.30	8.60	5.90	7.20
Average Depth	11.90	11.21	15.57	12.49
Average Redundancy**	5.50	14.30	22.40	21.00

significant at the 0.05 level; statistically significant differences were not found in terms of review readability.

Table 2 also shows how the *topic* version of the RA leads to reviews that have a greater topical breadth and reduced redundancy compared to *non-topic*, *Amazon+*, and *Amazon-*; once again these differences are statistically significant at the 0.05 level. For example, on average, the *topic* reviews cover 11.3 topics per review compared to only 5.9 and 7.3 topics per review for *Amazon+* and *Amazon-*. In addition, on average, the *topic* reviews contain very little redundancy (5.50 words per review) compared to much higher redundancy rates for *Amazon+* (22.4) and *Amazon-* (21) and even *non-topic* (14.3). It is worth noting that, though we did not find statistically significant differences, both groups of Amazon reviews seem to enjoy improved topical depth when compared to the RA groups.

4.4 Discussion

The above results tell an interesting and compelling story. While there may be little to choose between the *topic* and *non-topic* variations of the RA at present, it is clear that users found the RA system to be helpful and useful when it comes to supporting the review writing process, Moreover, the resulting reviews were rated more highly that their Amazon counterparts in a blind study of review quality.

As always there are limitations to be considered when evaluating the significance of these results. For a start the review domain was limited to digital cameras and the user studies were limited to focus groups of 40 users. Nevertheless the key results were found to be statistically significant rather than chance occurrences. Moreover, we have found no reason to suspect that by focusing on digital cameras we have in any way biased or skewed the evaluation. Certainly digital cameras are a popular class of online product sales and attract a critical mass of user reviews. Moreover they share similar characteristics (feature-based descriptions, feature tradeoffs, leading to complex purchasing decisions) with other classes of products and services (travel, gadgets, etc.). Given the above we feel confident that these results bode well for the promise of the RA and as such provide a clear demonstration of the value of a case-based approach to harness online experiences.

5 Conclusions

This work was inspired by early work on the GhostWriter systems [3,9], which highlighted the potential for web experiences and case-based reasoning to support users when creating user-generated content, whether in the form of adverts or product reviews. The main contribution of the work presented in this paper is twofold. Firstly, we have extended the original Ghostwriter approach by incorporating a combination of association rule mining and topic extraction to generate review recommendations that are more likely to match key product features as important review targets. Secondly, we have presented a comprehensive evaluation of the RA system, focusing on the overall user experience and a benchmarked study of real review quality. The results show that users find the RA to be useful and the resulting reviews are rated more highly than comparable Amazon reviews, even when compared against a set of best quality reviews.

In terms of future work there are a number of important possibilities. First of all, our current approach to topic extraction is very simply, modeling topics based on simple synonym sets, for example. Our next steps include exploring the use of automatic topic-detection and extraction techniques which will allow for a more sophisticated topic modeling approach. In addition, the RA currently does not consider the sentiment of review fragments during recommendation; for example, a given reviewer might speak positively about a particular product feature whilst another reviewer may speak negatively. This type of information can be useful when selecting recommendations, for example, by guiding users to review more controversial features of the product. Finally, the RA currently focuses on past review experiences that match the target product being reviewed. This potentially limits the scope of experiences that can influence recommendations and it is worth considering whether drawing on reviews from similar products is likely to be of benefit. For example, when reviewing a specific compact camera by Nikon it might be worth focusing on other compacts by Nikon or other manufacturers. We will consider this in future work by relaxing the similarity metric that is used during review retrieval.

Acknowledgments. This work is supported by Science Foundation Ireland under grant 07/CE/I1147.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. ACM SIGMOD Record 22, 207–216 (1993)
2. Aha, D.W., Breslow, L.A., Muñoz-Avila, H.: Conversational case-based reasoning. Applied Intelligence 14(1), 9–32 (2001)

3. Bridge, D., Healy, P.: The GhostWriter-2.0 case-based reasoning system for making content suggestions to the authors of product reviews. Knowledge-Based Systems 29, 93–103 (2012)
4. Bridge, D., Waugh, A.: Using experience on the read/write web: The ghostwriter system. In: Bridge, D., Plaza, E., Wiratunga, N. (eds.) Procs. of WebCBR: The Workshop on Reasoning from Experiences on the Web (Workshop Programme of the 8th International Conference on Case-Based Reasoning), pp. 15–24 (2009)
5. Chevalier, J.A., Dina Mayzlin, D.: The effect of word of mouth on sales: Online book reviews. Journal of Marketing Research 43(3), 345–354 (2006)
6. Dellarocas, C., Zhang, M., Awad, N.F.: Exploring the value of online product reviews in forecasting sales: The case of motion pictures. Journal of Interactive Marketing 21(4), 23–45 (2007)
7. Dhar, V., Chang, E.A.: Does chatter matter? the impact of user-generated content on music sales. Journal of Interactive Marketing 23(4), 300–307 (2009)
8. Dong, R., McCarthy, K., O'Mahony, M.P., Schaal, M., Smyth, B.: Towards an intelligent reviewer's assistant: Recommending topics to help users to write better product reviews. In: Proceedings of 17th International Conference on Intelligent User Interfaces (IUI 2012), Lisbon, Portugal, February 14-17, pp. 159–168 (2012)
9. Healy, P., Bridge, D.: The GhostWriter-2.0 System: Creating a Virtuous Circle in Web 2.0 Product Reviewing. In: Bridge, D., Delany, S.J., Plaza, E., Smyth, B., Wiratunga, N. (eds.) Procs. of WebCBR: The Workshop on Reasoning from Experiences on the Web (Workshop Programme of the Eighteenth International Conference on Case-Based Reasoning), pp. 121–130 (2010)
10. Hu, N., Liu, L., Zhang, J.: Do online reviews affect product sales? the role of reviewer characteristics and temporal effects. Information Technology and Management 9, 201–214 (2008)
11. Kim, S.-M., Pantel, P., Chklovski, T., Pennacchiotti, M.: Automatically assessing review helpfulness. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), Sydney, Australia, July 22-23, pp. 423–430 (2006)
12. Liu, Y., Huang, X., An, A., Yu, X.: Modeling and predicting the helpfulness of online reviews. In: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM 2008), Pisa, Italy, December 15-19, pp. 443–452 (2008)
13. O'Mahony, M.P., Smyth, B.: Learning to recommend helpful hotel reviews. In: Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys 2009), New York, NY, USA, October 22-25, pp. 305–308 (2009)
14. Wu, G., Greene, D., Cunningham, P.: Merging multiple criteria to identify suspicious reviews. In: Proc. 4th ACM Conference on Recommender Systems (RecSys 2010), pp. 241–244 (2010)
15. Wu, G., Greene, D., Smyth, B., Cunningham, P.: Distortion as a validation criterion in the identification of suspicious reviews. In: Proceedings of the First Workshop on Social Media Analytics (SOMA 2010), New York, pp. 10–13 (2010)
16. Zhu, F., Zhang, X.M.: Impact of online consumer reviews on sales: The moderating role of product and consumer characteristics. Journal of Marketing 74(2), 133–148 (2010)

Adapting Spatial and Temporal Cases

Valmi Dufour-Lussier[1,2,3], Florence Le Ber[1,2,4],
Jean Lieber[1,2,3], and Laura Martin[5]

[1] Université de Lorraine, LORIA, UMR 7503 — 54506 Vandœuvre-lès-Nancy, France
[2] CNRS — 54506 Vandœuvre-lès-Nancy, France
[3] Inria — 54602 Villers-lès-Nancy, France
[4] ENGEES, LHYGES, UMR 7517 — 67000 Strasbourg, France
[5] INRA, UR055 ASTER–Mirecourt — 88500 Mirecourt, France
{valmi.dufour-lussier,florence.leber,jean.lieber}@loria.fr,
laura.martin@mirecourt.inra.fr

Abstract. Qualitative algebras form a family of languages mainly used to represent knowledge depending on space or time. This paper proposes an approach to adapt cases represented in such an algebra. A spatial example in agronomy and a temporal example in cooking are given. The idea behind this adaptation approach is to apply a substitution and then repair potential inconsistencies, thanks to belief revision on qualitative algebras.

Keywords: adaptation, case-based reasoning, qualitative algebra, spatial reasoning, temporal reasoning.

1 Introduction

Qualitative spatial and temporal reasoning (QSTR) as a research domain has been active since the beginning of the 1980s. The paradigm has been exploited to help solve planning and constraint satisfaction problems, but rarely within case-based reasoning (CBR). Nevertheless, many domains in which QSTR is used could be addressed with CBR because the knowledge involved is usually contextual and incompletely formalised. This is the case in the domain of landscape agronomy, in which knowledge is acquired from farm surveys and from schematic descriptions of the spatial organisation of farming territories. Another example is the cooking domain, in which some knowledge is of a temporal nature.

This paper proposes an approach for the adaptation of spatial and temporal cases, which is based on a process integrating substitution and revision-based adaptation. One may, for instance, want to replace a plot of maize with a plot of a different crop in a farm, or to replace mushrooms with carrots in a recipe. Mere substitution is insufficient, because not all crops, nor ingredients, are used in the same way. It is therefore supplemented with a belief revision process through which the spatial representation of the farm or the temporal representation of the recipe are changed to be consistent with domain knowledge.

Section 2 justifies the paper's approach by introducing two examples from the farming and cooking domains. Section 3 then introduces the formal notions required for the approach, namely in terms of CBR, revision-based adaptation, and QSTR. The approach is then defined in details in section 4, and an algorithm is described in section 5. Section 6 illustrates those formal notions and the results of the algorithm using the two examples introduced in section 2. Related work is discussed in section 7. Section 8 concludes and presents some future work.

2 Introduction of the Running Examples

This section describes two examples in which a mere substitution would not yield an acceptable adaptation of a retrieved case with respect to the target problem. In the first example, the spatial adaptation of a farm must take into account agronomic knowledge about the location of fields with respect to the environment and cropping constraints. In the second example, the temporal adaptation of a risotto recipe must take into account cooking knowledge about the length required to cook different vegetables.

2.1 A Spatial, Agronomic Example

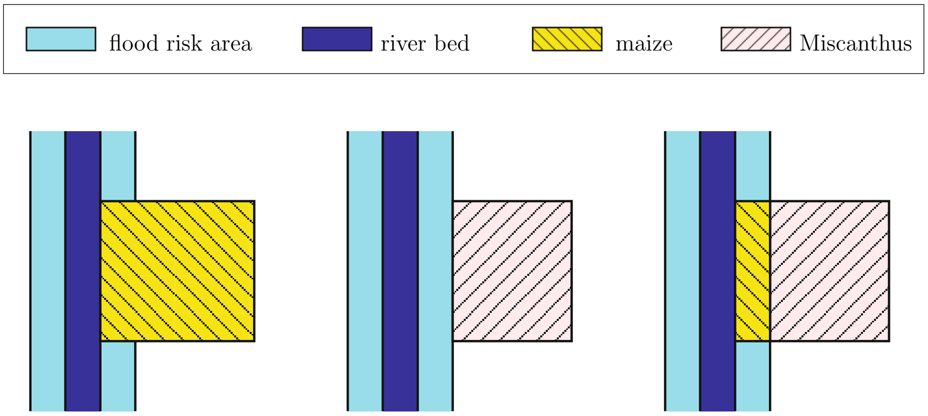
Spatial adaptation is illustrated using the example of Miscanthus allocation practices in agriculture. Miscanthus is a perennial grass currently promoted as a renewable source of energy in Europe to produce high yield of biomass with low input [1]. Its potential to reduce greenhouse gas emission is dependent to its spatial allocation into farmlands [2], therefore modelling spatial land use changes into farmlands is of great interest.

CBR can be used to model Miscanthus spatial allocation. The problem is the a crop production requirements and the farm description, and the solution is a crops spatial allocations. A farm description is defined by a cropping plan (the crop proportions/allocations into farmland) and by the spatial farmland features (e.g. the spatial relations of plots with buildings, woodland and rivers).

In this example, illustrated by figure 1, a farmer who wants to cultivate Miscanthus is considered. A case corresponding to a maize farm could be retrieved (figure 1a), based on real cases surveyed in France [3] and expert knowledge which identify similarities in Miscanthus and maize allocation requirements regarding temperature and soil moisture [4].

Replacing maize with Miscanthus (which is usually harvested from February to March in France) comes with a spatial constraint in the agronomic domain knowledge. Because access to plots by harvesting machinery is impaired by excess soil water in winter, Miscanthus must not be allocated near a river, in a flood-risk area, whereas maize can be planted up to a legal 5 metres from rivers.

Therefore, it is expected that the adaptation process would not only replace maize by Miscanthus, but also reduce the size of the plot so that it doesn't overlap with the flood plain of any nearby river (figure 1b). A better adaptation



(a) Source case: a maize farm

(b) Adapted case: a Miscanthus farm

(c) Another adapted case: a maize *and* Miscanthus farm

Fig. 1. Crop spatial allocation example

may break the plot into two parts and retain maize cultivation in the flood plain to ensure that this part of the farmland remains productive (figure 1c).

2.2 A Temporal, Cooking Example

Temporal adaptation is illustrated through recipe adaptation, in the spirit of Taable [5], a CBR application for cooking. If a user asks for a carrot risotto recipe and none is found in the case base, Taable may retrieve a mushroom risotto recipe and suggest reusing it with mushrooms replaced with carrots.

Say the mushrooms are added to the rice 2 minutes before the end, and the cooking domain knowledge indicates that carrots must be cooked for 25 minutes in order to be done, whereas the rice must be cooked for 18 minutes. A proper adaptation would require not only the lengthening of the cooking time of the vegetables, but also a reordering of the actions in the recipe.

3 Background

3.1 Case Adaptation

In this paper, `Source`, `Target` and `DK` respectively denote the case to be adapted, the target case and the domain knowledge. `Source` and `Target` are required to be consistent with `DK`.[1] Given `Source` and `Target`, the adaptation aims at building

[1] If a case `Source` from the case base is inconsistent with the domain knowledge, a consistency maintenance process should be triggered to restore this consistency. If a query to the CBR system is inconsistent with the domain knowledge, it either should be rejected (e.g. "I want a recipe with apples but without fruit" is rejected by Taable since `DK` entails that apples are fruits), or `DK` should be modified.

a new case, `AdaptedCase`. This case is built by adding some information to the target case (intuitively, `Target` specifies only the "problem part" of the query), and it has to be consistent with `DK`.

It is assumed that a matching step precedes the adaptation process, providing links between `Source` and `Target`. It is represented by a substitution σ, mapping descriptors of `Source` to descriptors of `Target`. As an example, in the system Taaable, matching is performed during retrieval [5]. This process, applied to the cooking example of the previous section, would have returned $\sigma = \texttt{mushroom} \rightsquigarrow \texttt{carrot}$. In the following, this preprocessing step of adaptation is considered to be given and, thus, σ is an input of the adaptation process described in section 4.

3.2 Belief Revision and Revision-Based Adaptation

Belief Revision. In a given representation formalism, a revision operator $\dot{+}$ maps two knowledge bases ψ and μ to knowledge base $\psi \dot{+} \mu$, the revision of ψ by μ. Intuitively, $\psi \dot{+} \mu$ is obtained by making a minimal change of ψ into ψ', so that the conjunction of ψ' and μ, $\psi' \wedge \mu$, is consistent. Then, $\psi \dot{+} \mu$ is this conjunction. The notion of minimal change can be modelled in various ways, so there are various revision operators. However, postulates have been proposed for such an operator, such as the AGM postulates [6]. These postulates have been applied to propositional logic [7] and well studied in this formalism. Given a distance `dist` on the set $\mathcal{U}$ of the interpretations, an operator $\dot{+}_{\texttt{dist}}$ can be uniquely defined (up to logical equivalence) as: the set of models of $\psi \dot{+}_{\texttt{dist}} \mu$ is the set of models of μ that have a minimal distance to the set of models of ψ.

Revision-Based Adaptation. Given a revision operator $\dot{+}$, $\dot{+}$-adaptation consists simply in using this revision operator to perform adaptation, taking into account the domain knowledge:

$$\texttt{AdaptedCase} = (\texttt{DK} \wedge \texttt{Source}) \dot{+} (\texttt{DK} \wedge \texttt{Target}) \tag{1}$$

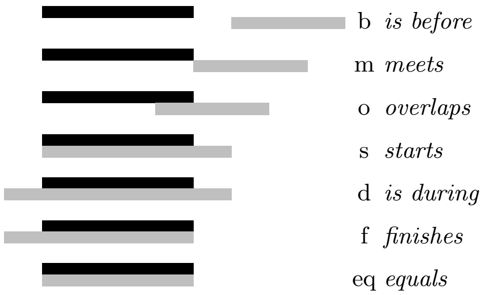
The intuition behind revision-based adaptation is to reduce adaptation to an inconsistency repair. It has been studied in propositional logic [8] and, more generally, in metric spaces [9]. It has been generalised to multiple case adaptation using integrity constraint belief merging in [10]. Its principle has also been applied to adaptation in an expressive description logic [11]. This paper applies it, with some differences, to qualitative constraint networks.

3.3 Qualitative Representation of Spatial and Temporal Knowledge

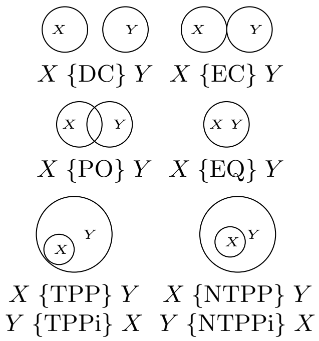
Definitions. A qualitative algebra is a relation algebra that defines a set $\mathfrak{B}$ of binary relations applicable between two variables, usually representing points, intervals or regions. Allen interval algebra [12], for instance, introduces 13 basic relations between intervals, corresponding to the 13 possible arrangements of their lower and upper bounds. 7 relations are illustrated in figure 2a. The 6 others are the inverse of the first 6 (eq is symmetric).

$\mathcal{INDU}$ [13] extends the set of Allen relations by combining them with relations over the interval durations. For 7 Allen relations, there is only one possible duration relation (e.g. i {d} j implies that the duration of i is shorter than the duration of j). For the other 6, all three duration relations $<$, $=$ and $>$ are possible. This yields a total of 25 basic relations. They are written as r^s, where r is an Allen relation and s is a duration relation.

Region connection calculi [14] are well-known spatial algebras. The most usual, RCC8, introduces 8 relations between regions, as shown in figure 2b.



(a) Allen interval algebra basic relations



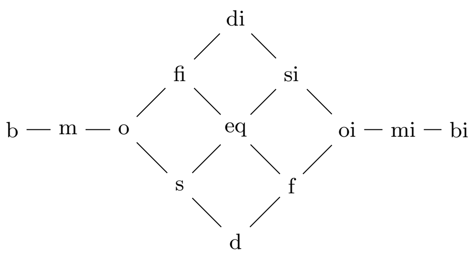
(b) RCC8 basic relations

Fig. 2. Two common qualitative algebras

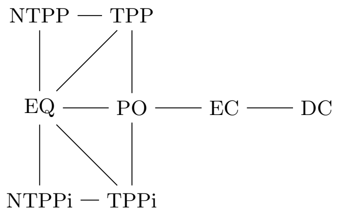
Qualitative knowledge can be represented as qualitative constraint networks (QCNs). A QCN is a pair (V, C), where V is a set of variables, and C is a set of constraints of the form $V_i\ C_{ij}\ V_j$ with $V_i, V_j \in V$, and C_{ij} a set of the basic relations defined by the algebra (C_{ij} is a relation that is a disjunction of the basic relations, i.e. $i\ \{r_1, r_2\}\ j$ means that i is related to j with either r_1 or r_2). In $\mathcal{INDU}$, shortcut notations $r^?$ and $?^s$ respectively represent the Cartesian product of r and all possible duration relations and the product of s and all possible Allen relations (e.g., $\{\mathrm{m}\}^? = \{\mathrm{m}^<, \mathrm{m}^=, \mathrm{m}^>\}$; $\{\mathrm{d}\}^? = \{\mathrm{d}^<\}$; $\{?\}^= = \{\mathrm{b}^=, \mathrm{m}^=, \mathrm{o}^=, \mathrm{eq}^=, \mathrm{oi}^=, \mathrm{mi}^=, \mathrm{bi}^=\}$.

A *scenario* is a QCN $\mathcal{S} = (V_\mathcal{S}, C_\mathcal{S})$ such that for each $V_i, V_j \in V_\mathcal{S}$, there exists one constraint $V_i\ \{r\}\ V_j \in C_\mathcal{S}$. $\mathcal{S}$ satisfies the QCN $\mathcal{N} = (V_\mathcal{N}, C_\mathcal{N})$ if $\mathcal{S}$ and $\mathcal{N}$ have the same set of variables and each constraint relation in $\mathcal{S}$ is a subset of the corresponding constraint relation in $\mathcal{N}$. A scenario is consistent if a valuation can be provided for the variables such that all constraints are observed, and a QCN is consistent if it has a consistent scenario. Two QCNs are said to be equivalent if every scenario of the former is a scenario of the latter and vice-versa.

Revision of QCNs. A QCN is a knowledge base and thus, the issue of revising a QCN ψ by a QCN μ can be addressed. In [16], such a revision operator is



(a) For Allen algebra [15]



(b) For RCC8 [14]

Fig. 3. The relation neighbourhood graphs of two common qualitative algebras

defined,[2] following the idea of an operator $\dot{+}_{\texttt{dist}}$ (cf. section 3.2), where an interpretation is a scenario, a model of a QCN is a scenario that satisfies it, and a distance `dist` between scenarios/interpretations is defined as follows.

First, a distance d between basic relations of the considered algebra is defined. Formally, a *neighbourhood graph* whose vertices are the relations of the algebra is given, and $d(r, s)$ is the distance between r and s in the graph. It represents closeness between relations. For instance, b and m are close ($d(\mathrm{b}, \mathrm{m}) = 1$) since they express similar conditions on the boundaries of the intervals (for the lower bounds: = for both; for the upper bounds: < for b and = for m). Figure 3 presents such graphs, respectively for Allen algebra and RCC8. d makes it possible to define `dist`, a distance between two scenarios $\mathcal{S} = (V, C_{\mathcal{S}})$ and $\mathcal{T} = (V, C_{\mathcal{T}})$ based on the same set of variables V, as:

$$\texttt{dist}(\mathcal{S}, \mathcal{T}) = \sum_{V_i, V_j \in V, i \neq j} d(r_{\mathcal{S}}(V_i, V_j), r_{\mathcal{T}}(V_i, V_j)) \tag{2}$$

where $r_{\mathcal{S}}(V_i, V_j)$ is the relation r such that $V_i \; \{r\} \; V_j \in C_{\mathcal{S}}$.

Given two QCNs ψ and μ, the revision of ψ by μ returns the set R of scenarios satisfying μ that are the closest ones to the set of scenarios satisfying ψ.[3]

4 Formalisation

4.1 Representation of the Adaptation Problem

Parametrised QCNs. It is assumed that the variables of the considered QCNs can be parametrised by elements of a given set $\mathcal{P}$. A parameter $p \in \mathcal{P}$ is either a *concrete parameter*, $p \in \mathcal{CP}$, or an *abstract parameter*, $p \in \mathcal{AP}$: $\mathcal{P} = \mathcal{CP} \cup \mathcal{AP}$,

[2] Technically, the authors of [16] define a merge operator taking a coercive QCN as a parameter. A revision operator can be defined using this merge operator.

[3] This slightly differs from the definition of revision given in section 3.2 where $\psi \dot{+} \mu$ is a knowledge base, not a set of models. This is due to the fact that there may be no QCN that has R as set of scenarios satisfying it.

$\mathcal{CP} \cap \mathcal{AP} = \emptyset$. A concrete parameter denotes a concept of the application domain, e.g. $\mathtt{mushroom} \in \mathcal{CP}$ for the cooking example. In this example, the formal interval $\mathtt{cooking}(\mathtt{mushroom})$ represents the temporal interval of the mushroom cooking. The domain knowledge $\mathtt{DK} = (V_{\mathtt{DK}}, C_{\mathtt{DK}})$ is a set of constraints, for example:

$$C_{\mathtt{DK}} = \left\{ \begin{array}{ll} \mathtt{cooking}(\mathtt{rice}) \overset{?}{=} \mathtt{18\_min} & \textit{(rice requires 18 min of cooking)} \\ \mathtt{cooking}(x)\ \{\mathrm{m}\}^? \ \mathtt{cooked}(x) & \textit{(when the action of cooking } x \textit{ is} \\ & \quad \textit{finished, } x \textit{ is cooked)} \\ \mathtt{18\_min} \overset{?}{<} \mathtt{25\_min} & \textit{(18 min is shorter than 25 min)} \end{array} \right\} \quad (3)$$

where $\mathtt{rice} \in \mathcal{CP}$ and $x \in \mathcal{AP}$. An abstract parameter must be understood with a universal quantification over the concrete parameters; e.g. $\mathtt{cooking}(x)\ \{\mathrm{m}\}^?\ \mathtt{cooked}(x)$ entails $\mathtt{cooking}(\mathtt{mushroom})\ \{\mathrm{m}\}^?\ \mathtt{cooked}(\mathtt{mushroom})$.

Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be two QCNs. $\mathcal{N}_1 \wedge \mathcal{N}_2$ is the QCN $\mathcal{N} = (V, C)$ such that $V = V_1 \cup V_2$ and C contains the constraints of C_1, the constraints of C_2, and the constraints that are deduced by instantiation of the abstract parameters by concrete parameters appearing in $\mathcal{N}_1$ and $\mathcal{N}_2$ (technically, this instanciation of $x \in \mathcal{AP}$ to $p \in \mathcal{CP}$ is a unification [17]). For example, if $\mathcal{N}_1 = C_{\mathtt{DK}}$ defined by equation (3) and $\mathcal{N}_2 = (\{\mathtt{cooking}(\mathtt{tomato}), \mathtt{cooked}(\mathtt{tomato})\}, \emptyset)$, then $\mathcal{N}_1 \wedge \mathcal{N}_2 = (V, C)$ with $C = C_{\mathtt{DK}} \cup \{\mathtt{cooking}(\mathtt{tomato})\ \{\mathrm{m}\}^?\ \mathtt{cooked}(\mathtt{tomato})\}$.

Substitutions. The *atomic substitution* $\sigma = p \rightsquigarrow q$, where $p, q \in \mathcal{P}$, is the function from $\mathcal{P}$ to $\mathcal{P}$ defined by $\sigma(a) = \begin{cases} q & \text{if } a = p \\ a & \text{otherwise} \end{cases}$. A *substitution* is a composition $\sigma_1 ; \ldots ; \sigma_n$ of atomic substitutions σ_i.[4]

Let $\sigma = p \rightsquigarrow q$ be an atomic substitution. σ is *concrete* if $p, q \in \mathcal{CP}$. σ is an *atomic abstraction* if $p \in \mathcal{CP}$ and $q \in \mathcal{AP}$. σ is an *atomic refinement* if $p \in \mathcal{AP}$ and $q \in \mathcal{CP}$. A *concrete substitution* (resp., an abstraction, a refinement) is a composition of concrete atomic substitutions (resp., of atomic abstractions, of atomic refinements). Any concrete substitution σ can be written $\sigma = \alpha ; \varrho$ where α is an abstraction and ϱ is a refinement, as the following equation illustrates:

$$\mathtt{mushroom} \rightsquigarrow \mathtt{carrot} = \mathtt{mushroom} \rightsquigarrow x \ ; \ x \rightsquigarrow \mathtt{carrot}$$

where $\mathtt{mushroom}, \mathtt{carrot} \in \mathcal{CP}$ and $x \in \mathcal{AP}$. This can be shown as follows. First, σ can be written $p_1 \rightsquigarrow q_1 ; \ldots ; p_n \rightsquigarrow q_n$ with $p_i, q_i \in \mathcal{CP}$ and $p_i \neq p_j$ if $i \neq j$.[5] Let $x_1, \ldots, x_n$ be n abstract parameters, let $\alpha_i = p_i \rightsquigarrow x_i$, let $\varrho_i = x_i \rightsquigarrow q_i$, let $\alpha = \alpha_1 ; \ldots ; \alpha_n$, and let $\varrho = \varrho_1 ; \ldots ; \varrho_n$. α is an abstraction, ϱ is a refinement and $\sigma = \alpha ; \varrho$.

Let σ be a substitution. σ is extended on qualitative variables by applying it to their parameters. For example, if $\sigma = \mathtt{mushroom} \rightsquigarrow \mathtt{carrot}$ then

[4] The composition of σ and σ', denoted by $\sigma ; \sigma'$, is the function that associates to $p \in \mathcal{P}$, $\sigma ; \sigma'\ (p) = \sigma'(\sigma(p)) \in \mathcal{P}$.

[5] Which can be shown inductively thanks to the following lemmas: (1) if $p \neq q$ then $p \rightsquigarrow q ; p \rightsquigarrow q' = p \rightsquigarrow q$, (2) $p \rightsquigarrow q ; q \rightsquigarrow r = p \rightsquigarrow r$, and (3) if $p \neq p'$, $q \neq p'$ and $q' \neq p$, then $p \rightsquigarrow q ; p' \rightsquigarrow q' = p' \rightsquigarrow q' ; p \rightsquigarrow q$.

$\sigma(\texttt{cooking}(\texttt{mushroom})) = \texttt{cooking}(\texttt{carrot})$. Then, σ is extended to a constraint $c = (V_i\ C_{ij}\ V_j)$ by $\sigma(c) = (\sigma(V_i)\ C_{ij}\ \sigma(V_j))$. Finally, σ is extended on a QCN by applying it to its variables and constraints: $\sigma((V, C)) = (\sigma(V), \sigma(C))$ where $\sigma(V) = \{\sigma(V_i) \mid V_i \in V\}$ and $\sigma(C) = \{\sigma(c) \mid c \in V\}$.

Adaptation Problem. An adaptation problem is given by a tuple $(\texttt{Source}, \texttt{Target}, \texttt{DK}, \sigma)$. Source and Target are the representations of the source and target cases by QCNs with concrete variables (i.e. not parametrised by any abstract parameter). DK is a QCN representing the domain knowledge. $\sigma = p_1 \rightsquigarrow q_1\ ;\ \ldots\ ;\ p_n \rightsquigarrow q_n$ is a concrete substitution such that each p_i (resp., q_i) parametrises a variable of Source (resp., Target). $\texttt{DK} \wedge \texttt{Source}$ and $\texttt{DK} \wedge \texttt{Target}$ are assumed to be consistent (cf. section 3.1). The goal of adaptation is to build a consistent QCN AdaptedCase that entails $\texttt{DK} \wedge \texttt{Target}$, whose qualitative variables are obtained by applying σ on the qualitative variables of Source, and that is obtained thanks to minimal modification of $\texttt{DK} \wedge \texttt{Source}$.

4.2 Principles of Revision-Based Adaptation of a QCN

A first idea to perform the adaptation, given a tuple $(\texttt{Source}, \texttt{Target}, \texttt{DK}, \sigma)$, is to apply σ on Source, thus obtaining a QCN $\texttt{DK} \wedge \sigma(\texttt{Source})$ that may be inconsistent, and then restoring consistency. Although this gives a good intuition of the revision-based adaptation of a QCN, it is not consistent with the irrelevance of syntax principle[6]. Indeed, any two inconsistent knowledge bases (e.g. two inconsistent QCNs) are equivalent: their sets of models are both empty. Thus, at a semantic level, repairing an inconsistent knowledge base is meaningless. By contrast, revision aims at modifying a *consistent* knowledge base with another *consistent* one, the conjunction of which may be inconsistent. This is one reason why we do not follow straightforwardly this first idea. Another reason is more practical: using revision of QCNs makes it possible to exploit the work of [16].

The revision-based adaptation consists first in decomposing σ in an abstraction α and a refinement ϱ: $\sigma = \alpha\ ;\ \varrho$ (cf. previous section). Then, α is applied to Source: a QCN $\texttt{DK} \wedge \alpha(\texttt{Source})$ is built that is necessarily consistent since $\texttt{DK} \wedge \texttt{Source}$ is consistent and every constraint of $\texttt{DK} \wedge \alpha(\texttt{Source})$ corresponds to a constraint of $\texttt{DK} \wedge \texttt{Source}$. In other words, $\texttt{DK} \wedge \texttt{Source}$ is consistent and is more or equally constrained as $\texttt{DK} \wedge \alpha(\texttt{Source})$, so $\texttt{DK} \wedge \alpha(\texttt{Source})$ is consistent.

The third step involves revision. The idea is to make a revision of ψ by μ where $\psi = \texttt{DK} \wedge \alpha(\texttt{Source})$ and $\mu = \texttt{DK} \wedge \texttt{Target} \wedge \mathcal{N}_\varrho$ where $\mathcal{N}_\varrho$ represents the following statement: "Each qualitative variable V_i of $\alpha(\texttt{Source})$ is constrained to be equal to its refinement $\varrho(V_i)$." For this purpose, the relation *eq* for equality is used:[7] $V_i\ eq\ \varrho(V_i)$. Therefore, $\mathcal{N}_\varrho = (V_\varrho, C_\varrho)$ where

$$V_\varrho = \alpha(V) \cup \sigma(V) \qquad\qquad C_\varrho = \{V_i\ eq\ \varrho(V_i) \mid V_i \in \alpha(V)\}$$

[6] This states that an inference remains valid when replacing formulas by logically equivalent formulas. This principle is usually observed in belief revision [7], i.e. if $\psi \equiv \psi'$ and $\mu \equiv \mu'$, then $\psi \dotplus \mu \equiv \psi' \dotplus \mu'$.

[7] *eq* is $\text{eq}^=$ for $\mathcal{INDU}$ and EQ for RCC8.

μ is consistent since $\texttt{DK} \wedge \texttt{Target}$ is and since each constraint $V_i\ eq\ \varrho(V_i)$ of $\mathcal{N}_\varrho$ either is a tautology (when V_i does not contain any abstract parameter refined by ϱ) or links a variable V_i that does not appear in $\texttt{DK} \wedge \texttt{Target}$ with $\varrho(V_i)$.

Then, $\psi \dotplus \mu$ gives a set of scenarios and `AdaptedCase` is chosen among them.

5 Algorithm and Implementation

Input and Output. The revision algorithm takes as input $\psi = \texttt{DK} \wedge \alpha(\texttt{Source})$, $\mu = \texttt{DK} \wedge \texttt{Target} \wedge C_\varrho$, as well as a relation neighbourhood graph and a transitivity table for the algebra used. The neighbourhood graph enables to define a distance d between relations and the transitivity table defines a relation composition function $\circ : \mathfrak{B} \times \mathfrak{B} \rightarrow 2^{\mathfrak{B}}$, for example, m $\circ$ mi = {eq, f, fi} in Allen algebra. The revision algorithm returns a set of scenarios of μ and their distance to ψ.

Algorithm. First, it is necessary to ensure that all variables in either QCN are present in the other QCN as well. All pairs of variables that have no relation associated to them are given the relation $\mathfrak{B}$–the unspecified relation.

The algorithm must then generate all the scenarios of μ and of ψ and measure their distance pair-wise. The amount of scenarios for a given QCN is of the order of $O\left(|\mathfrak{B}|^{\frac{|V|\cdot(|V|-1)}{2}}\right)$. For each pair of scenarios, the distance is calculated using equation (2). The distance between a scenario $\mathcal{S}$ of μ and ψ is the smallest distance between $\mathcal{S}$ and any scenario of ψ. The distance between the QCNs μ and ψ is the smallest distance between a scenario of μ and the QCN ψ. Once all scenarios of μ and of ψ have been compared pair-wise, the distance between μ and ψ is known and the scenarios of μ equal to this distance are returned.

Only consistent scenarios are to be considered. Path-consistency of a scenario[8] is a necessary and sufficient condition for consistency.[9] This condition is verified in time $O(|V|^3)$ if, for each $V_i, V_j, V_k \in V$, $r(V_i, V_k) \in r(V_i, V_j) \circ r(V_j, V_k)$. All inconsistent scenarios are discarded.

Optimisations. Because of the complexity, limiting the search space is essential. Considering that the minimum of sums is never less than the sum of minimums, a lower bound on the distance between two QCNs can be obtained in time $O(|V|^2 \cdot |\mathfrak{B}|^2)$ by computing the pair-wise minimal distance for each constraint and summing those. Empirically, it appears that the actual distance is usually closer to this lower bound than to the maximal bound, which is a function of the length of the longest path in the neighbourhood graph.

Therefore, it is often profitable to set an initial upper bound on the distance between ψ and μ which is equal to the lower bound, and search incrementally.

[8] A scenario is path-consistent if, in each 3-tuples (V_i, V_j, V_k) of variables, for each consistent valuation of V_i and V_j, there exists a consistent valuation of V_k. In RCQs, this is checked with transivity tables indicating the possible results of the composition of any two relations.

[9] The same is not true for QCNs in general, as shown in [12].

This search can be further optimised by computing the lower bound on the distance between a scenario $\mathcal{S}$ of μ and ψ, which can be done in time $O(|V|^2)$. This bound makes it possible to discard altogether certain scenarios of μ and thus avoid having to generate all the scenarios of ψ over again. Of course, whenever a scenario of μ is found to be at an acceptable distance to a scenario of ψ, incremental search means it is not necessary to examine other scenarios of ψ.

Another worthwhile optimisation in problems larger than a few variables is computing the algebraic closure of the QCNs, which is obtained by enforcing path-consistency. That is, for each $V_i, V_j, V_k \in V$, C_{ik} is replaced with $C'_{ik} = C_{ik} \cap (C_{ij} \circ C_{jk})$ where $\circ$ is extended on $2^{\mathfrak{B}} \times 2^{\mathfrak{B}} \rightarrow 2^{\mathfrak{B}}$ by $R \circ S = \bigcup_{r \in R, s \in S} r \circ s$. This is repeated until stability, i.e. no relation is changed after considering all 3-tuples of variables.

The optimisations proposed herein maintain the completeness and correctness of the algorithm, but they may prove insufficient to obtain a usable system. We think that approximation algorithms may give satisfactory results while running significantly faster. This will be the subject of future work.

6 Application on the Running Examples

This section revisits the examples from section 2. First, the agronomic example is taken in its simple form (where only Miscanthus is cultivated) to illustrate the algorithm. Then, the formalisation and the results are shown and discussed for the more complex form of the agronomic example and for the cooking example.

6.1 Simple Agronomic Problem

Consider a farm with one maize plot being adjacent to a river. To address the fact that there is a difference in possible agricultural uses between the bed of the river and the zone with flood risks, it is broken in two regions, `low_water_channel` and `flood_plain`, such that the former is a proper part of the latter and that their boundaries don't touch. This is expressed in RCC8 as `low_water_channel` {NTPP} `flood_plain`. The fact that a maize plot is adjacent to a river is represented as `plot(maize)` {EC} `low_water_channel`.

A farmer wishes to cultivate Miscanthus in a similar setting, prompting the retrieval of the farm case just described. A substitution must be applied: $\sigma = \texttt{maize} \rightsquigarrow \texttt{Miscanthus} = \texttt{maize} \rightsquigarrow x \; ; \; x \rightsquigarrow \texttt{Miscanthus}$. An important knowledge about Miscanthus is that it must not be cultivated in a zone susceptible to flooding, which can be expressed as `plot(Miscanthus)` {DC, EC} `flood_plain`.

In this example, ψ contains the constraints

$$C_{\texttt{DK}} = \{\texttt{plot(Miscanthus)}\ \{\text{DC, EC}\}\ \texttt{flood\_plain}\}$$
$$C_{\alpha(\texttt{Source})} = \left\{\begin{array}{l} \texttt{low\_water\_channel}\{\text{NTPP}\}\ \texttt{flood\_plain} \\ \texttt{plot}(x)\ \{\text{EC}\}\ \texttt{low\_water\_channel} \end{array}\right\}$$

and μ contains the constraints

$$\begin{aligned} C_{\texttt{DK}} &= \{\texttt{plot(Miscanthus)}\ \{\text{DC, EC}\}\ \texttt{flood\_plain}\} \\ C_{\texttt{Target}} &= \{\texttt{low\_water\_channel}\ \{\text{NTPP}\}\ \texttt{flood\_plain}\} \\ C_{\varrho} &= \{\texttt{plot}(x)\ \{\text{EQ}\}\ \texttt{plot(Miscanthus)}\} \end{aligned}$$

The first step in the algorithm is to add missing variables and constraints. In the example, all four variables are present in both QCNs, but some relations are missing, e.g. between `plot(Miscanthus)` and `low_water_channel`. A constraint `plot(Miscanthus)` {DC, EC, PO, TPP, NTPP, TPPi, NTPPi, EQ} `low_water_channel` is therefore added to both μ and ψ.

This manipulation may complexify the QCNs, which is part of the reason why computing the algebraic closure is interesting. Here, the amount of potential scenarios is reduced from 1024 to 16 for ψ, and from 1024 to 4 for μ.

Then, the lower bound on the distance between ψ and μ is computed. Here, the lower bound is 3, which happens to be the distance between ψ and μ. Only one scenario $\mathcal{T}$ of μ is found at this distance: $\psi \dot{+} \mu = \{\mathcal{T}\} = \{(V_{\mathcal{T}}, C_{\mathcal{T}})\}$ with

$$C_{\mathcal{T}} = \left\{ \begin{array}{l} \texttt{low\_water\_channel}\{\text{NTPP}\}\ \texttt{flood\_plain} \\ \texttt{low\_water\_channel}\{\text{DC}\}\ \texttt{plot}(x) \\ \texttt{low\_water\_channel}\{\text{DC}\}\ \texttt{plot(Miscanthus)} \\ \texttt{flood\_plain}\{\text{EC}\}\ \texttt{plot}(x) \\ \texttt{flood\_plain}\{\text{EC}\}\ \texttt{plot(Miscanthus)} \\ \texttt{plot}(x)\ \{\text{EQ}\}\ \texttt{plot(Miscanthus)} \end{array} \right\}$$

The distance is the sum of the following replacements: EC becomes DC between `plot`(x) and `low_water_channel` ($d = 1$), {TPP, NTPP, PO} becomes EC between `flood_plain` and `plot`(x) ($d = 1$), and {DC, EC, NTPPi, PO} becomes EQ between `plot`(x) and `plot (Miscanthus)` ($d = 1$).

In this scenario, the region `plot`(x) was reduced in order not to overlap with `flood_plain` as it was equated to `plot (Miscanthus)`. This corresponds to the allocation shown in figure 1b. It can be seen that the modification is indeed minimal, as the plot becomes externally connected to the flood plain, maximising the area used for Miscanthus cultivation. For instance, a result including `flood_plain` {DC} `plot(Miscanthus)` would have been consistent with the domain knowledge but would not have constituted a minimal modification of ψ. Therefore, the adaptation is successful.

6.2 Complete Agronomic Problem

To obtain the more productivity-increasing adaptation described in section 2.1, the plot region is broken into `plot1`(x) and `plot2`(y), and $\varrho = x \rightsquigarrow \texttt{Miscanthus}\ ;\ y \rightsquigarrow \texttt{maize}$. In ψ, the only information about both plots is that they are externally connected to the low water channel.

The revision algorithm returns 5 scenarios. All of them address the domain knowledge and the constraint to maximise the size of the Miscanthus plot, but they vary in their allocation of maize. One corresponds to the allocation shown in figure 1c. The other ones are similar.

This example also shows that the algorithm handles multiple substitutions.

6.3 Cooking Example

Most temporal aspects of recipes can be represented in $\mathcal{INDU}$ by reifying cooking actions, ingredient states, and durations as intervals. For instance, the following could be included in the domain knowledge: $\texttt{cooking(carrot)}\ \{\mathrm{m}\}^{?}\ \texttt{cooked(carrot)}$ and $\texttt{cooking(carrot)}\ ?^{=}\ \texttt{25\_min}$, with the provision that, e.g. $\texttt{18\_min}\ ?^{<}\ \texttt{25\_min}$.

In such a simple problem, combining constraints is straightforward and makes it possible to limit the amount of variables. The problem described in section 2.2 can be compressed to just 4 variables by replacing duration intervals by duration relations between the relevant action intervals. In this representation, ψ contains

$$C_{\texttt{DK}} = \left\{ \begin{array}{l} \texttt{cooking(rice)}\ ?^{<}\ \texttt{cooking(carrot)} \\ \texttt{cooking(rice)}\ \{\mathrm{m}\}^{?}\ \texttt{serve} \\ \texttt{cooking(carrot)}\ \{\mathrm{m}\}^{?}\ \texttt{serve} \end{array} \right\}$$
$$C_{\alpha(\texttt{Source})} = \left\{ \texttt{cooking}(x)\{\mathrm{f}^{<}\}\texttt{serve} \right\}$$

In TAAABLE, there is no firm adaptation constraint from $\texttt{Target}$ ($C_{\texttt{Target}} = \emptyset$) therefore μ contains simply the constraints

$$C_{\texttt{DK}} = \left\{ \begin{array}{l} \texttt{cooking(rice)}\ ?^{<}\ \texttt{cooking(carrot)} \\ \texttt{cooking(rice)}\ \{\mathrm{m}\}^{?}\ \texttt{serve} \\ \texttt{cooking(carrot)}\ \{\mathrm{m}\}^{?}\ \texttt{serve} \end{array} \right\}$$
$$C_{\varrho} = \{\texttt{cooking}(x)\ ?^{=}\ \texttt{cooking(carrot)}\}$$

The revision algorithm returns two scenarios which are predictably distinguished only by the duration relation between $\texttt{serve}$ and the other actions, since this relation is defined as being unimportant in the domain knowledge. One scenario $\mathcal{T} = (V_{\mathcal{T}}, C_{\mathcal{T}})$ is such that $C_{\mathcal{T}}$ is

$$\left\{ \begin{array}{ll} \texttt{cooking}(x)\{\mathrm{m}^{>}\}\texttt{serve}, & \texttt{cooking}(x)\{\mathrm{eq}^{=}\}\texttt{cooking(carrot)} \\ \texttt{cooking(carrot)}\{\mathrm{m}^{>}\}\texttt{serve}, & \texttt{cooking}(x)\{\mathrm{fi}^{>}\}\texttt{cooking(rice)} \\ \texttt{cooking(rice)}\{\mathrm{m}^{>}\}\texttt{serve}, & \texttt{cooking(carrot)}\{\mathrm{fi}^{>}\}\texttt{cooking(rice)} \end{array} \right\}$$

In both scenarios, the lengthening of the vegetable cooking is associated with the inversion of the relation between the vegetable and the rice, i.e. $\mathrm{f}^{<}$ becomes $\mathrm{fi}^{>}$, which corresponds to the expected order inversion between the start of both actions. Therefore, the adaptation is successful.

7 Related Work

Some recent work deals with a combination of CBR and spatial reasoning, for instance in order to improve web services for spatial information [18], or for spatial event prediction in hostile territories [19]. Older work already underlined the interest of CBR to analyse geographical data, e.g. for soil classification [20]. We worked on a CBR system to help agronomists analysing farm surveys [21]. The model was based on conceptual graphs, with labelled vertices and edges, describing the spatial organisation of farm territories. The assumption was that similar spatial organisations correspond to similar functional organisations. The *spatio-functional cases* were represented within a description logic system, and reasoning relied on a combination of hierarchical classification (in the description logic sense), CBR and QSTR.

Several research work focused on the representation of time within the CBR framework. Most were interested in the analysis or in the prediction of temporal processes (e.g. breakdown or disease diagnosis starting from regular observations or successive events). The temporal aspect is generally taken into account from sequences of events or sometimes from relative or absolute time stamps [22–24]. Particularly, the problem of temporal adaptation has been given much attention in CBR with a workflow representation [25]. Only a few work [26, 27] adopted a qualitative representation of time, such as the Allen interval algebra. In [27], cases are represented by temporal graphs and the retrieval step is based on graph matching. In [26], cases are indexed by chronicles and temporal constraints, which are represented with a subset of Allen relations. Case-based planning (CBP, see e.g. [28]) is a research field which also deals with time. The main difference between classical CBP and our approach is that they deal with different types of knowledge: CBP deals with the achievement of goals and models actions by their applicability and effects, whereas our approach, applied to a process represented by temporal constraints between actions, deals with the known constraints between actions reified as intervals. Theoretically, these approaches could be combined, but they are generally designed for different purposes: classical CBP usually deals with efficiency and uses complete problem-solving knowledge whereas our approach deals with incompletely described knowledge (e.g. the effects of cooking actions are incompletely formalised).

8 Conclusion

Qualitative algebras are important to the field of knowledge representation and are especially useful for qualitative reasoning on space and on time, but their use in CBR has received very little attention so far. This paper focuses on the adaptation of cases represented in a qualitative algebra. A landscape agronomy example uses the spatial algebra RCC8, and a cooking example uses the temporal algebra $\mathcal{INDU}$. This adaptation uses the principles of revision-based adaptation and combines it with a matching between the source and target cases.

A prototype for adaptation of cases represented in a qualitative algebra has been implemented in Perl and applied to the examples of this paper, but it is very time-consuming and requires a lot of improvement in order to be integrated into an operational system like TAAABLE. Several optimisations are planned. First, the program can be optimised thanks to certain characteristics of the revision problem that are not taken into account by the current prototype:

- The fact that the source case usually represents a specific problem-solving episode, thus the QCN `Source` is usually satisfied by only few scenarios.
- The fact that the difference between QCNs arise because of σ, which should therefore be looked upon as the possible origin of inconsistencies.

Second, the study of how a process of "repair propagation" in a QCN can be designed is planned. This is similar to the classical constraint propagation algorithm, and can also be likened to the adaptation process presented in [11].

As the landscape agronomy example shows, the QCN can appear in the query of the CBR system. This means that for such an application, the comparison of QCNs also has to be done at retrieval time. A future work will be to study how to implement such a retrieval process.

At a more abstract level, this work, as well as all the previous studies on the use of belief revision and belief merging for single and multiple case adaptation (cf. section 3.2), shares some intuitive ideas with the notion of reuse based on asymmetric and symmetric amalgams (see, e.g. [29]). A precise comparison between these two general approaches to adaptation remains to be carried out.

References

1. Clifton-Brown, J.C., Stampfl, P.F., Jones, M.B.: Miscanthus biomass production for energy in Europe and its potential contribution to decreasing fossil fuel carbon emissions. Global Change Biology 10(4), 509–518 (2004)
2. Hillier, J., Whittaker, C., Dailey, G., Aylott, M., Casella, E., Richter, G.M., Riche, A., Murphy, R., Taylor, G., Smith, P.: Greenhouse gas emissions from four bioenergy crops in England and Wales: Integrating spatial estimates of yield and soil carbon balance in life cycle analyses. GCB Bioenergy 1(4), 267–281 (2009)
3. Martin, L., Wohlfahrt, J., Le Ber, F., Benoît, M.: L'insertion territoriale des cultures biomasses pérennes: le cas du miscanthus par dix agriculteurs de Côte d'Or. L'Espace Géographique (in press, 2012)
4. Zub, H.W., Brancourt-Hulmel, M.: Agronomic and physiological performances of different species of Miscanthus, a major energy crop. a review. Agronomy for Sustainable Development 30(2), 201–214 (2010)
5. Cojan, J., Dufour-Lussier, V., Gaillard, E., Lieber, J., Nauer, E., Toussaint, Y.: Knowledge extraction for improving case retrieval and recipe adaptation. In: Computer Cooking Contest Workshop (2011)
6. Alchourrón, C.E., Gärdenfors, P., Makinson, D.: On the Logic of Theory Change: partial meet functions for contraction and revision. Journal of Symbolic Logic 50, 510–530 (1985)
7. Katsuno, H., Mendelzon, A.: Propositional knowledge base revision and minimal change. Artificial Intelligence 52(3), 263–294 (1991)
8. Lieber, J.: Application of the Revision Theory to Adaptation in Case-Based Reasoning: The Conservative Adaptation. In: Weber, R.O., Richter, M.M. (eds.) ICCBR 2007. LNCS (LNAI), vol. 4626, pp. 239–253. Springer, Heidelberg (2007)
9. Cojan, J., Lieber, J.: Conservative Adaptation in Metric Spaces. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 135–149. Springer, Heidelberg (2008)
10. Cojan, J., Lieber, J.: Belief Merging-Based Case Combination. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS (LNAI), vol. 5650, pp. 105–119. Springer, Heidelberg (2009)

11. Cojan, J., Lieber, J.: An Algorithm for Adapting Cases Represented in $\mathcal{ALC}$. In: 22th Internationational Joint Conference on Artificial Intelligence (2011)
12. Allen, J.F.: Maintaining knowledge about temporal intervals. Communications of the ACM 26(11), 832–843 (1983)
13. Pujari, A.K., Kumari, G.V., Sattar, A.: INDU: An Interval & Duration Network. In: Foo, N.Y. (ed.) AI 1999. LNCS, vol. 1747, pp. 291–303. Springer, Heidelberg (1999)
14. Randell, D.A., Cui, Z., Cohn, A.G.: A spatial logic based on regions and connection. In: Knowledge Representation, pp. 165–176 (1992)
15. Ligozat, G.: On generalized interval calculi. In: Proceedings of the 9th National Conference of the American Association for Artificial Intelligence (AAAI), pp. 234–240. AAAI Press/MIT Press, Anaheim, CA (1991)
16. Condotta, J.F., Kaci, S., Schwind, N.: A framework for merging qualitative constraints networks. In: Wilson, D., Lane, H.C. (eds.) FLAIRS Conference, pp. 586–591. AAAI Press (May 2008)
17. Goguen, J.A.: What is unification? In: Resolution of Equations in Algebraic Structures, pp. 217–261. Academic Press (1989)
18. Osman, T., Thakker, D., Yang, Y., Claramunt, C.: Semantic Spatial Web Services with Case-Based Reasoning. In: Carswell, J.D., Tezuka, T. (eds.) W2GIS 2006. LNCS, vol. 4295, pp. 247–258. Springer, Heidelberg (2006)
19. Li, H., Muñoz-Avila, H., Bramsen, D., Hogg, C., Alonso, R.: Spatial Event Prediction by Combining Value Function Approximation and Case-Based Reasoning. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS (LNAI), vol. 5650, pp. 465–478. Springer, Heidelberg (2009)
20. Holt, A., Benwell, G.L.: Case-Based Reasoning and Spatial Analysis. Journal of the Urban and Regional Information Systems Association 8, 27–36 (1996)
21. Le Ber, F., Napoli, A., Metzger, J.L., Lardon, S.: Modeling and comparing farm maps using graphs and case-based reasoning. Journal of Universal Computer Science 9(9), 1073–1095 (2003)
22. Dojat, M., Ramaux, N., Fontaine, D.: Scenario recognition for temporal reasoning in medical domains. Artificial Intelligence in Medicine 14, 139–155 (1998)
23. Ma, J., Knight, B.: A Framework for Historical Case-Based Reasoning. In: Ashley, K.D., Bridge, D.G. (eds.) ICCBR 2003. LNCS (LNAI), vol. 2689, pp. 246–260. Springer, Heidelberg (2003)
24. Sánchez-Marré, M., Cortés, U., Martínez, M., Comas, J., Rodríguez-Roda, I.: An Approach for Temporal Case-Based Reasoning: Episode-Based Reasoning. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS (LNAI), vol. 3620, pp. 465–476. Springer, Heidelberg (2005)
25. Minor, M., Bergmann, R., Görg, S., Walter, K.: Towards Case-Based Adaptation of Workflows. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010. LNCS (LNAI), vol. 6176, pp. 421–435. Springer, Heidelberg (2010)
26. Jaczynski, M., Trousse, B.: WWW Assisted Browsing by Reusing Past Navigations of a Group of Users. In: Smyth, B., Cunningham, P. (eds.) EWCBR 1998. LNCS (LNAI), vol. 1488, pp. 160–171. Springer, Heidelberg (1998)
27. Jære, M.D., Aamodt, A., Skalle, P.: Representing Temporal Knowledge for Case-Based Prediction. In: Craw, S., Preece, A.D. (eds.) ECCBR 2002. LNCS (LNAI), vol. 2416, pp. 174–188. Springer, Heidelberg (2002)
28. Munoz-Avila, H., Cox, M.: Case-Based Plan Adaptation: An Analysis and Review. IEEE Intelligent Systems 23(4), 75–81 (2008)
29. Manzano, S., Ontañón, S., Plaza, E.: Amalgam-Based Reuse for Multiagent Case-Based Reasoning. In: Ram, A., Wiratunga, N. (eds.) ICCBR 2011. LNCS (LNAI), vol. 6880, pp. 122–136. Springer, Heidelberg (2011)

eCo: Managing a Library of Reusable Behaviours*

Gonzalo Flórez-Puga, Guillermo Jiménez-Díaz, and Pedro A. González-Calero

Complutense University of Madrid, Madrid, Spain
{gflorez,gjimenez}@fdi.ucm.es, pedro@sip.ucm.es

Abstract. Building the behaviour for non-player characters in a game is a complex collaborative task among AI designers and programmers. In this paper we present a visual authoring tool for game designers that uses CBR techniques to support behaviour reuse. Our visual editor (called *eCo*) is capable of storing, indexing, retrieving and reusing behaviours previously designed by AI programmers. One of its most notable features is the *sketch-based retrieval*: that is, searching in a repository for behaviours that are similar to the one the user is drawing, and making suggestions about how to complete it. As this process relies on graph behaviour comparison, in this paper, we describe different algorithms for graph comparison, and demonstrate, through empirical evaluation in a particular test domain, that we can provide structure-based similarity for graphs that preserves behaviour similarity and can be computed at reasonable cost.

1 Introduction

Building behaviours for non-player characters (NPC) in a game is a collaborative effort among designers and programmers. Programmers provide designers with the *building blocks* for specifying behaviour in the game, as a collection of parameterized systems, entity types and actions those entities may execute. Designers compose those basic pieces to specify complex behaviours.

Typically in a large game we can find simple behaviours that are replicated within different complex behaviours. For instance, in a soccer game, *defend* could be a complex behaviour that is composed of two simpler behaviours like *go to the ball* and *clear*; meanwhile *attack* could be made up of *go to the ball*, *dribbling* and *shoot*.

A common approach to help designers is to let them use visual languages that are supposed to facilitate the process by hiding the formal syntax of the underlying programming language. *UnrealKismet*, integrated in the *Unreal Development Kit* game editor, and *Flow-Graph Editor*, integrated in the Sandbox Editor of *CryENGINE 3 SDK*, are of two such visual scripting tools that let designers model the gameplay of a level without touching a single line of code through some variation of data flow diagrams. Also, the celebrated *Unity 3D* has plugins like *Behave* or *Playmaker*, that allow designers to use well-known

* Funded by Complutense University of Madrid. Supported by the Spanish Ministry of Science and Education (TIN2009-13692-C03-03).

techniques like Finite State Machines (FSMs) [1] or Behaviour Trees (BTs) [2] to design the behaviours. There is a whole range of genre-specific tools to assist in content authoring for particular types of games, such as serious games for procedural training [3], or even to autonomously generate content as in [4]. Most of these tools try to facilitate the design task by using different techniques for representing behaviours but do not cover the reusing of behaviours. Those that do just allow adding and removing behaviours to a collection. Searching in the collections, when it is allowed, is limited to a textual name search.

The motivation for the work presented here is an authoring tool for game designers, that integrates CBR techniques into the game authoring domain. The novelty of our approach is to leverage a collection of reusable behaviours. Without supporting tools and technology, reuse is barely an option, and game designers tend to develop new behaviours from scratch, resulting in variations of similar behaviours coexisting in the same game, ignoring the benefits of reuse in terms of quality and scalability. *eCo*, the editor we are developing, provides the tools to store, index, retrieve and reuse previously designed behaviours. Although in this paper we exemplify the approach with behaviours represented as Hierarchichal Finite State Machines (HFSMs) [5], the editor can deal with other formalisms typically employed for designing behaviour in video games, such as FSMs and BTs.

One of the most notable features of our editor is its capability for *sketch-based retrieval*: searching a repository for behaviours that are similar to the one the user is drawing, and making suggestions about how to complete it. Despite the usefulness of this feature, it poses a difficult problem and an open question.

Regardless of the formalism employed –FSMs, HFSMs or BTs–, sketch-based retrieval essentially translates into comparing a graph, the one being sketched, against a collection of graphs representing reusable behaviours. Unfortunately, the problem of assessing similarity between two graphs easily becomes intractable when using methods that take the graph structure into account. Some methods for assessing similarity between two graphs, such as graph edit distance, are based on finding a subgraph isomorphism, which is an NP-complete problem [6]. Essentially, it requires enumerating every possible mapping from the nodes and edges of one graph into the other, in order to determine which mapping maximizes similarity. Given the difficulty of finding the best isomorphism between two graphs, we make use of an alternative heuristic approach that finds solutions which are practically indistinguishable from the optimum solution in a particular domain.

The open question in sketch-based retrieval of similar behaviours through graph comparison is whether structurally similar graphs actually represent similar behaviours. In order to answer this question, we characterize behaviours in a quantifiable and parametric perspective by using gameplay metrics. Gameplay metrics are data extracted from computer game engines during play. The analysis of these metrics has been used to derive *play-personas*, archetypes that describe the behaviour pattern of a human player [7], which we are using here to characterize a synthetic one.

One of the advantages of our approach is that it can be applied to different behaviour representations, provided that these representations are based in a graph. To use sketch-based retrieval we just have to extract the graph and apply the structural similarity functions to find other similar graphs.

In this paper, we present different algorithms for graph comparison, and demonstrate, through empirical evaluation in a particular domain, that we can provide structure-based similarity for graphs that preserves behaviour similarity and can be computed at reasonable cost.

The rest of the paper runs as follows. The next Section describes SoccerBots, our experimental domain. Section 3 describes the editor we are implementing and Section 4 describes the structure-based similarity functions that we consider and presents the heuristic used. Section 5 introduces the experimental set-up while Section 6 discusses the experimental results. Finally, Section 7 presents conclusions and future work.

2 Domain Description

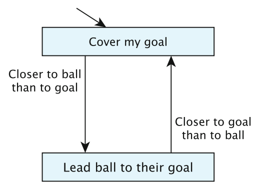
A very popular method to model the behaviour of NPCs in videogames are FSMs. FSMs are graphs in which the nodes represent the different states an NPC can be in, while the edges represent the transitions between the states. Nodes are labelled with the actions that are executed when the NPC reaches that state, and edges are labelled with the conditions that control the state changes.

One of the drawbacks of FSMs is that their complexity grows along with the number of states. A possible solution is to use HFSMs to represent the behaviours. HFSMs are an extension of the classic FSMs. Besides basic actions, a node in a HFSM can be labelled with another state machine. This way, the overall complexity of the behaviours is reduced, thus favouring their legibility. Any HFSM can be *flattened*, that is, transformed into an equivalent, non-hierarchical FSM, by following a simple algorithm.

Several reasons drove us to use HFSMs as the manner to represent the behaviours. Firstly, it is a technique widely used in video game development. Furthermore, for applying sketch-based retrieval we need behaviours that can be represented as graphs, so their structures can be compared. This makes techniques like HFSMs or BTs appropriate to represent them. And finally, we already have an extensive collection of state machines provided by the students of the Knowledge Based Systems course at the Complutense University of Madrid [8]. Every year, a competition is held between teams created by groups of students. We used the teams to create a repository and obtain the experimental results.

The visual editor described in this paper (see Section 3) helps the development of NPC behaviours by using HFSMs for SBTournament [8], a framework built on top of Soccerbots[1], a well-known simulation environment that simulates the dynamics and dimensions of a regulation RoboCup[2] small size robot league game.

[1] SoccerBots: `http://www.cs.cmu.edu/~trb/TeamBots/Domains/SoccerBots`
[2] Robocup: `http://www.robocup.org/`



Basic Behaviours

Cover goal

```
Vec2 dest =
   new Vec2(myRobotAPI.getOurGoal());
dest.add(myRobotAPI.getBall());
myRobotAPI.setSteerHeading(dest.t);
myRobotAPI.setSpeed(1.0);
```

Lead ball to goal

```
myRobotAPI.alignedToBallandGoal();
myRobotAPI.kick();
```

Fig. 1. "Forward" behaviour

Two teams of five robots compete on a soccer field by pushing and kicking a ball into the opponent's goal.

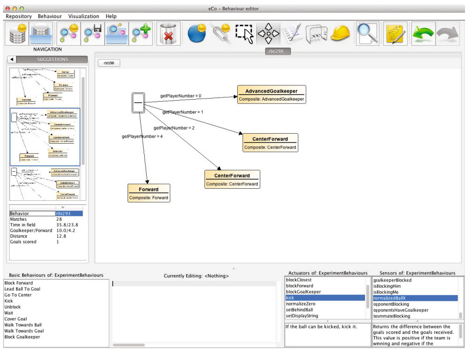
In order to implement robot behaviours, SBTournament provides users with a set of sensors and actuators, which are actually a superset of those provided by SoccerBots. Actuators are the most simple actions that a robot can execute, while sensors are the pieces of information that a robot can gather from the game world. For example, actuators in SBTournament allow users to kick the ball or set the desired heading and speed for a robot. Likewise, sensors provide information about the ball position or the position of the opponent's goal. We use sensors and actuators to build the HFSMs that our robots will execute. On one hand, we use sensors to build the conditions for the edges of the HFSMs. On the other hand, we use actuators to build the *basic behaviours*, i.e. the basic building blocks for the robot's behaviour. Basic behaviours are the simplest actions that can be executed in a node of a robot's HFSM. These basic behaviours generally consist of a sequence of calls to different actuators.

Figure 1 shows an example of a simple state machine for a striker. The state machine consists of two nodes. The initial one is labelled with the basic behaviour `Cover My Goal`, which uses the actuators `setSteerHeading` and `setSpeed` to go to a position in the path between the ball and the goal. The edge labelled `Closer to ball than to goal` checks the sensors `Distance to ball` and `Distance to goal` and compares their magnitude until the former is less than the latter. At that moment the state changes to `Lead Ball to Their Goal`, which tries to direct the ball towards the opponent's goal. This basic behaviour consists of two actuators: `Align with ball and goal` and `kick`. There is another edge, `Closer to goal than to ball`, that will be activated when the sensor `Distance to ball` has a value greater that `Distance to goal`.

3 Behaviour Editor

eCo[3], the behaviour editor we are presenting in this paper, allows users to "draw" HFSMs to specify the behaviour of SBTournament's robots and teams. As the

[3] eCo: http://gaia.fdi.ucm.es/research/eco-behaviour-editor



Behavior	rbt293
Matches	28
Time in field	35.8/23.8
Goalkeeper/Forward	10.0/4.2
Distance	12.8
Goals scored	1

Basic Behaviours of: ExperimentBehaviours
Block Forward
Lead Ball To Goal
Go To Center
Kick
Unblock
Wait
Cover Goal
Walk Towards Ball
Walk Towards Goal
Block Goalkeeper

Currently Editing: <Nothing>

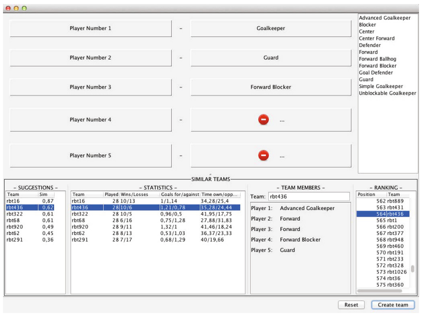
Actuators of: ExperimentBehaviours
blockClosest
blockForward
blockGoalKeeper
kick
normalizeZero
setBehindBall
setDisplayString

If the ball can be kicked, kick it.

Sensors of: ExperimentBehaviours
goalkeeperBlocked
isBlockingHim
isBlockingMe
normalizedBallX
opponentBlocking
opponentsHaveGoalkeeper
teammateBlocking

Returns the difference between the goals scored and the goals received. This value is positive if the team is winning and negative if the

(a) Player Edition Perspective



Player Number 1 – Goalkeeper
Player Number 2 – Guard
Player Number 3 – Forward Blocker
Player Number 4 – ...
Player Number 5 – ...

Advanced Goalkeeper
Blocker
Center
Center Forward
Defender
Forward
Forward Ballhog
Forward Blocker
Goal Defender
Guard
Simple Goalkeeper
Unblockable Goalkeeper

SIMILAR TEAMS

– SUGGESTIONS –

Team	Sim
rbt16	0,87
rbt436	0,62
rbt322	0,61
rbt68	0,61
rbt920	0,49
rbt62	0,45
rbt291	0,36

– STATISTICS –

Team	Played	Wins/Losses	Goals for/against	Time own/opp...
rbt16	28	10/13	1/1,14	34,28/25,4
rbt436	28	10/6	1,21/0,78	35,28/24,44
rbt322	28	10/5	0,96/0,5	41,95/17,75
rbt68	28	6/16	0,75/1,28	27,88/31,83
rbt920	28	9/11	1,32/1	41,46/18,24
rbt62	28	8/13	0,53/1,03	36,37/23,33
rbt291	28	7/17	0,68/1,29	40/19,66

– TEAM MEMBERS –
Team: rbt436
Player 1: Advanced Goalkeeper
Player 2: Forward
Player 3: Forward
Player 4: Forward Blocker
Player 5: Guard

– RANKING –

Position	Team
562	rbt889
563	rbt431
564	rbt436
565	rbt1
566	rbt200
567	rbt377
568	rbt948
569	rbt460
570	rbt191
571	rbt233
572	rbt328
573	rbt1026
574	rbt36
575	rbt360

(b) Team Edition Perspective

Fig. 2. Editing perspectives of the behaviour editor

user draws, the partially completed HFSM is used as a sketch to retrieve previously created behaviours. The users can reuse the retrieved behaviours to complete the one being edited. Once a HFSM is finished it can be exported and executed in SBTournament. Finished behaviours are added to a repository in order to be reused in the future. It is important to note that the use of HFSMs is only an aid to the user. For other tasks presented later in the paper, more precisely for the calculation of similarity, we transform the HFSMs and work with the flattened FSMs.

We distinguish between two perspectives for our editor: the Player Edition Perspective and the Team Edition Perspective.

Figure 2(a) shows the Player Edition Perspective. This perspective allows users to design individual players. The area in the middle is a canvas where users can draw the HFSMs that represent robot behaviour. Under the drawing canvas there is a code editor where users can create and modify the basic behaviours.

On the left of the canvas there is a suggestions panel. While building a behaviour, this panel shows the HFSMs representing other behaviours stored in the repository that are similar to the one being edited. The idea here is to use the behaviour being edited as a sketch of a complete behaviour. The sketch is an unfinished version of the desired behaviour that has empty nodes, undefined conditions in edges, missing pieces of functionality, etc. We retrieve similar behaviours based on the sketched one and present them to the user, who can combine them with or use instead the currently edited behaviour. When the user selects a suggested behaviour, the editor shows some statistics about this behaviour in the table below. The statistics are gathered by making the teams play versus a predefined set of *trainer teams*. The inner workings of the sketch-based retrieval procedure will be explained in the following sections.

The adaptation process is not automatized, but the system offers some assistance for manual adaptation. Information regarding the gameplay of the teams suggested can be employed by the users to adjust the team being built. For instance, if the user wants to develop a team that has a defensive gameplay she could compare her team with the teams suggested. She could then find a more defensive team (with few goals against or a small percentage of matches lost) but still similar to hers, and use it as a model to modify its configuration.

The Team Edition Perspective is shown in Fig. 2(b). It is designed to configure teams using the HFSMs created in the Player Edition Perspective. In the central area, users can assign a previously created HFSM to each robot within the team. The available HFSMs are displayed in the rightmost area.

The lower panel is populated with previously designed teams that are similar to the user's team, and with statistics regarding the gameplay of the teams suggested (e.g. average of goals or rate of wins). Using that information, users can complete their teams by adding the suggested behaviours to their formation, or finding out which teams will play in a similar fashion.

In addition to the statistical information, teams in the repository are ranked using an algorithm based on the Elo rating system [9]. Elo was originally a chess rating system, which nowadays has became a popular rating system for multiplayer competitions in several computer games. In particular, Elo has been adapted to team sports. Team ranking –on the lower right corner of the Team Edition perspective (see Fig. 2(b))– shows the position of the teams that are similar to the one being editing. This way the user can get an approximate idea of how well her team will perform.

The most important feature of the editor is that it uses the sketches of the behaviours being drawn by the user to retrieve similar behaviours from the repository. The retrieved behaviours are presented to the user, who can use them to transform the sketch into a complete behaviour. For complete behaviours like the ones in the repository, we can make them play and gather statistics about their gameplay to see if they are similar. But in the case of a sketch, that is not possible, because the behaviour is not finished yet. Instead, we have to rely on another similarity metric that allows us to compare behaviours and predict which of them behave similarly.

Next section describes the graph similarity functions we used in the sketch-based retrieval process.

4 Similarity Functions for Behaviours

As we said in the last section, we use HFSMs to represent the behaviours in the editor, but in the retrieval process we convert them to flat FSMs.

An FSM is a directed labelled graph. It can be defined as a tuple of 4 elements: $G = \langle N, E, \mu, \nu \rangle$, where

N is the set of nodes,
$E \subseteq \{N \times N\}$ is the set of edges,
$\mu : N \rightarrow L_N$ is the node labelling function. It assigns a label to each node in N. L_N, the set of labels, is composed of the basic actions,
$\nu : E \rightarrow L_E$ is the edge labelling function. It assigns a label to each edge. L_E is the set of labels for the edges, that are built using the sensors.

```
Input: Two graphs $G = \langle N, E, \mu, \nu \rangle$ and $G' = \langle N', E', \mu', \nu' \rangle$
Output: The edit distance between $G$ and $G'$
begin
    $best\_cost \longleftarrow \infty$
    Add $\{\emptyset\}$'s to $N$ or $N'$ until $|N| = |N'|$
    foreach permutation $p = \{p_1, \ldots, p_{|N'|}\}$ of the elements of $N'$ do
        Create a mapping for nodes $M_N = \{(n_1, p_1), \ldots, (n_{|N|}, p_{|N'|})\}$
        Extend $M_N$ to a valid mapping $M$ by adding the corresponding edges
        Transform $M$ into its corresponding edit sequence $es$
        if $c(es) < best\_cost$ then $best\_cost = c(es)$
    end
    return $best\_cost$
end
```

Algorithm 1. "Classic" algorithm to obtain the edit distance

As a measure of similarity we use the edit distance for graphs [6], which is a generalization of the string edit distance. To calculate the edit distance we must define a set of elementary *edit operations*. We will consider the following set: adding a node (A), deleting a node (D) and changing the label of a node (C), and adding an edge (A'), deleting an edge (D') and modifying its label (C'). Each edit operation has a cost $c(op) \geq 0$. We used constant costs for the adding and deleting operations. For the label modifying operations' cost we used the similarity between source and target labels. This expresses more accurately the intuitive idea that changing one label for another is cheaper in cost if the labels are similar. Hence, to completely define the modifying operations we need a node similarity function $(sim_N(\mu(n), \mu'(n')))$ and an edge similarity function $(sim_E(\nu(e), \nu'(e')))$ that allows us to compute the similarity value between two labels. The cost of a modifying operation will be inversely proportional to the similarity of the labels.

The edit distance between two graphs, G and G', is the minimum cost among all the possible *edit sequences*, the sequencies of edit operations that transform G into G'. Although it is a measure of dissimilarity, it can be easily converted into a similarity measure:

$$\mathrm{sim}(G, G') = \frac{1}{1 + \mathrm{dist}(G, G')}$$

A straightforward way to calculate the edit distance between G and G' is using Algorithm 1. The idea of the algorithm is to obtain all the valid mappings from G to G', calculate their cost and keep the one with the minimum cost.

A *valid mapping* M between two graphs G and G' is a mapping where:

- All nodes from G are mapped to a node in G' or to the *empty node* $\emptyset$, and vice versa.

$$\forall n \in N \quad (n, \emptyset) \in M \vee \exists n' \in N', (n, n') \in M$$
$$\forall n' \in N' \quad (\emptyset, n') \in M \vee \exists n \in N, (n, n') \in M$$

We will refer as the *node mapping* of M (M_N) to the subset of M that refers only to the nodes:

$$M_N = \{ (n, n') \in M | \, n \in N \cup \{\emptyset\} \wedge n' \in N' \cup \{\emptyset\}\}$$

– All edges from G are mapped to an edge in G' or to the *empty edge* $\emptyset$, and vice versa.

$$\begin{array}{ll} \forall e \in E & (e, \emptyset) \in M \vee \exists e' \in E', (e, e') \in M \\ \forall e' \in E' & (\emptyset, e') \in M \vee \exists e \in E, (e, e') \in M \end{array}$$

– The structure is preserved by the mapping. That is, if a node o in G is mapped to a node o' in G', and a node t in G is mapped to a node t' in G' then, if there is an edge e from o to t and an edge e' from o' to t', e has to be mapped to e'. If there is no edge from o' to t', e will be mapped to the *empty edge* $\emptyset$.

$$\begin{array}{l} \forall e \in E, \; e = (o, t) \\ \forall e' \in E', \; e' = (o', t') \\ (o, o') \in M \wedge (t, t') \in M \Leftrightarrow (e, e') \in M \end{array}$$

The node mapping M_N can be extended to a valid mapping M by adding the missing edge mappings. The new pairs of edges to be added are implied by the node pairs already in M_N [10]. We can transform the valid mapping into an edit sequence es by applying the following transformations to each pair:

– $(n, n') \longrightarrow C(n, l')$, with $n \in N, n' \in N', l' \in L_N$ and $\mu(n') = l'$
– $(n, \emptyset) \longrightarrow D(n)$, with $n \in N$
– $(\emptyset, n') \longrightarrow A(l')$, with $n' \in N', l' \in L_N$ and $\mu(n') = l'$
– $(e, e') \longrightarrow C'(e, l')$, with $e \in E, e' \in E', l' \in L_E$ and $\nu(e') = l'$
– $(e, \emptyset) \longrightarrow D'(e)$, with $e \in E$
– $(\emptyset, e') \longrightarrow A'(l')$, with $e' \in E', l' \in L_E$ and $\nu(e') = l'$

The edit distance is defined as the cost of the cheapest edit sequence that can transform G into G'. This means that we should evaluate each possible edit sequence to obtain the minimal cost.

From the conditions given before it is easy to infer that, given a *node mapping* M_N there is only one valid mapping M such that $M_N \subseteq M$. This means that the number of possible valid mappings equals the number of possible node mappings, which is:

$$\text{mappings}(N, N') = \frac{|N|!}{(|N| - |N'|)!}, \text{ supposing that } |N| > |N'|$$

Thus, in the general case, if we use the proposed algorithm the number of evaluations of the similarity functions grows factorially with the number of nodes in the graphs, making it unfeasible when the number of nodes begins to grow. The FSMs that we are evaluating have around 20 or 25 nodes. Consequently, we need to use a method with lower complexity.

```
Input: Two graphs $G = \langle N, E, \mu, \nu \rangle$ and $G' = \langle N', E', \mu', \nu' \rangle$
Output: A heuristic edit distance between $G$ and $G'$
1 begin
2     Add $\{\emptyset\}$'s to $N$ or $N'$ until $|N| = |N'|$
3     Create a cost matrix $CM$ where each $CM_{ij} = c((n_i, n'_j))$
4     Obtain the best node mapping $M_N^* \longleftarrow$ Hungarian($CM$)
5     Extend $M_N^*$ to a valid mapping $M^*$
6     Transform $M^*$ into an edit sequence $es$
7     return $c(es)$
8 end
```

Algorithm 2. Heuristic edit distance algorithm

4.1 Proposed Heuristic

To reduce complexity we opted to use the heuristic method proposed in [10]. Instead of searching the whole solution space, the idea is to generate only one mapping that is close enough to the best one. Algorithm 2 shows the pseudocode.

To obtain a good edit sequence, we generate the *best node mapping*, M_N^* , that is, the node mapping with the lowest cost. The problem of obtaining the *best node mapping* is equivalent to the *assignment problem.* In the assignment problem there are a number of agents and a number of tasks. Any agent can be assigned to perform any task, incurring some cost that may vary depending on the agent-task assignment. The problem is solved by assigning exactly one agent to each task in such a way that the total cost of the assignment is minimized. The assignment problem can be solved by the Hungarian algorithm with complexity in $O\left(N^3\right)$, N being the number of agents or tasks [11]. The algorithm takes as input a cost matrix CM. Each element of this matrix, CM_{ij}, is the cost of assigning the task i to the agent j.

In our case, instead of assigning tasks to agents we have to assign the nodes of a graph to the nodes of the other (or to the empty node). Thus, to build the cost matrix we use the cost of the corresponding edit operations. Each element CM_{ij} of the cost matrix is the cost of the edit operation concerning nodes n_i and n'_j. There are three possible situations:

- $n_i \neq \emptyset$ and $n'_j = \emptyset \Rightarrow CM_{ij} = c(D(n_i))$
- $n_i = \emptyset$ and $n'_j \neq \emptyset \Rightarrow CM_{ij} = c(A(l'))$, with $\mu(n'_j) = l'$
- $n_i \neq \emptyset$ and $n'_j \neq \emptyset \Rightarrow CM_{ij} = c(C(n_i, l'))$with $\mu(n'_j) = l'$

As we can see, the cost function is of crucial importance because we use it to obtain the best node mapping and, therefore, the mapping we are using to obtain the distance between both graphs. Depending on the information contributed by this function, the results of the similarity will be more or less accurate. We propose two heuristic functions that differ in the cost function employed:

- Identity (f_{id}): the cost of modifying a node is the *identity* function, that is, 0 if the target node has the same label and 1 if the labels are different.

– Identity with edges (f_{edge}): this function compares two nodes using the identity function f_{id}, and adds to the result the similarity of the edges entering the nodes and leaving them.

To consider the initial node in the cost function, we add an extra cost value if one of the nodes is the initial and the other is not.

The node mapping M_N^* is extended to a valid mapping (M^*) by adding the missing edge mappings. To finally obtain the heuristic similarity value we transform the mapping M^* into an edit sequence an obtain its cost.

To compute the cost of the operation of modifying the label of an edge and, hence, the cost of modifying a node using the function f_{edge}we also need a similarity function for the labels of the edges. We used the Jaccard coefficient of the sets of sensors of each edge:

$$sim_E(l, l') = \frac{|\text{sensors}(l) \cap \text{sensors}(l')|}{|\text{sensors}(l) \cup \text{sensors}(l')|}$$

5 Experimental Setup and Procedure

The main feature of the described editor is the sketch-based retrieval. The challenge in this retrieval feature lies in finding HFSMs that *behave* similarly to a given one without the necessity of executing them, but by comparing their structure. What we want to determine is if the HFSMs retrieved using the proposed structural similarity functions also share a similar behaviour.

To run the experiment we built a test set of 700 teams, automatically generated from a set of roles. We applied some rules to the generation process to obtain consistent teams (for instance, we allow only one goalkeeper per team). The total number of nodes in the teams range from 14 to 35.

Then we issued several queries using the structural similarity function, and compared the ranked results with a reference list with the HFSMs that had a similar behaviour to the query.

5.1 Functional Similarity and Game Metrics Dataset

We need a reference measure that, given two HFSMs, verifies that they actually behave in a similar fashion. To obtain this function we simply "let the teams play" and gather some statistics about their gameplay. We use the values of those statistics to compare the behaviour of both teams. To extract the data we used the tools included in SBTournament, which allow users to generate SoccerBots tournaments and traces of robot behaviour[4].

To generate the statistics we selected a group of teams (the *trainers*) from the standard distribution of SoccerBots and from the winning teams of the students competition mentioned in Sect. 2. Each team in the test set played two matches of 1 minute versus each of the 14 trainers. Following this procedure for each

[4] SBTournament: `http://gaia.fdi.ucm.es/research/sbtournament`

team t, we obtained a vector of 21 attributes, $\boldsymbol{a_t} = (a_{0t}, \ldots, a_{20t})$, where each a_{jt} is the average value of attribute a_j for all the matches played by team t.

We measured attributes like the times in each field, near each goal and in the center region, the average distances to the goals, center of the field and the ball, the number of kicks the number of goals for and against or the instant of the first goal against.

The reference similarity measure employed to evaluate the proposed heuristic is built using those statistics. We can compute a local similarity measure for each attribute of each pair of teams. Thus, if we have two teams, t_i and t_j, we obtain a vector of local similarities:

$$sim_L(t_i, t_j) = (ls_0, \ldots, ls_{20}), \text{ with } ls_k = sim(a_{kt_i}, a_{kt_j}),$$
$$\text{where } sim(a_{kt_i}, a_{kt_j}) = 1 - \frac{\left|a_{kt_i} - a_{kt_j}\right|}{range(a_k)}$$

We can now use the local similarity vector of each pair of teams to compute a global similarity measure:

$$sim_{ref}(t_i, t_j) = \sum_{k=0}^{20} w_k \cdot ls_k$$

where $w = (w_0, \ldots, w_{20})$ is a vector of weights in which $\sum_{i=0}^{20} w_i = 1$. We will refer to this global similarity measure as the *functional similarity*.

We use the weights w_i to adjust the relevance of the attributes in our similarity function. Irrelevant or redundant attributes can have a negative effect on the results of the reference similarity function [12]. To obtain the weights we use an attribute selection algorithm that keeps the "good" attributes and rejects the "bad" ones. In terms of classification, we can state that an attribute is good if it is relevant to the class concept but not redundant to any other relevant attribute. That is, if it is highly correlated with the class but has little correlation with other attributes. In our case, the class concept is the team that produced the statistics. To measure this correlation we used *symmetric uncertainty* [13].

The attributes with higher weights were the times in each field, the time near the opponent's goal and in the center region and the distances to the center of the field and each goal

The *functional similarity* function is used to compute, for any pair of teams, a value of similarity that tells us how similarly they play. With this dataset we are ready to compare both functional and structural similarity functions.

5.2 Evaluating the Structural Similarity Function

To evaluate the structural similarity functions we compare the results provided by a retrieval process using the structural similarity functions with the results generated by using the reference (i.e. functional) similarity function described in the previous section. To do that, we compute the similarity of the FSMs associated to all the possible pairs of teams in the test set. We use each team t_q

as a query, and compare it with the remaining teams from the test set. For each query we obtain a list, l_{t_q}, of the similarities of t_q with all the remaining teams in the test set. The j-th element of the list is $\text{sim}\left(t_q, t_j\right)$.

The lists are sorted from most to least similar. Given a similarity function sim and a team t_q, we will refer to the ranking list generated as R_{sim,t_q}. In this case, the similarity values are unimportant: the topmost results returned is the only feature that matters.

The ranked lists are created using two structural similarity functions, according to the heuristics proposed in Section 4.1: f_{id} and f_{edge}. This way, for each team t_q we have an R_{id,t_q} list and an R_{edge,t_q}. Additionally, we created another reference list using the functional similarity function, R_{ref,t_q}.

The evaluation of the structural similarity functions is implemented by computing the *Precision at k documents retrieved* (P@k) [14] on the ranked lists. P@k is a variant of precision that only takes into account the first k documents retrieved by the query. For instance, when $k = 10$:

$$\text{P@10} = \frac{\text{relevant documents in the first 10}}{\text{documents retrieved (=10)}}$$

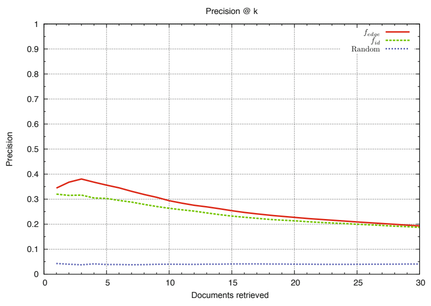
P@k is useful to compare the goodness of the results when we vary the number of elements retrieved (i.e. the value of k). When the user makes a query, it is important that the system will return good results, but also how many relevant results exist on the small list that the editor shows to the user.

To calculate precision we need to define which of the retrieved behaviours are relevant. We consider that a team t_i is relevant for a query if it behaves similarly, that is, if its similarity value $sim_{ref}\left(t_q, t_i\right)$ is over the percentile 95. It means that at least the 95% of the teams are less similar to t_q than t_i. In our case, having a sample size of 700 teams, this percentile represents that t_i should be ranked in the 30 first elements of R_{ref,t_q}.

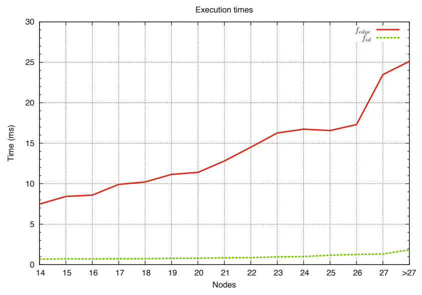
6 Results

In this section we present the results obtained from running the experiment described in the previous section. The experiments have been executed on a Dual Core Intel Xeon processor with 2.33 GHz, with 4 Gb of RAM, running Windows 2008 Server 32 bits and the Java JDK 6.0. Additionally to the two aforementioned structural similarity functions, we added a random similarity function that returns a random ranking for each query. We use the random ranking as a baseline to compare it with the results using the remaining functions.

Figure 3(a) shows the values of the P@k for different values of k in the interval between 1 and 30. When compared with other methods, the best results are returned by the one that uses the identity function together with the edges entering and leaving the nodes (f_{edge}). We have assessed the statistical significance of the differences between the average P@k for each k using the Wilcoxon rank sum test. The differences between the baseline (Random) and both our proposals (f_{edge} and f_{id}) are statistically significant with a 95% confidence level.



(a) Precision at k documents retrieved



(b) Experiment execution times

Fig. 3. Experiment precision and execution times

We can see that the information regarding the edges (f_{edge}) makes it behave better than the identity (f_{id}) to some extent. This enhancement in the precision is statistically significant with a 95% confidence level when the retrieved list contains between 2 and 19 elements (k value). However, as the number of elements increases, this difference can not be considered as significant, according to the statistical test employed.

The highest values of precision corresponds to lower values of k. This means that the relevant elements are more likely to be found at the first places of the ranking. For instance, for f_{edge}, the value of P@5 = 0.356 means that if we retrieve 5 documents there is an average of almost 2 relevant documents between the first 5:

$$\text{P@5} = \frac{\text{relevant documents retrieved in the first 5}}{5} = 0.356$$

$$\text{relevant documents retrieved} = 5 \cdot 0.356 = 1.78$$

Whereas P@15 = 0.254 indicates that in the first 15 teams retrieved there are around 4 relevant documents. That is, if we show 10 more elements to the user she will only find 2 more relevant elements among them.

Additionally to the precision, we have measured the times of the retrieval process. Figure 3(b) shows the average execution times for each function. X-axis represents the number of nodes of the queries and Y-axis represents the time in milliseconds. Each point in the graph represents the average time elapsed to compare a query with all the remaining behaviours in the test set. For instance, a query with 14 nodes takes an average time of 7.48 ms when using the f_{edge} heuristic. Note that we only measured times for the heuristic functions. Using data from past experiments [15] we can estimate that executing a query with 14 nodes using the non-heurisitic structural similarity, the expected execution time is approximately of 141 days.

When comparing the times for the heuristics, we can see that f_{edge} takes more time to execute. This is due to the extra calculations needed to obtain the similarity of the edges of each pair of nodes. The difference grows as long as the number of nodes increases. The reason for this behaviour is attributable to the different costs of the heuristic functions. f_{edge}, requires more time to be evaluated than f_{id}, and when the number of nodes increases, the number of times the function has to be evaluated does the same.

7 Conclusions and Future Work

In this paper, we have presented an authoring tool for game designers that leverage a collection of reusable behaviours by providing tools and techniques for storing, indexing, retrieving and reusing previously designed behaviours. The main contribution of the proposed tool is to provide a smooth mechanism for retrieving similar behaviours to the one being built, understanding that retrieval is crucial for this approach to scale-up when dealing with large collections of reusable behaviours.

Sketch-based retrieval relies on comparing a behaviour graph that is partially drawn, with respect to a repository of existing behaviours. We have described different algorithms to allow for this graph comparison, and provided and heuristic structure-based similarity for graphs that preserves behaviour similarity and can be computed at reasonable cost.

We have presented a case study domain: behaviours of SoccerBots robots represented as HFSMs. In our experiment we have demonstrated that if we consider only the best node mapping, we still find behaviours that "behave" in a similar way. We have proposed this heuristic algorithm, and we have incorporated it in a tool to aid game designers in the edition of HFSMs. In the editor, the user sketches the behaviour she wants to obtain by drawing an incomplete HFSM. The sketch is used by the editor to suggest similar behaviours using the proposed structural similairty function. Suggestions are done at two levels: at the single behaviour level we used player HFSMs to give suggestions and at the team behaviour level we used team HFSMs.

As we have also shown in Section 6, we obtain relevant results within the first few results retrieved. This means that, when showing the results to the user, she will be more likely to find the relevant behaviours in the first page of the results. This is a desirable quality for the users.

Regarding the applicability of the proposed approach, it should be noticed that sketch-based retrieval from a library of reusable behaviours only makes sense when reusing behaviours actually provides benefits to the authoring process. While in game genres such as sport games, shooters, or action games behaviours tend to repeat and therefore reuse should bring benefits, in more narrative-driven games it may be more difficult to find significant portions of reusable behaviours.

Regarding future work, we plan to carry out extensive experiments in order to detect usability issues in the tool, and actually assess the gains in productivity that can be obtained through the proposed approach. In such experiments, users would create several HFSMs for the SoccerBots environment from scratch and using our tool, so that we will be able to measure development time, quality of the results and user experience.

In addition, and in order to improve precision in retrieval, we plan to integrate more domain knowledge into the node similarity measure. To this end we are planning to embed the behaviours in a domain ontology and use a similarity measure that takes into account the relations between them. We expect that this added knowledge translates into a more accurate retrieval of the behaviours.

References

1. Bourg, D.M., Seemann, G.: AI for Game Developers. O'Reilly Media, Inc. (2004)
2. Flórez-Puga, G., Llansó, D., Gómez-Martín, M.A., Gómez-Martín, P.P., Díaz-Agudo, B., González-Calero, P.A.: Empowering Designers with Libraries of Self-Validated Query-Enabled Behaviourtrees. In: González-Calero, P.A., Gómez-Martín, M.A. (eds.) Artificial Intelligence for Computer Games, pp. 55–82. Springer Science+Business Media, LLC (2011)
3. Gerbaud, S., Mollet, N., Ganier, F., Arnaldi, B., Tisseau, J.: Gvt: a platform to create virtual environments for procedural training. In: IEEE VR Conference 2008, pp. 225–232 (2008)
4. Smith, G., Whitehead, J., Mateas, M., Treanor, M., March, J., Cha, M.: Launchpad: A rhythm-based level generator for 2-d platformers. IEEE Trans. Comput. Intellig. and AI in Games 3(1), 1–16 (2011)
5. Millington, I.: Artificial intelligence for games. The Morgan Kaufmann series in interactive 3D technology. Morgan Kaufmann Publishers Inc. (2006)
6. Bunke, H., Messmer, B.T.: Similarity Measures for Structured Representations. In: Haton, J.-P., Manago, M., Keane, M.A. (eds.) EWCBR 1994. LNCS, vol. 984, pp. 106–118. Springer, Heidelberg (1995)
7. Canossa, A., Drachen, A.: Patterns of play: Play-personas in user-centred game development. In: Barry, A., Helen, K., Tanya, K. (eds.) Breaking New Ground: Innovation in Games, Play, Practice and Theory: Proceedings of the 2009 DiGRA Conference, London, Brunel University (September 2009)
8. Jiménez-Díaz, G., Díaz-Agudo, B.: SB Tournament: Competiciones de robots en asignaturas de Inteligencia Artificial. In: Procs. of the 9th SIIE (November 2007)
9. Glickman, M.E.: Chess rating systems. American Chess Journal 3, 59–102 (1995)
10. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. Image Vision Comput. 27(7), 950–959 (2009)
11. Burkard, R.E., Dell'Amico, M., Martello, S.: Assignment Problems. SIAM (2009)
12. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Kaufmann series in data management systems. Morgan Kaufmann (June 2005)
13. Yu, L., Liu, H.: Feature selection for high-dimensional data: A fast correlation-based filter solution. In: ICML, pp. 856–863 (2003)
14. Manning, C.D., Raghavan, P., Schtze, H.: Introduction to Information Retrieval. Cambridge University Press, New York (2008)
15. Flórez-Puga, G., Díaz-Agudo, B., González-Calero, P.A.: Similarity measures in hierarchical behaviours from a structural point of view. In: Guesguen, H.W., Murray, R.C. (eds.) FLAIRS 2010, pp. 330–335. AAAI Press (May 2010)

Toward Measuring the Similarity of Complex Event Sequences in Real-Time

Odd Erik Gundersen[1,2,⋆]

[1] Verdande Technology, Norway
[2] Department of Computer and Information Science, Norwegian University of Technology and Science, Norway
odderik@verdandetechnology.com

Abstract. Traditional sequence similarity measures have a high time complexity and are therefore not suitable for real-time systems. In this paper, we analyze and discuss properties of sequences as a step toward developing more efficient similarity measures that can approximate the similarity of traditional sequence similarity measures. To explore our findings, we propose a method for encoding sequence information as a vector in order to exploit the advantageous performance of vector similarity measures. This method is based on the assumption that events closer to a point of interest, like the current time, are more important than those further away. Four experiments are performed on both synthetic and real-time data that show both disadvantages and advantages of the method.

Keywords: Similarity measures, sequence similarity, multi set similarity, set similarity, vector similarity, complex events, case-based reasoning.

1 Introduction

Combining events into situations, or complex events, is a recent trend which has coined the term complex event processing (CEP) [9]. CEP systems have the ability to detect complex patterns of events, and among them are event abstractions and detecting relationships between events, such as timing, ordering and sequencing [13] [11]. Many CEP systems utilize IF THEN rules, so if a given pattern of events is observed then a specific action should be taken [6]. CEP systems operate in high speed environments such as capital markets, which have strict requirements on latency, and hence they are shown to be very efficient. Events in CEP systems are significant, instantaneous and atomic occurrences [2] that notify about state changes, something that happens or a detectable condition [4]. Typical events include notifications when keys are pressed on a keyboard or placing an order in the capital marked.

Multi dimensional temporal abstraction (MDTA) on the other hand is a term used to describe methods that recognize patterns in online data streams in order

⋆ Corresponding author.

to add qualitative descriptions of the recognized patterns to the knowledge base [3]. Stacey et al. [14] present a system that performs MDTA on low level quantitative data from patient data streams logging parameters such as blood pressure and oxygen saturation in order to provide a better alarming system than traditional threshold alarms, which produce a large number of false positives. Data mining methods are used to recognize patterns in the data streams, and when they do, events are fired, adding qualitative interpretations of the quantitative data streams. The events trigger rules and activates reasoning in an ontology. While some data mining methods works directly on raw quantitative time-series data, MDTA methods produce qualitative, abstracted events in order to reduce the complexity when mining knowledge.

Systems that monitor stock markets, equipment, and the like and diagnostic systems that, for instance, support doctors in diagnosing patients, require identification of how events or symptoms relate to each other in regard to their temporal distribution. One way of solving this problem is to measure the similarity of events sequences. With the growth of temporal information, comparing sequences of events is getting more and more important. However, even though the growth in computing power still increases, comparing sequences of events is a computationally demanding task when the size of the sequences are large.

The research presented in this paper focus on sequences of events and not on raw quantitative time-series data, and thus it is applicable for both CEP systems and systems that perform MDTA, as recognizing similar situations comprised of sequences of events is important for a wide range of applications. We aim at developing fast methods for similarity assessment of sequences of events that can be used by case-based reasoning (CBR) systems [1] that can perform under real-time demands for large sequences of events.

This paper is structured as follows. We will start with describing the background and properties of our real-time CBR system in Section 2. Then, we will describe and formulate the problem we seek to solve in detail in Section 3. In Section 4, we analyse different strategies for comparing sequences and present our method. Results from some experiments are presented in Section 5, and finally, we will conclude with some remarks and future work in Section 6.

2 Background

We have implemented a real-time decision support system [5] that performs MDTA on real-time data streams. The MDTAs are represented as events, which specify different types of anomalies that are recognized in the data streams. Events are used by a real-time CBR system when comparing the current situation to past historic situations stored in the case base. In its simplest form, a case in this CBR system is a sequence of events distributed over time, and it represents a situation that is described by this event sequence, which has a specified start and end time. The CBR system captures new input cases continuously and compares them to all the cases in the case base, so it acts as a moving window of events describing the current situation. The main performance issue in this system is the event sequence similarity comparison, so improving the case retrieval step in the CBR engine will not meet our needs. We need to improve the performance of the sequence similarity measure, and it must approximate the similarity score of traditional sequence similarity measures. The CBR system has the following requirements:

– **Real-time demands:** Real-time performance is required from the case retrieval step. Currently, the update frequency of the data streams is between one and ten seconds, but the methods support higher update frequencies.
– **Case retrieval should not be tied to the recognition of new events:** The similarity of two sequences of events can change even though no new events are generated, as events move inside the input event sequence when time flows. Thus, the sequence in the input case might become more or less similar to sequences in the case base.
– **Varying number of events in the event sequences:** The event sequences will have a varying size; some might contain three while others hundreds.
– **Normalized similarity:** Normalizing the similarity score is convenient when combining similarities from different features, as is typically done in CBR and when comparing objects visually. As our decision support system utilizes CBR and visualizes how similar different situations are when compared, normalizing similarity scores is a requirement.

Every time step the current situation (the input case) is compared to each case in the case base, and the similarities between the current situation and the past cases are plotted as *case similarity graphs*, one for each case, with time on the x-axis and similarity on the y-axis. Case similarity graphs enable the user to understand how a case has matched with the current situation at each time step to track the similarity history of each case. More detail about this is given in Section 5.3.

3 Problem Description

In this paper we consider the problem of comparing the similarity between two sequences of complex events. The traditional approach is to divide and conquer. In this regard, divide and conquer means to compute the overall similarity of the sequences by comparing pairs of events from the sequences for similarity and combining the similarities of the pairs in order to find the overall sequence similarity. All the events in the two sequences must be compared to find the global maximum similarity score. As this is a combinatorial problem, the time order complexity grows so fast that the computation time degrades beyond usability with very few events in each sequence.

The concept of an edit distance has been introduced as a way of approximating the similarity of two sequences [16]. An edit distance is the minimum cost sequence of edit operations that is required in order to transform one sequence into another. Each edit operation has an associated cost. So a high cost of edit operations will result in a low global similarity score. The main assumption is that the more work that has to be done to transform one sequence into another, the less similar the sequences are. The concept was first introduced by [8].

Wagner and Fischer [16] propose the three operations *delete*, *insert* and *change*. The *change* operation is defined as changing one of the characters in one sequence with one in the other. The cost of the *change* operation is related to how far the characters are away from each other in the sequence; that is, how many characters the charcters that are being moved have to cross.

Mannila and Ronkainen [10] investigate event sequenceses of simple events distributed over a time period and not characters in strings. By simple events, they refer to events that only have a time stamp and a type. They propose what they call a generalization of edit distance, which is to relate the cost of moving one event to the point in time of another one of the same class directly to the distance in time. We argue that this is not a generalization.

On the one hand, it can be argued that they generelize the *concept of distance* used by the cost function *from* the more specialized cost of jumping over discrete character spaces *to* the more general cost of moving a distance on a continuous time scale. On the other hand, though, the cost function proposed by Wagner and Fisher measures order of events in the sequence while the cost function proposed by Mannila and Ronkainen measures distance between them on a time scale. A similarity measure, which is to take everything about a sequence of complex events into account, both operations, that is moving past another event and the distance moved on a time scale, should be measured. Seen in this light, the distance cost function proposed by Mannila and Ronkainen is not a generalization, but it is measuring an independent property of sequences.

Both string matching and simple event matching, as defined by Mannila and Ronkainen, compare events with a class and a position on a scale. The scale is continuous for time and discrete for the position of characters in a string. Because of this an event of a given type always matches best with the event that is closest to it on this scale. This property of simple event matching makes the time complexity of the original method by Wagner and Fisher $O(m \cdot n)$, where m is the length of one and n the length of the other. Ukkonen [15] has improved the method to work in $O(s \cdot min(m,n))$ both in space and time, where s is the maximum cost of computing the result.

These methods will not produce the maximum similarity score for sequences composed of more complex events, that is, events that have more than the one class attribute. As more complex events are described by more than one attribute, the closeness of two events of the same type on a scale is not the only basis for the similarity computation. Thus, the highest global similarity of a sequence is not necessarily found by pairing the closest events of a type, and it might not even include the event pair that produce the highest similarity individually. To find the global similarity then, an exhaustive search is required. An exhaustive search for the optimal event pairings is a combinatorial problem, and thus the computation time increases accordingly. Note that complex events in CEP systems are comprised of several events, while we differ between simple events that only are described by a class and a position and complex events that have several additional properties.

Obweger et al. [12] have developed a generic method for similarity searching in sequences of time-stamped complex events. This method is also based on the assumption that similarity is computed through finding the deviations between the pattern sequence (input sequence) and candidate sequences (stored sequence). The model they propose is highly configurable and balances aspects such as single event similarities, order, timing and missing events.

When the number of events in sequences vary between very few and up to hundreds, an exhaustive search for an optimal similarity comparison is not feasible in real-time. We have solved this by using a greedy implementation of an Edit Distance sequence similarity measure. Still, when the number of the events in a sequence grows above a certain number, the case retrieval starts lagging behind the required real-time update. Obweger et al. [12] reports the same problem; when the number of events grows beyond a threshold, the execution time increases drastically. Thus, together with the fact that real-time event systems are becoming more common, this clearly indicates that there is a need to develop faster sequence similarity measures for complex events. Our long term goal is to develop sequence similarity measures that are magnitudes faster than the current solutions while approximating the accuracy of the similarity measurement.

4 Faster Similarity Measurement of Event Sequences

In this section, we will propose a novel method for how to measure the similarity of sequences of complex events. First, we will present a short discussion on the properties of sequences. Then the idea behind the method, the actual approach and different versions of it will be presented.

4.1 On Measuring the Similarity of Sequences

The property that distinguishes a sequence from a multi set is the ordering of the elements. In a multi set, more than one instance of an element is allowed, which is what separates a multi set from a set, as sets are allowed to contain only one instance of an element. When measuring the similarity of two sequences the sequences can be regarded as sequences, reduced to sets or multi sets or any combination of these. In addition, the elements in the sequences can be measured for similarity individually. The one property of individual elements we will give most attention is the distance between them, but generalized to any scale and not restricted to time. The following list summarizes our view on the properties that characterize sequences and should be used when comparing event sequences:

– **Order:** Events in a sequence have a specific sequencing or order in the sequence. For example, it is the order of the characters that distinguish the meaning of text strings, such as *era* and *are*.

- **Dimensions:** The elements have a type that categorize them. Alarms might for instance be burglar or fire alarms. The types can be thought of as dimensions, and if the sequences are defined over orthogonal dimensions they will not be similar.
- **Amount:** The amount of events in a sequence is an important property to measure, and more specific is the amount of events of a specific type. Mannila and Ronkainen [10] define the similarity of sequences when it comes to amount so that the more occurrences an event type has, the less effect the difference in additional events of the same type have.
- **Individual events:** Individual events of the same type might be more or less similar. Properties of individual elements include, but are not limited to, position, as mentioned above, importance, severity and confidence. Even though two events share the same class they can be quite dissimilar.

Sequence similarity measures should take into account all of the above mentioned properties in order to measure the similarity accurately, which is is exactly what the model Obweger et al. [12] propose does. However, as mentioned above, it suffers from performance issues when the number of events grow. We aim at designing sequence similarity measures that combine these properties in novel ways while not degrading to the same degree when the number of events grows.

4.2 The Idea: Transforming a Sequence into a Vector

As sequence similarity measures that compare complex events are of higher time complexity, real-time comparison is only possible for sequences with few elements. This is not a good solution for CEP systems as they typically operate on a large number of events. Vector similarity measures, such as cosine similarity, which measures the amount and diversity of types, are typically of linear time complexity, which give them speed advantages. Vector similarity measures are often used in information retrieval for identifying texts that are similar. When comparing texts, each vector represents one text, and the components of the vectors represents the frequency of a given term that is found in that same text. The similarity of the two texts can then typically be found by computing the angle between or the length of the two vectors. However, vector similarity measures do not consider the sequence of the elements they compare.

As pointed out by Lamport in [7], "the order in which events occur in distributed systems is only a partial ordering." This is an argument for not only basing the similarity comparison of event sequences produced by distributed systems on the order of the events, as it is impossible to decide the order fully under these circumstances.

Our hypothesis is that, by transforming information about the sequence, such as the order of the elements, into a vector, we can reduce the time complexity while approximating the similarity measurements of more elaborate sequence similarity measures. In this paper, we will test this hypothesis by implementing several variants of a method that does exactly this and experiment with these on both synthetic and real-world data.

4.3 The Approach

The method works by converting the input and stored sequences into two vectors for which the components represent the union of event types in the sequences. Then, instead of comparing these sequences of complex events using a traditional sequence similarity measure for simple events, the vectors are compared using a vector similarity measure. In this way the comparison can be reduced from $O(s \cdot min(n, m))$ treating the events as simple to $O(n + m)$ including more features of the events than distance, where s is the maximum cost of producing the result and m and n are the lengths of the two sequences.

The two sequences S_A and S_B with k and l events will be converted to the vectors V_A and V_B, both with m components. Given that the k events in S_A represent E_A events and k in S_B represent E_B events, then $m = E_A \cup E_B$. Thus, each event, e_i, in a sequence belongs to one event type, E_j. Hence, $j \leq i$.

All events e in a sequence S are converted to a number that indicates the importance of the event in the sequence; it is given a weight, which we denote w. This weight w_i is related to the position pos_i of an event e_i in the sequence S. Each sequence have a start and end, and pos_i is computed in relation to these. The assumption is that events closer to the end of the sequence are more important. For the current situation, the events happening now are more relevant than events that happened two hours ago, which again are more relevant than the events that happened two years ago. Also, we assume that the sequences do not contain all events that have occurred since t_0, but that $end = t_{now}$ and $start = t_{now} - C$, where C denotes a window which specify an interesting situation that is represented by a sequence of events. Furthermore, the halving distance is the distance that an event has to be moved for the weight to be halved, and we define the halving distance, h, as: $h = end - start$, i.e. events at $start$ are half as important as those at end.

Hence, the weight, w_i, of an event, e_i, is computed based on its position, pos_i, relative to a halving distance, h, according to equation 1:

$$w_i = \frac{1}{2^{((end - pos_i)/h)}} \tag{1}$$

The weight, W_j of all events of the same type, E_j, is summed to represent the importance of the given type of events in the sequence, as shown in equation 2:

$$W_j = \sum_{i=0}^{n} w_i \tag{2}$$

The resulting vector representation of the sequence, S_A, is the vector, V_A, with event type weights as its components. Thus, each component in the vector is the sum of event weights, W_j, for a given type of events, E_j:

$$V_A = < W_0, W_1, ..., W_j > \tag{3}$$

The closer an event, e_i is to the end of the sequence, or point of interest (POI), the higher the weight, w_i, for that event. Given that a window $C = 100$ is chosen, an event that is positioned at position 100 (end) in the sequence has a weight of 1.0, while an event positioned at 0 (start) has a weight of 0.5. The effect of choosing a weighting function such as the one specified above is that events closer to the POI is deemed more important than events further away.

Equation 1 can be expanded in the following way to include more complex events than those composed of a type and a position on a scale:

$$w_i = \frac{1}{2^{((end - pos_i)/h)}} I_i, \tag{4}$$

where I_i represents other features of an event, e_i, such as confidence, severity and significance. For instance, $I_i = confidence_i \cdot severity_i \cdot importance_i$. The parameter I then specify how much the system believes in the event, how severe the event is, and how significant it is to describe the sequence, respectively. I can be set to 1 for w_i to only specify order, but if not, w_i shifts from specifying order to specify how meaningful the event is to the sequence it belongs to, that is, how much the event counts when measuring the similarity. In this way, the similarity measure can encode more information about events than only type and thus supports more complex events than events with a type only.

The importance of an event is determined by the position and varies between 0.5 and 1. Confidence, severity and significance are normalized between 0 and 1, and hence also the weight, w_i, of one event. However, the weight, W_j, of all events of the same type has a lower limit of 0, but no upper limit.

An advantage of the proposed method is that the stored sequences can be preprocessed and the vector cached, so that only the input sequence needs to be converted to a vector at every time step.

4.4 Proposed Vector Similarity Measures

Three vector similarity measures have been converted to work with vectors or invented for testing the specified approach. Set similarity measures such as the Dice and the Jaccard similarity coefficient were tested too, but as they did not differentiate between the synthetic test sequences, the results are omitted.

In the following equations upper case letters, such A, denote vectors while lower case letters, such as a, denote vector elements. Subscripts on lower case letters indicate the position of a vector element. Hence, a_i denotes the i-th vector element in vector A. $||A||$ denotes the length of vector A while $A \cdot B$ is the dot product of vector A and B. $|a_i - b_i|$ is the absolute value of the difference between the two vector elements a_i and b_i. The two functions $min(a, b)$ and $max(a, b)$ return the minimum and maximum value of the two parameters, respectively, while $\times$ is regular multiplication.

Cosine Similarity. The cosine similarity is a well known vector similarity measure that measures the angle between the two vectors, and it is commonly used in text mining and computer vision. A disadvantage of the similarity measure is that it does not differentiate between parallel vectors with differing magnitudes.

$$Cosim(A, B) = \frac{A \cdot B}{||A||||B||} = \frac{\sum_{i=0}^{n} a_i \times b_i}{\sqrt{\sum_{i=0}^{n} a_i^2} \times \sqrt{\sum_{i=0}^{n} b_i^2}} \tag{5}$$

Relative Component Fraction * Cosim. The relative component fraction (RCF) similarity divides the smallest vector component of a dimension by the largest one:

$$RCF(A, B) = \frac{\sum_{i=0}^{n} min(a_i, b_i)}{\sum_{i=0}^{n} max(a_i, b_i)} \tag{6}$$

Both the direction and the magnitude of the vectors will influence the the similarity score when multiplying the vector length fraction to the cosine similarity. This similarity measure will handle both parallel vectors and vectors of equal magnitude pointing in different directions.

$$RCFCos(A, B) = RCF(A, B) \times Cosim(A, B) \tag{7}$$

Nice Relative Component Fraction * Cosim. The nice version of RCF, NRCF, does not add min and max values if either a_i or b_i is equal to 0.0 (missing events of a given type), as the idea is that the cosine similarity penalizes for the dissimilarity in dimensionality.

$$NRCFCos(A, B) = NRCF(A, B) \times Cosim(A, B) \tag{8}$$

5 Experiments

In this section, we present the results from four different experiments designed to benchmark the similarity assessment and performance of the similarity measures presented in the previous section by comparing them to two regular divide and conquer sequence similarity measures, which serve as baselines. First, however, we present two sequence similarity measures that are used as baseline when comparing accuracy and performance results.

5.1 Baseline Sequence Similarity Measures

Edit Distance. The edit distance is computed by comparing all pairs of events from the input and stored sequences. The event pairs are sorted on similarity score, and the best match is computed by picking the pairs with highest similarity score that have not been picked yet. This implementation, ED, is of $O(n \cdot m)$ time complexity, where m and n are the lengths of the two sequences.

Edit Order. Is the basic variant presented by Wagner and Fisher [16] that measures order rather than distance.

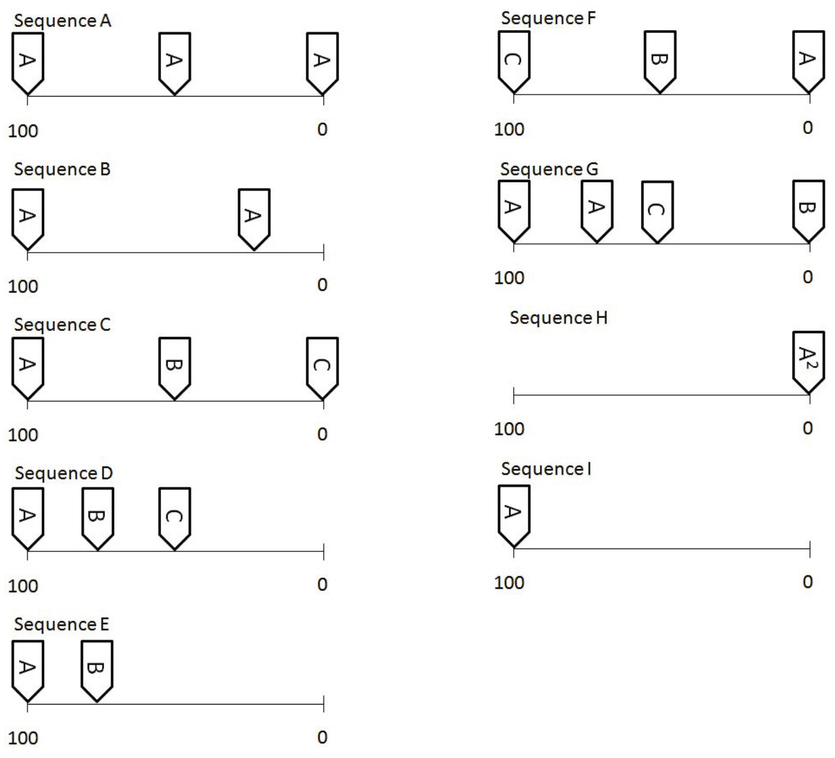


Fig. 1. Illustration of example sequences. The flags indicate events and 100 indicates the point of interest (POI) of the sequence, where the weight of an event is equal to 1.0. Events placed at POI - 100 has a weight of 0.5.

5.2 A Qualitative Analysis of Synthetic Sequences

The result of comparing the event sequences illustrated in figure 1 using the similarity measures presented in section 4.4 and 5.1 are presented in table 1 and table 2. The sequences have been chosen to highlight difficult cases, and the similarity score is multiplied by 100 to indicate percentage. Equation 4 is not considered in this test in order to investigate how order, distance and amount is approximated by the proposed method. In this way, it is easier to compare against the baseline methods Edit Order and Edit Distance, which only apply to simple events.

The comparison of sequence A and B uncover the problem with the cosine similarity as all events in both sequences are of the same type; it measures dimensions and not the amount in each dimension. Because of this, the cosine similarity measure will return 1.0 even though the amount can differ substantially. Both Edit Distance and Edit Order find the sequences quite similar, but penalize the difference in amount more than the other similarity measures.

Table 1. The result of comparing the sequences illustrated in figure 1 using the vector measures in 4.4 and baseline similarity measures presented in section 5.1

	EO	ED	CoSim	RCF*Cos	NRCFCos
sim(A,B)	75.8	90.7	100	95.8	95.8
sim(C,D)	87.7	63.1	99.2	82.3	82.3
sim(C,E)	62.4	94.0	65.9	33.3	60.2
sim(C,F)	50.0	99.0	86.6	55.2	55.2
sim(C,G)	50.2	99.0	90.6	56.4	56.4
sim(G,E)	34.2	100	89.3	35.2	47.1
sim(H,I)	25.7	27.2	100	98.0	98.0

Table 2. This table presents the difference in measured similarity between the baseline similarity measure, Edit Order, and the other similarity measures

EO-Sim	ED	CoSim	RCF*Cos	NRCFCos
EO-sim(A,B)	14.9	24.2	20.0	20.0
EO-sim(C,D)	24.6	11.5	5.4	5.4
EO-sim(C,E)	31.6	3.5	29.1	2.2
EO-sim(C,F)	49.0	36.6	5.2	5.2
EO-sim(C,G)	48.8	40.4	6.2	6.2
EO-sim(G,E)	65.8	55.1	1.0	12.9
EO-sim(H,I)	1.5	74.3	72.3	72.3
Average	33.7	35.1	19.9	17.7
Median	31.6	36.6	6.2	6.2

All the similarity measures find sequence C and D to be quite similar, which makes sense as they contain the same set of events in the same order. The cosine similarity penalize too little while the edit distance penalize too much.

C and E contain different types and different amounts of events. The order of the shared events are similar though. The similarity measures do not agree on this one, as RCF*Cos has the lowest score of 33.3 and Edit Distance has the highest with 94.0. Edit distance clearly does not penalize the missing type enough, while RCF*Cos penalize it too much. Both the RCF part and the Cos part of RCF*Cos penalize the missing dimension. This is why we designed the nice variant, NRCF, of RCR, as it does not penalize missing dimensions.

Sequence F is C inverted, so order and distance are differing while the amount and dimensions are the same. Hence, qualitatively, they are around 50% similar. The Edit Order similarity measure agrees with the qualitative analysis, and so do RCF*Cos and NRCF*Cos. The Edit Distance measure does not find the sequences dissimilar at all, while CoSim finds them quite similar.

The amount of events are differing between C and G and so is the order. Only the dimensions are similar. The events of type A are both located early in sequence G. The results are almost completely the same as for $sim(C, F)$.

Sequence E and G are quite dissimilar in regards to amount, dimensions, order and distance. Edit Order and RCF*Cos recognize this while NRCF*Cos does it to a lesser extent. Both Edit Distance and CoSim are far off.

The two last sequences, H and I, are designed to illustrate a problem with the vector measures. Sequence H contains two events of the same type that are located at the end of the sequence while sequence I contain one event of the same type at the start. In this situation, both Edit Distance and Edit Order find them dissimilar while the vector measures find them completely similar. The vector measures mix the weight with position and do not compare amount directly.

Overall the Edit Order similarity measure is assessed as the most accurate, and this is why table 2 shows the difference between this measure and the others. Because of the last sequence, the average differences of the vector measures increase quite much. However, the results are not completely discouraging.

The assumption that events near the end of the sequence are more important than events farther away does not hold for all applications. As the experiment shows, the vector similarity measures use the weighting function to differentiate between order and distribution of events on a scale, but there are situations in which they are fooled. There are application domains where the disadvantages of the vector similarity measures are advantages. These are domains in which high frequencies of events can out-weight (literally) fewer important ones and where events close to the POI are considered more important.

5.3 Sequence Matching in DrillEdge Using Real-World Data

In this section, we present three experiments that show some properties of the proposed similarity measures and how they perform using real-world data. The performance tests are designed to show how the performance of the different similarity measures degrade in extreme, but realistic, situations where the number of events are quite high. First, we will present DrillEdge briefly.

DrillEdge. Verdande Technology has developed a real-time decision support system, DrillEdge, that supports oil well drilling engineers in making decisions for how to handle problematic situations [5]. The system identifies the problematic situations automatically and provide advices in real-time and is thus providing inexperienced drillig personnel with targeted advices. Complex situations that were comprised of several events, indicating symptoms of larger problems, which occur over a period of time, are represented as cases in a CBR system. Events are recognized using MDTA performed by agents in an agent system. The system has been commercially deployed since 2010 and has analyzed over 200 oil-well drilling operations. In March 2011, Verdande Technology was awarded the Meritous Award for Engineering Excellence for its DrillEdge software platform by E&P Magazine. Currently, it monitors around 35 wells at any given time.

Performance Test #1. In this experiment, 369 input cases created from actual real-time oil well drilling data have been matched against a case base of 19 cases that are created by experts to describe actual oil well drilling situations. The events in the cases have been automatically detected by MDTA agents. The input cases are captured from real-time data. Hence, this test reflects a real-world scenario similar to the ones DrillEdge encounter every day in regard to the content of the cases.

In this experiment, every input case is matched against all cases in the case base. We only measure the time of the actual sequence comparison in order to avoid all overhead of the CBR system. The 19 cases in the case base have a total of 211 events, which gives an average of 17.6 events per case. The case with fewest events have 3 and the one with most events have 46. Furthermore, the 369 input cases contain 7825 events, which gives an average of 21.2 events per case. One case contains only one event while three cases contain 143 events. Overall, each sequence similarity measure have to perform 7011 sequence comparisons. These numbers are presented to let the reader understand the amount of events that represent typical cases in DrillEdge.

Table 3. Total time (in milliseconds) for comparing all cases

Performance test	EO	ED	CoSim	RCF*Cos	NRCFCos
#1	4325	99278	3985	3924	3899
#2	12698	100526	5325	5352	5358

The result from the first performance is documented in the first row of table 3, and the vector sequence similarity measures are comparable when it comes to performance. They are approximately 25 times faster than the Edit Distance implementation while Edit Order comes close with 23 times faster.

Performance Test #2. In order to investigate how the performance of the similarity measures degrade, we have designed another performance test. This section presents the results from this performance test, which uses a real-world data set too. The cases are all captured in an area of a well that have a very high occurrance of events. This is not the typical well, but one that is encountered from time to time. We have captured 10 cases from this section of the well and they have a total of 3883 events, which gives an average of 388.3 events per case. The case with fewest events contains 50 events while the one with most events have 601. All cases are matched against each other, which results in 90 sequences being compared for each similarity measure. Thus, this is not a typical, but still highly probable, situation for the DrillEdge system.

The second row of table 3 shows the result of performance test #2 and that the vector sequence similarity measures are around 18 to 19 times faster than the Edit Distance similarity measure while the Edit Order similarity measure is only 8 times faster. As expected, this indicates that the vector similarity measures degrades more gracefully than the Levenshtein Distance sequence measure.

Applying the Similarity Measures in DrillEdge. Figure 2 shows two case graphs plotted in DrillEdge. A case graph is a graph plotting the similarity score of the comparison between the input case and one of the stored cases against time. As can be seen in the figure, time flows along the x-axis and similarity scores between 0 and 1 is plotted in the y-axis. The topmost similarity graph shows the case graph of a test case in which sequence comparison is performed with the Edit Distance (ED) similarity measure. The lower similarity graph shows a plot of the same case using the $NRCF * Cos$ similarity measure. At the point in time where both similarity measures have a similarity score of 1.0, the case was captured. A closer look reveals a problem with $NRCF * Cos$. ED behaves as expected; when new events occur, the similarity score increases, and when time passes without new events, the similarity score decreases. This is not how $NRCF * Cos$ behaves; it starts increasing even though no new events are identified, which is indicated by a circle in the figure. The reason is that it is relative; it divides the smallest weight of an Event type by the largest one as part of the comparison, so when events are getting so old that they are not inside the sequence anymore, the similarity increases rather than decreases.

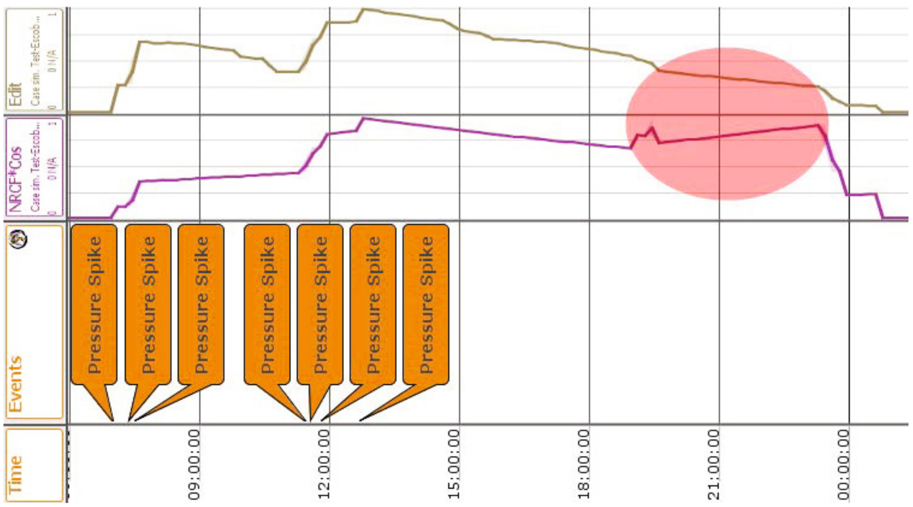


Fig. 2. Events and two case graphs: Same stored case, different similarity measures

6 Conclusion and Future Work

In this paper we have presented an analysis of sequence properties, which lead us to propose a new similarity measure for sequences of complex events based on converting the sequence information to a vector and then comparing the vectors instead of the sequences. The results have shown that the proposed similarity measure are faster than the baseline similarity measures when comparing sequences of large events. Our plan is to investigate whether we can improve the vector similarity measures such that they approximate the more elaborate sequence similarity measure that compares elements directly. We will not consider successful until $sim(H, I)$ and the case graph experiment improve considerably.

References

1. Aamodt, A., Plaza, P.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications 7(1), 39–59 (1994)
2. Adi, A., Etzion, O.: The Situation Manager Rule Language. In: Proceedings of the International Workshop on Rule Markup Languages for Business Rules on the Semantic Web, RuleML (2002)
3. Catley, C., Stratti, H., McGregor, C.: Multi-Dimensional Temporal Abstraction and Data Mining of Medical Time Series Data: Trends and challenges. In: Engineering in Medicine and Biology Society, pp. 4322–4325 (August 2008)
4. Chandy, K.M., Schulte, W.R.: Event Processing - Designing IT Systems for Agile Companies. McGraw-Hill (2010)
5. Gundersen, O.E., Sørmo, F., Aamodt, A., Skalle, P.: A Real-Time Decision Support System for High Cost Oil-Well Drilling Operations. In: Innovative Applications of Artificial Intelligence, IAAI (to appear, 2012)
6. Kavelar, A., Obweger, H., Schiefer, J., Suntinger, M.: Web-Based Decision Making for Complex Event Processing Systems. In: 6th World Congress on Services, pp. 453–458 (July 2010)
7. Lamport, L.: Time, Clocks, and the Ordering of Events in a Distributed System. Commun. ACM 21(7), 558–565 (1978)
8. Levenshtein, V.I.: Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics Doklady 10, 707 (1966)
9. Luckham, D.C., Frasca, B.: Complex Event Processing in Distributed Systems. Tech. rep., Stanford University (1998)
10. Mannila, H., Ronkainen, P.: Similarity of Event Sequences. In: Proceedings of the 4th International Workshop on Temporal Representation and Reasoning. In: TIME. IEEE Computer Society (1997)
11. Margara, A., Cugola, G.: Processing Flows of Information: From Data Stream to Complex Event Processing. ACM Computing Suveys (to appear, 2012)
12. Obweger, H., Suntinger, M., Schiefer, J., Raidl, G.: Similarity Searching in Sequences of Complex Events. In: 2010 Fourth International Conference on Research Challenges in Information Science (RCIS), pp. 631–640 (May 2010)
13. Perrochon, L., Mann, W., Kasriel, S., Luckham, D.C.: Event Mining with Event Processing Networks. In: Zhong, N., Zhou, L. (eds.) PAKDD 1999. LNCS (LNAI), vol. 1574, pp. 474–478. Springer, Heidelberg (1999)
14. Stacey, M., McGregor, C., Tracy, M.: An Architecture for Multi-Dimensional Temporal Abstraction and Its Application to Support Neonatal Intensive Care. In: Engineering in Medicine and Biology Society, pp. 3752–3756 (August 2007)
15. Ukkonen, E.: Algorithms for Approximate String Matching. Information and Control 64(1-3), 100–118 (1985)
16. Wagner, R.A., Fischer, M.J.: The String-to-String Correction Problem. J. ACM 21(1), 168–173 (1974)

Case-Based Project Scheduling

Mario Gómez[*] and Enric Plaza

IIIA - Artificial Intelligence Research Institute, CSIC - Spanish National Research Council, Campus U.A.B. 08193 Bellaterra. Catalonia (Spain)

Abstract. This paper presents a new approach for solving the Resource-Constrained Project Scheduling Problem using Case-Based Reasoning in a constructive way. Given a project to be scheduled our method retrieves similar projects scheduled in the past, selects the most similar project, and reuses as much as possible from the old solution to build a schedule for the project at hand. The result of this process is a partial schedule that is later extended and revised to produce a complete and valid schedule by a modified version of the Serial Schedule Generation Scheme. We present experimental results showing that our approach works well under reasonable assumptions. Finally, we describe several ways to modify our algorithm in the future so as to obtain even better results.

1 Introduction

Scheduling is the process of deciding how to commit limited resources among a variety of activities, subject to a set of constraints. Different scheduling domains pose different types of constraints. Examples of scheduling constraints include deadlines (e.g. activity i must be completed by time t), resource capacities (e.g. there are only four drills), and precedence constraints on the order of tasks (e.g. a piece must be sanded before it is painted). Scheduling is a class of optimization problems in which one seeks to minimize or maximize a function (e.g. minimizing the time span) while meeting some constraints. The Resource Constrained Project Scheduling Problem (RCPSP) is a class of scheduling problems consisting of a network of activities, precedence relationships and resource constraints (for general surveys on the subject see for example [1,2].

[3] showed that the RCPSP, as a generalization of the classical *job-shop scheduling* problem, belongs to the class of NP-hard optimization problems. Due to this complexity, proven optimal solutions have been computed only for very small problems; in other words, exact scheduling methods are mainly for generating benchmark solutions, and not for real-world applications. Approximate methods and heuristics are thus required for solving large scheduling problems as they usually appear in the real-world.

Synthetic tasks like planning and scheduling can be approached either by *constructive methods* or by *repair-based methods*. Constructive methods start with a description of requirements to build a completely new solution, while repair-based methods take a previous solution and some new requirements and modify

[*] Current affiliation is Departamento de Sistemas Informáticos y Computación, Universitat Politècnica de València, Camino de Vera s/n., 46022 Valencia, Spain.

the original solution to meet the new requirements. In this paper, we present a Case-Based Reasoning (CBR) framework for solving the Resource-Constrained Project Scheduling Problem (RCPSP) in a constructive way. This framework follows the general cycle of the CBR process: It begins with a retrieval step, which obtains past scheduling problems that are similar to the current problem; then, part of the solution to the most similar problem is reused and adapted to build a partial solution to the new problem; finally, this partial solution is extended and revised to produce a complete and valid solution by applying a modified version of the Serial Schedule Generation Scheme (SSGS).

The paper is organized as follows: Section 2 describes the RCPSP and the SSGS; Section 3 describes our framework for solving the RCPSP using CBR; Section 4 evaluates the proposed framework through some experiments; Section 5 surveys related work; finally, Section 6 draws some conclusions and suggests future work.

2 The Resource-Constrained Project Scheduling Problem

In project scheduling, a *project* consists of a set of *activities* that have to be carried out in order to finish the project. Every activity has certain *duration.* Some of these activities require that some other activities are already finished (precedence relationships), and some of them require the availability of certain *resources.* The outcome of project scheduling is a specification of the start(or finish) times of every activity —a schedule— so that the project can be completed and all the project constraints are met. Optimization functions are desirable to select the best among all possible schedules; for example, the *minimum time span* function is used to minimize projects completion.

Project scheduling is easy if only precedence relationships constrain the project: the Critical Path Method [4] provides allowable time windows for the activities of a project in polynomial time. However, scheduling becomes much harder when required resources are available in limited amounts, so that some tasks compete for the same resources. This class of problems is referred to as the Resource-Constrained Project Scheduling Problem (RCPSP), and is known to be NP-hard [3].

The single-mode Resource-Constrained Project Scheduling Problem for renewable resources can be defined as follows:

A **project** is a tuple $\langle \mathcal{J}, \mathcal{K} \rangle$ consisting of:

– A set $\mathcal{J} = \{0, 1, ..., J, J+1\}$ of **activities** to be executed, where J is the real number of activities, and activities 0 and $J+1$ are dummy activities used to represent the *project start* and *project end* respectively.
– A set $\mathcal{K} = 1, ..., K$ of **resources** available, where K is the number of constrained resources (non constrained resources are ignored).

For every **activity** $j \in \mathcal{J}$ there is a tuple $\langle d_j, \mathcal{P}_j, \mathcal{R}_j \rangle$ consisting of:

– A non-preemptable **duration** d_j, which specifies the amount of time required to complete the activity. Non preemptable means once started an activity cannot stop before completion.

- A set $\mathcal{P}_j \subseteq (\mathcal{J} - j)$ of **predecessors**. Activity i is predecessor of activity j $(i \in \mathcal{P}_j)$ if the former has to be finished before the later can be started.
- A vector $\mathcal{R}_j = \langle r_{j,1}, ..., r_{j,K} \rangle$ of **resource requirements**. Each requirement $r_{j,k}$ represents the capacity of resource k required by activity j to be executed.

Every **resource** $k \in \mathcal{K}$ has a limited resource capacity of R_k. This capacity is reduced when being used by an activity, but it is restored when the activity ends.

The start and end activities are not real activities of the project: they do not take up time and do not require resources; however, they are included in the model for convenience, since they are useful for describing and implementing algorithms (actually, this has become a *de facto* standard representation).

The parameters d_j, $r_{j,k}$, and R_k are assumed to be deterministic; they remain the same for the entire duration of the project. These parameters are expressed discretely, using positive integers.

The objective of the RCPSP is to satisfy precedence- and resource-feasible completion times for all activities such that the timespan of the project is minimized. Since activities have deterministic durations, we may indistinctly use start times or finish times. Let f_j denote the finish time of activity j. A schedule $\mathcal{S}$ is given by a vector of finish times $\langle F_1, F_2, ..., F_J \rangle$. Let $\mathcal{A}(t) = j \in \mathcal{J} | F_j - d_j \leq t < F_j$ be the set of activities which are being processed (active) at time instant t. We can now specify the conceptual decision model (1) – (4) [5] for the RCPSP.

$$Min\ F_{n+1} \tag{1}$$

$$F_h \leq F_j - d_j \qquad j = 1, ..., J+1;\ h \in P_j \tag{2}$$

$$\textstyle\sum_{j \in \mathcal{A}(t)} r_{j,k} \leq R_k \qquad k \in K;\ t \geq 0;\ r_{j,k} \leq R_k \tag{3}$$

$$F_j \geq 0 \qquad j = 1, ..., J+1 \tag{4}$$

The objective function (1) minimizes the finish time of the end activity and thus the make-span of the project; constraints (2) enforce the precedence relations between activities; and constraints (3) limit for each resource type k and each time instant t that the resource demand of the activities which are currently processed does not exceed the capacity. Finally, (4) defines the decision variables.

3 Case-Based Project Scheduling (CBPS)

3.1 Preliminary Notions: The Serial Schedule Generation Scheme

Most heuristic methods to solve the RCPSP are based on a *schedule generation scheme* (SGS), which starts from scratch and builds a feasible schedule by stepwise extension of a partial schedule. There are two different schedule generation schemes: Serial and Parallel. The Serial SGS (SSGS) performs activity increments (a new activity is scheduled each step), while the Parallel SGS (PSGS) performs time increments (time is incremented by one each step).

The SSGS consists of $g = 1, ..., J$ stages. At each stage one activity is selected and scheduled at the earliest precedence- and resource-feasible completion time. Associated with each stage g are two disjoint activity sets. The scheduled set $\mathcal{S}_g$ comprises the activities which have been already scheduled, the eligible set $\mathcal{D}_g$ comprises all activities which are eligible for scheduling. Let $\tilde{R}_k(t) = R_k - \sum_{j \in \mathcal{A}(t)} r_{j,k}$ be the remaining capacity of resource type k at time instant t and let $F_g = \{F_j | j \in S_g\}$ be the set of all finish times. Further let $\mathcal{D}_g = \{j \in \mathcal{J} \setminus \mathcal{S}_g \mid \mathcal{P}_j \subseteq \mathcal{S}_g\}$ the set of eligible activities. Given these definitions, we can now describe the SSGS algorithm as follows.]

Algorithm 1. The SSGS algorithm

```
 1: initialization: $F_0 = 0, \mathcal{S}_0 = 0$
 2: for $g = 1$ to $J$ do
 3:     Compute $D_g, F_g, \tilde{R}_k(t)(k \in K; t \in F_g)$
 4:     Select one $j \in D_g$
 5:     $EF_j = max_{h \in P_j}\{F_h\} + d_j$
 6:     $F_j = min\{t \in [EF_j - d_j, LF_j - d_j] \cap F_g|$
 7:     $r_{j,k} \leq \tilde{R}_k(\tau), k \in K, \tau \in [t, t + d_j[ \cap F_g\} + d_j$
 8:     $\mathcal{S}_g = \mathcal{S}_{g-1} \cup \{j\}$
 9:     $F_{n+1} = max_{h \in \mathcal{P}_{n+1}}\{F_h\}$
10: end for
11: return $F$
```

The initialization assigns the start activity $j = 0$ a completion time of 0 and puts it into the partial schedule. At the beginning of each step g the decision set $\mathcal{D}_g$, the set of finish times $\mathcal{F}_g$ and the remaining capacities $\tilde{R}_k(t)$ at the finish times $t \in \mathcal{F}_g$ are computed. Afterwards, one activity j is selected from the decision set. The finish time of j is calculated by first determining the earliest precedence-feasible finish time EF_j and then calculating the earliest resource-feasible finish time F_j within $[EF_j, LF_j]$, where LF_j denotes the latest finish time as calculated by backward recursion [6] from an upper bound of the project finish time T.

The SSGS always generates feasible schedules which are optimal for the resource-unconstrained scheduling problem (1), (2), (4). Time complexity of SSGS is $O(n^2 \cdot K)$ [7]. The quality of the solutions obtained by the SSGS highly depends on the mechanism used to select the next activity $j \in D_g$ to be scheduled. Typically, this mechanism is based on the use of priority rules. A *priority rule* is a mapping which assigns each activity j in the decision set D_g a priority value $v(j)$ and an objective stating whether the activity with the minimum or the maximum value is preferred. In case of ties, one or several tie breaking rules have to be employed (e.g. choose the activity with the smallest activity label). For a description of different priority rules see for example [8].

A useful way of representing a schedule is an *activity list*: a list $\lambda = \langle j_1, j_2, ..., j_n]$ of activities that are precedence-feasible, i.e. we have $\mathcal{P}_{j_g} \subseteq \{j_1, ..., j_{g-1}\}$ where $g \in 1, 2, ..., n$. An activity list containing all the activities of a project encodes a schedule for that project. There is a version of the SSGS for activity lists, which simply selects the next activity to be scheduled each step as the next activity in the activity list. Formally, if we use j_g to denote the g^{th} activity of λ, then in the SSGS (Algorithm 1) we simply have to replace 'Select one $j \in D_g$' (line 4) with 'Select $j = j_g$', and remove D_g from the initialization step [9].

3.2 Overview of the Framework

Our proposal to solve the RCPSP uses the SSGS as the underlying mechanism to ensure that the solutions obtained meet the precedence and resource capacity constraints. In particular, we introduce the notion of *generalized activity list* (GAL), which results of removing the precedence condition from the definition of an activity list. A GAL also encodes a solution schedule, one which can be obtained by a modified version of the SSGS: the *SSGS for generalized activity lists*, abbreviated SSGS-GAL, and described later as Algorithm 4.

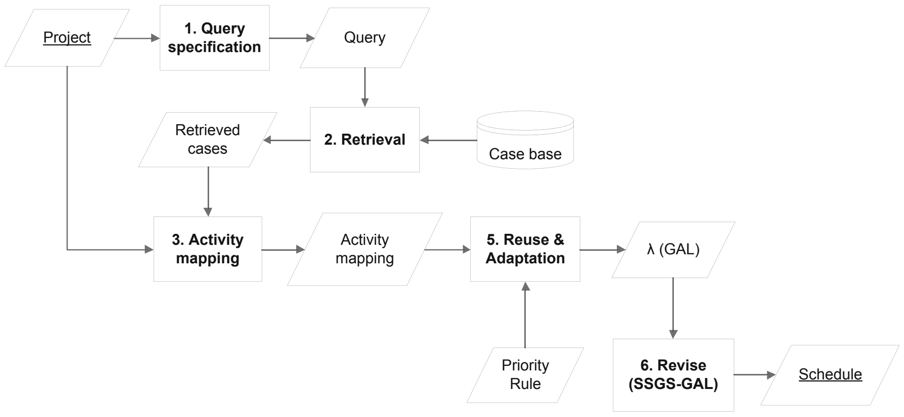


Fig. 1. Process of building a case-based activity list

Figure 1 shows a flowchart of the overall process. The process starts with a project to be scheduled, a priority rule and a case-base, and produces a complete and valid schedule for the project. The *case-base* contain resource-constrained projects and activity lists that solve those projects. This framework contains the usual stages in a CBR system with one remarkable difference : the retrieval step is divided in two phases. During the retrieval step some cases are retrieved using a standard similarity measure computed over global project properties, as usual, but there is an additional retrieval step (the activity mapping) that uses a graph-based similarity measure to compare projects at the structural level. Next sections describe the entire method step by step.

3.3 Query Specification and Case Retrieval

Query specification and case retrieval are tightly related, the former analyses the target project to find the values of some properties that are then used by the later to retrieve cases containing projects that are similar the target project. These properties are specified as attribute-value pairs, a representation that supports a wide range of well known and very fast similarity measures and retrieval algorithms. In particular we use the classical k-Nearest Neighbour (k-NN) algorithm and compute project similarity as a weighted average of similarity measures for certain global project properties summarized below.

Network complexity (NC): the average number of (non redundant) arcs per node (including super-source and sink)

Resource Factor (RF): reflects the average portion of resources requested per activity. $RF = 1$ means each activity requests all resources, while $RF = 0$ indicates the opposite, which corresponds to the unconstrained case. For the single mode RF is calculated as follows:

$RF = \frac{1}{J}\frac{1}{|\mathcal{K}|}\sum_{j=1}^{J}\sum_{k\in\mathcal{K}} Q_{j,k}; Q_{j,k} = \{1 \text{ if } r_{jk} > 0 \text{ ; } 0 \text{ otherwise}\}$

Resource Strength (RS): aims at expressing how difficult it is to satisfy the resource constrains of a project. Several ways to compute RS have been proposed. We have chosen the one used to generate the PSPLIB datasets, because it seems to capture better this notion of difficulty, and PSPLIB is the preferred benchmark to compare scheduling algorithms. RS is a scaling parameter in the interval $[0, 1]$ used to compute resource availability (R_k) as a convex combination of a minimum and a maximum level, R_k^{min} and R_k^{max} respectively. For the single mode, R_k is defined as follows:

$$R_k = R_k^{min} + round(RS \times (R_k^{max} - R_k^{min}))$$

There is no exact procedure to isolate RS in the former equation, but it can be approximated by *numerical methods*: computing lower and upper bounds for RS and interpolating between them.

3.4 Activity Mapping

The previous retrieval step obtains a list of cases containing similar projects, where project similarity is based on global project properties, which is appropriate to find a set of projects that are potential candidates for reuse. But in order to select the best candidate for reuse, we have to compare projects at a deeper level, considering the structural components of a project, that is, the network of activities, precedence relations and resource constraints.

In order to reuse a solution, we also need a mapping of activities between the project to be scheduled and the projects retrieved from the case base. Actually, computing the similarity and obtaining the activity mappings are part of the same process, since computing the structural similarity involves maximizing the set of activity mappings.

Mathematically, a project can be described by a DAG, where nodes correspond to activities, and edges correspond to precedence relations; therefore, computing the structural similarity of two projects becomes a problem of computing the similarity of two graphs. Typically, graph similarity algorithms are variations or derivatives of the *maximum common subgraph isomorphism* (MCS) problem, which takes two graphs G1 and G2 as input, and aims at finding the largest induced subgraph of G1 isomorphic to a subgraph of G2. Exact MCS algorithms compute several alternative mappings of nodes in G1 that are isomorphic to nodes in G2, and return the maximal set of node mappings. If we use nodes to represent project activities, then the maximal node mapping returned by an MCS algorithm is precisely the activity mapping we are seeking.

The MCS isomorphism problem is NP-hard, so exact algorithms do not scale well with the size of the problem. There are also approximate algorithms reported in the literature and available as free software libraries, like Absurdist II [10]. Nevertheless, our experience with this algorithm has not been satisfactory, since the quality of the solutions obtained was rather poor.

Everything considered, we have developed our own algorithm to compute the similarity of two projects. This is an approximate algorithm in two ways: on the one hand, it does not enforce the isomorphism condition to map nodes; on the other hand, it does not compute all possible node mappings. In particular, our algorithm reduces the number of potential activity mappings by using heuristics based on some of the features making a project a special case of a DAG, namely: (a) any project has a single start node and a single end node (the dummy start/end activities), and (b) activities are pre-sorted taking into account precedence relations: activity j can not be a predecessor of an activity k given $k < j$.

Let us introduce some notation and definitions needed to describe the activity mapping algorithm.

$Q = \langle \mathcal{J}^Q, \mathcal{K}^Q \rangle$ is the target project, the one to be scheduled.
$P = \langle \mathcal{J}^P, \mathcal{K}^P \rangle$ is a project retrieved from the case base.
$S \in [0, 1]$ is the similarity between projects P and Q
M and M' are sets of activity mappings between $\mathcal{J}^P$ and $\mathcal{J}^Q$. An activity mapping is a pair of activities (j, j'), where $j \in \mathcal{J}^P$ and $j' \in \mathcal{J}^Q$.
$sim(j, j')$ is a function that returns the similarity of $j \in \mathcal{J}^P$ and $j' \in \mathcal{J}^Q$.

Algorithm 2 is a pseudocode description of the algorithm for computing the similarity between two projects and obtaining a mapping of activities.

First (lines 2 to 6), the similarity of every possible pair of activities is computed, and the pairs with a similarity greater than certain threshold α are added to the set of potential activity mappings M'. Activity similarity measures are described later.

Next (lines 7 to 13), for each activity j' in Q, a single activity mapping (h, j') from M' is selected and included in the final set M of mappings. The selection of the final activity mappings uses a simple heuristic: the mappings comprising the most similar activities are preferred, and in case of ties, the mapping whose left activity h has a lower identifier is selected. This is an approximate procedure, since not all possible combinations of activity mappings are considered, but only one.

Finally (lines 14 to 17), project similarity S is computed as the sum of similarities for all activities in the final set of activity mappings, divided by the number of activities in the target project ($\mathcal{J}^Q$). Since the similarity between activities is a value in [0,1], and the maximum number of activity mappings is precisely $\mathcal{J}^Q$, then project similarity is also a value in [0,1].

We still have to define a similarity measure for activities. The results of the algorithm and its complexity can be strongly influenced by the way we compute similarity between activities. There are two aspects to take into account to assess the similarity between two tasks: resource requirements and precedence relations.

Algorithm 2. Activity Mapping Algorithm

```
1: initialization: $M = \emptyset$; $M' = \emptyset$; $S = 0$
2: for all $j \in \mathcal{J}^P$, $j' \in \mathcal{J}^Q$ do
3:     if $sim(j, j') > \alpha$ then
4:         $M' = M' \cup (j, j')$
5:     end if
6: end for
7: for all $(j, j') \in M'$ do
8:     if $\nexists h : (h, j') \in M'$ then
9:         $M = M \cup (j, j')$
10:    else if $sim(j, j') > sim(h, j') \vee (sim(j, j') = sim(h, j') \wedge j < h)$ then
11:        $M = (M \setminus (h, j')) \cup (j, j')$
12:    end if
13: end for
14: for all $(j, j') \in M$ do
15:     $S = S + sim(j, j')$
16: end for
17: $S = S \ / \ |\mathcal{J}^Q|$
18: return $M, S$
```

We have adopted a binary similarity approach to compute activity similarity, that is, activity similarity can take only two values: 0 or 1. Conceptually, we define activity mapping using equivalence relations, as follows:

Two activities are *resource-equivalent* if there is a mapping of resources such that all pairs of resource requirements are *capacity-proportional*.

Two resource requirements are *capacity-proportional* if they amount for the same proportion of resource capacity.

Two activities are *equivalent* if they are resource-equivalent and there is a mapping of successors such that all pairs of successors are resource-equivalent.

Now we can define the similarity of two tasks j and j':

(a) $sim(j, j') = 1 \iff j$ and j' are *equivalent*; otherwise $sim(j, j') = 0$

The similarities obtained by the mapping algorithm are used to select the case with the highest similarity S, which is referred as the **best case** henceforth. The mapping M of activities between the target project and the project included in the best case are passed to the next step to produce an activity list, as described in the following section.

3.5 Reuse and Adaptation

The previous step obtains a mapping of activities between the target project and the project in the best case. The ordering of mapped activities in the solution of the best case provides a tentative ordering of activities in the new project to build the required schedule. In order to produce a complete activity list we have conceived a method that uses both the schedule from the best case and a standard priority rule, as described in Algorithm 3.

$\mathcal{J}^Q$ and $\mathcal{J}^P$ represent the sets of activities in the target project Q and the project P stored in the best case respectively. Let M be the set of activity

Algorithm 3. Reuse and Adaptation Algorithm

```
1: initialization: $\theta = \emptyset$
2: for all $j \in \mathcal{J}^Q$ do
3:     if $\exists (j', j) \in M : j' \in \mathcal{J}^P$ then
4:         $\theta = \theta \cup j$
5:     end if
6: end for
7: $\lambda = SORT(\mathcal{J}^Q, \theta, \psi)$
8: return $\lambda$
```

mappings between $\mathcal{J}^Q$ and $\mathcal{J}^P$ and let ψ denote a standard priority rule. First the algorithm computes a partial activity list θ (lines 2 to 6) by traversing $\mathcal{J}^Q$ and adding to θ those activities that are mapped to activities in $\mathcal{J}^P$. Finally, the algorithm calls the $SORT$ function (line 7) to produce λ, a complete activity list that encodes a solution for Q. $SORT$ takes $\mathcal{J}^Q$, θ and ψ as inputs, and obtains λ as output by applying the following order relation on $\mathcal{J}^Q$: if two activities $i, j \in \mathcal{J}^Q$ are both included in θ then their order is kept the same in λ, otherwise their order is obtained by applying priority rule ψ.

However, being the mapping of activities approximate, the ordering of activities obtained from the best case's solution does not guarantees the satisfaction of Q's constraints, so θ is actually a partial GAL. As a consequence, the resulting list λ is also a GAL.

3.6 Revise

The reuse step returns a partial GAL (λ), so the precedence and resource-capacity conditions are not guaranteed. As a consequence, we can not use the standard SSGS for activity lists to obtain the solution encoded by λ, since it might produce an incorrect schedule. In order to obtain a valid schedule we have developed a new version of the SSGS for generalized activity lists, abbreviated SSGS-GAL, which is described in Algorithm 4, where $g(j) \in \{1, 2, ..., J\}$ denotes the position of j in λ. A solution obtained by the SSGS-GAL is guaranteed to be correct even though the activity list used as input is not, since the project constraints are enforced by the SSGS-GAL when computing the decision set D_g (Line 3).

Algorithm 4. The SSGS algorithm for generalized activity lists (SSGS-GAL)

```
1: initialization: $F_0 = 0, \mathcal{S}_0 = 0$
2: for $g = 1$ to $n$ do
3:     Compute $D_g, F_g, \tilde{R}_k(t)(k \in K; t \in F_g)$
4:     Select $j \in D_g | \forall j' \in D_g : g(j) > g(j')$
5:     $EF_j = max_{h \in P_j}\{F_h\} + d_j$
6:     $F_j = min\{t \in [EF_j - d_j, LF_j - d_j] \cap F_g | r_{j,k} \leq \tilde{R}_k(\tau), k \in K, \tau \in [t, t + d_j[ \cap F_g\} + d_j$
7:     $\mathcal{S}_g = \mathcal{S}_{g-1} \cup \{j\}$
8:     $F_{n+1} = max_{h \in \mathcal{P}_{n+1}}\{F_h\}$
9: end for
10: return $F$
```

The SSGS-GAL, as the SSGS, needs a priority rule to compute the decision set (D_g).

4 Experimental Evaluation

In order to evaluate a learning approach such as CBR, one should demonstrate its ability to generalize, that is, the learning method should be able to solve new problems that were not included in the learning dataset. In order to have a pool of datasets for learning and testing purposes we have taken the well known Patterson's dataset [11] , and we have used it as a source to create new datasets by randomly modifying its problems.

In particular, we have created two groups of new datasets: On the one hand, **PatI5** comprises 3 data sets obtained by removing between 1 and 5 random activities from Patterson's original problems. On the other hand, **PatD** comprises a sequence of 9 interdependent data sets; the first data set in the sequence (PatD1) results of removing 1 random job from projects in the Patterson's data set, the next data set (PatD2) results of removing 1 random activity from the previous data set in the sequence (thus accumulates 2 removals from the original data set), and so on.

In order to compare different methods, we have first obtained the optimal solutions for the new data sets using the *branch and bound* algorithm proposed by [12]; that way, we can compare algorithms by comparing their solutions with the optimal solution.

We have performed two groups of experiments: In the first group, we assess the impact of k and the size of the case base on the performance of the Case-Based Project Scheduling (CBPS) algorithm. In the second group we assess how the differences between the learning and the test data sets impact the performance of the CBPS algorithm.

In our experiments, we use the *average relative error* to measure the performance of a scheduling algorithm. The relative error of a single project is computed as $(timespan - optimalTimespan)/optimalTimespan$, thus the lower the error the better. In all the experiments that follow we include a comparison with a single-pass SSGS using LST (Latest Start Time) as the priority rule. Correspondingly we have also used LST as the priority rule ψ to complete the partial activity list obtained from the best case, as described in Algorithm 3.

Impact of *k* and the Size of the Case Base. The aim here is to study how the size of the case base and the number of cases retrieved (k) influence the performance of the CBPS algorithm. We expect CBPS to obtain better solutions as the size of the case base and the number of cases retrieved increase.

Table 1 shows the results for the PatI5 data sets using different case bases, where CBPS:k denotes the CBPS algorithm with k being the number of cases retrieved. data sets with suffix 'a' and 'b' used for learning, while the one with suffix 'c' is used for testing purposes. CBPS obtains better results than LST in all the scenarios, and the results improve consistently with the number of cases retrieved . First and second columns show results when using PatI5a and Pat5b respectively to build the case base. Since they have the same size, results

Table 1. Average relative errors for PaIt5 data sets using different case bases

CB / Algor.	PatI5a	PatI5b	PatI5a ∪ PatI5b	PatI5c
LST	0,090	0,090	0,090	0,090
CBPS:1	0,083	0,085	0,078	0,006
CBPS:3	0,074	0,084	0,073	0,000
CBPS:5	0,073	0,082	0,069	0,000
CBPS:10	0,068	0,073	0,064	0,000
CBPS:20	0,066	0,070	0,058	0,000

are quite similar. Third column shows results when using PatI5a and PatI5b together for the case base; as expected, we obtain better results due to the increase in the size of the case base. The last column is provided to check out that the algorithm works properly: since we use the same data set for learning and testing, the CBPS algorithm should be able to retrieve exactly the project being solved, and thus it should provide the optimal solution, i.e. error would be zero. As observed, CBPS is able to find the optimal solutions when $k \geq 3$, and the error when retrieving a single case ($k = 1$) as as low as 0.06. These result suggest that the global project characterization (network complexity, resource factor and resource strength) is a good discriminant between projects.

Impact of the Degree of Difference between the Learning Data Sets and the Test Data Sets. The experiments reported here study the impact of the degree or amount of difference between the learning data set and test data set. We expect CBPS to obtain worse results as the amount of difference increases.

Table 2. Average relative errors for PatD data sets, with Pat5 as the case-base

Algorithm	Pat1	Pat2	Pat3	Pat4	**Pat5**	Pat6	Pat7	Pat8	Pat9
LST	0,096	0,096	0,099	0,091	0,093	0,085	0,080	0,079	0,071
CBPS:1	0,097	0,099	0,085	0,066	0,006	0,047	0,068	0,067	0,059
CBPS:3	0,095	0,096	0,076	0,052	0,000	0,033	0,054	0,059	0,071
CBPS:5	0,091	0,095	0,073	0,047	0,000	0,030	0,052	0,060	0,071
CBPS:10	0,095	0,091	0,075	0,046	0,000	0,025	0,047	0,050	0,062
CBPS:20	0,092	0,085	0,069	0,046	0,000	0,027	0,037	0,043	0,058

Table 2 shows results when solving PatD data sets using Pat5 as the case base. As expected, Pat5 is solved optimally for a small k (when $k \geq 3$). In Pat1,...,Pat4 the projects to be solved have more activities than the projects in the case-base (Pat1 has 4 more activities, Pat2 has 3 more activities, etc.), so only a portion of the activities could potentially be mapped to activities in a retrieved project. The opposite happens to projects in Pat6 to Pat9, which have fewer activities than projects in the case base. Therefore we expect CBPS to behave much better when solving the later than the former; which is actually confirmed by the experiments. In all the tests CBPS obtained better solutions than LST. Notice that the fewer the differences between the test and learning data sets, the comparatively better our algorithm behaves.

5 Related Work

The first system experimentally evaluated is CABINS [13], a framework for iterative schedule repair based on user optimization preferences. CABINS uses a a case-based approach for capturing human experts' preferential criteria about scheduling quality and control knowledge to speed up schedule revision. Through iterative schedule repair, CABINS improves the quality of sub-optimal schedules by using past repair experiences for (1) repair action selection, (2) evaluation of intermediate repair results, and (3) recovery from revision failures. Extensive experimentation on a job-shop scheduling domain shows an improvement in the efficiency of the revision process while preserving the quality of the resultant schedule. The first constructive CBR approaches for scheduling including experimental evaluation were reported by [14,15]. The first paper describes a CBR method to solve the Traveling Salesman Problem (TSP) by reusing and adapting complete solutions. The proposed method, called T-CBR, was compared against Simulated Annealing (SA) and a Myopic algorithm, concluding that SA produces the best solutions but takes the longest, the Myopic algorithm is the faster algorithm but produces the worst solutions, and T-CBR produces medium quality solutions considerably faster than SA. The second paper compares two CBR methods for solving the *job-shop* scheduling problem with a single machine and sequence dependent setup times. CBR has also been proposed to deal with *workforce rostering* problems [16,17]. Cases represent rostering constraints and generalized patterns of workforce allocation (shift patterns) that satisfy those constraints. In this proposal CBR is used in a constructive way to build up a schedule that is then fixed using rules to remove constraint violations. Fixes are expressed as ordered series of shift swapping rules. This separation between the roster generation using CBR and the fix mechanisms is well suited to perform reactive scheduling with no extra effort: the very same mechanisms used to fix a new roster can be used to fix rosters that have subsequently become broken. Typically, scheduling problems the proposals introduced above adopt the classical representation of cases as lists of attribute-value pairs. A radically different approach using graphs was proposed in [18] to deal with timetabling problems. Experimental results showed the effectiveness of the retrieval and adaptation steps. This method was further developed and evaluated across a wider range of problems [19]. Later, a multiple-retrieval approach was proposed that partitions a large problem into small solvable sub-problems [20].

CABAROST (CAse-BAsed Rostering) is a repair-based method for solving nurse rostering problems [21,22]. CABAROST retrieves previously encountered constraint violations and reuses their repairs to solve new rostering problems.

All in all, research on the application of Case-Based Reasoning (CBR) to scheduling is scarce and limited in scope. On the one hand, the application domain has been limited to very specific forms of scheduling, such as the job-shop problem, rostering problems, and time-tabling problems. On the other hand, really few systems have been evaluated experimentally. Finally, in several proposals CBR is not the actual scheduling method, but just a complementary tool used either to select the actual scheduling method or to configure it (for example to choose one amongst a number of alternative heuristic rules).

6 Conclusions and Future Work

Compared with other attempts to apply CBR to scheduling problems, we have addressed a more general class of problems. In fact, a wide area of combinatorial problems are known to be special cases of the RCPSP, including: the *two-dimensional cutting stock* problem, the *bin packing* problem, and production scheduling problems such as the *job-shop* scheduling problem, the *flow-shop* problem and the *open-shop* problem.

In this paper, we describe a Case-Based Reasoning (CBR) framework to solve the Resource-Constrained Project Scheduling Problem (RCPSP) in a constructive way, using past schedules as a starting point to build new schedules. The aim is not to compete with the best results reported in the literature by special purpose specific approaches, but to generate competitive results across a wider range of problems: for example, problems with poorly defined domains or high degrees of uncertainty. Note that in general real scheduling problems are much more complex and uncertain than the mathematical models used in academic research. The more complex and uncertain the domain, the more appropriate CBR would be compared with theoretical approaches. Besides, a CBR approach such as CBPS will be able to adapt and improve results over time by storing new cases in the case base and removing old ones. The experience obtained from real world examples will probably capture complexities of the domain that will be much harder to represent in a theoretical model.

In the experiments performed so far we have consistently obtained better results than using the best simple-pass priority-rule based heuristics. These results demonstrate the feasibility of our proposal and indicate that under reasonable assumptions this approach may become a useful tool to trade-off quality of the solutions and efficiency, which is a requirement to address real world problems.

Our framework extends the well-known Serial Schedule Generation Scheme by introducing the SSGS for generalized activity lists (SSGS-GAL). By using SSGS as the underlying validation mechanism we ensure that only valid solutions are generated, together with other interesting properties. Our method is compatible with the standard view of heuristic priority rules as functions that compute a priority value for all activities of a project. This approach supports the use of single-pass SSGS methods as well as most multi-pass SSGS methods, including *multi-priority rule* methods, *forward-backward* scheduling methods, and *sampling* methods. Furthermore, our framework is flexible enough to support a wide range of similarity measures to compare projects, which opens many possibilities to tune up the algorithm and adapt it to different circumstances.

A limitation of our approach is the fact that in order to work properly, CBR requires good cases, that is, cases that are representative of the application domain, and cases that have good solutions. In the real world, sometimes there may be a repository of past projects which can be used to generate an initial case base, but that will not be true in general. When no previous experience is available, one has to generate new cases using another (non-CBR) method, and therefore, the quality of the solutions obtained by CBR would depend upon the quality of the solutions obtained by the non-CBR method. In other words, if there is no previous experience it would probably be better to use a non-CBR approach, at least until gaining experience.

There is a number of ways to further expand our framework. Firstly, there is plenty of room to experiment with different similarity metrics to retrieve projects. Secondly, it would be possible to experiment with different scheduling algorithms, like the Parallel Scheduling Generation Scheme and some multi-pass methods (eg. sampling and multi-rule methods). Another interesting extension would be the building of new schedules by combining portions of multiple solutions, instead of reusing a single solution from the case-base.

Acknowledgments. Support for this work came from project Next-CBR MICIN TIN2009-13692-C03-01.

References

1. Herroelen, W., De Reyck, B., Demeulemeester, E.: Resource-constrained project scheduling: a survey of recent developments. Computers and Operations Research 25(4), 279–302 (1998)
2. Brucker, P., Drexl, A., Möhring, R., Neumann, K., Pesch, E.: Resource-constrained project scheduling: Notation, classification, models, and methods. European Journal of Operational Research 112(1), 3–41 (1999)
3. Blazewicz, J., Lenstra, J., Kan, A.R.: Scheduling subject to resource constraints classification and complexity. Discrete Applied Mathematics 5, 11–24 (1983)
4. Kelley, J.: The Critical-Path Method: Resources Planning and Scheduling. In: Industrial Scheduling, pp. 347–365 (1963)
5. Christofides, N., Alvarez-Valdés, R., Tamarit, J.: Project scheduling with resource constraints: A branch and bound approach. European Journal of Operational Research 29(3), 262–273 (1987)
6. Elmaghraby, S.: Activity networks: Project planning and control by network models. John Wiley & Sons, New York (1977)
7. Pinson, E., Prins, C., Rullier, F.: Using tabu search for solving the resource-constrained project scheduling problem. In: Proceedings of the 4th International Workshop on Project Management and Scheduling, pp. 102–106 (1994)
8. Alvarez-Valdés, R., Tamarit, J.: Heuristic algorithms for resource-constrained project scheduling. In: Advances in Project Scheduling, pp. 113–134. Elsevier (1989)
9. Kolisch, R., Hartmann, S.: Heuristic Algorithms for solving the resource-constrained project scheduling problem: classification and computational analysis. In: Handbook of Recent Advances in Project Scheduling, pp. 147–178. Kluwer Academic Publishers (1999)
10. Feng, Y., Goldstone, R.L., Menkov, V.: A graph matching algorithm and its application to conceptual system translation. International Journal on Artificial Intelligence Tools 14, 77–79 (2004)
11. Patterson, J.H.: A comparison of exact procedures for solving the multiple constrained resource project scheduling problem. Management Science, 854–867 (1984)
12. Sprecher, A., Drexl, A.: Multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm. European Journal of Operational Research 107(2), 431–450 (1998)
13. Miyashita, K., Sycara, K.P.: Cabins: A framework of knowledge acquisition and iterative revision for schedule improvement and reactive repair. Artificial Intelligence 76(1-2), 377–426 (1995)

14. Cunningham, P., Smyth, B., Hurley, N.: On the Use of CBR in Optimisation Problems Such as the TSP. In: Aamodt, A., Veloso, M.M. (eds.) ICCBR 1995. LNCS, vol. 1010, pp. 401–410. Springer, Heidelberg (1995)
15. Cunningham, P., Smyth, B.: Case-based reasoning in scheduling: Reusing solution components. International Journal of Production Research 35(11), 2947–2962 (1997)
16. Scott, S., Simpson, R., Ward, R.: Combining case-based reasoning and constraint logic programming techniques for packaged nurse rostering systems. In: Proceedings of the Third UK Case-Based Reasoning Workshop (1997)
17. Scott, S., Simpson, R.M.: Case-Bases Incorporating Scheduling Constraint Dimensions - Experiences in Nurse Rostering. In: Smyth, B., Cunningham, P. (eds.) EWCBR 1998. LNCS (LNAI), vol. 1488, pp. 392–401. Springer, Heidelberg (1998)
18. Burke, E.K., MacCarthy, B., Petrovica, S., Qu, R.: Structured cases in case-based reasoning: reusing and adapting cases for time-tabling problems. Knowledge-Based Systems 13(2-3), 159–165 (2000)
19. Burke, E.K., MacCarthy, B.L., Petrovic, S., Qu, R.: Case-Based Reasoning in Course Timetabling: An Attribute Graph Approach. In: Aha, D.W., Watson, I. (eds.) ICCBR 2001. LNCS (LNAI), vol. 2080, pp. 90–104. Springer, Heidelberg (2001)
20. Burke, E., MacCarthy, B., Petrovic, S., Qu, R.: Multiple-retrieval case-based reasoning for course timetabling problems. Journal of the Operational Research Society 57(2), 148–162 (2006)
21. Beddoe, G., Petrovic, S.: A novel approach to finding feasible solutions to personnel rostering problems. In: Proceedings of the 14th Annual Conference of the Production and Operations Management Society (April 2003)
22. Petrovic, S., Beddoe, G., Vanden Berghe, G.: Storing and Adapting Repair Experiences in Employee Rostering. In: Burke, E.K., De Causmaecker, P. (eds.) PATAT 2002. LNCS, vol. 2740, pp. 148–165. Springer, Heidelberg (2003)
23. Beddoe, G., Petrovic, S.: Selecting and weighting features using a genetic algorithm in a case-based reasoning approach to personnel rostering. European Journal of Operational Research 175(2), 649–671 (2006)
24. Beddoe, G., Petrovic, S.: Enhancing case-based reasoning for personnel rostering with selected tabu search concepts. Journal of the Operational Research Society 58(12), 1586–1598 (2007)
25. Beddoe, G., Petrovic, S., Li, J.: A hybrid metaheuristic case-based reasoning system for nurse rostering. Journal of Scheduling, 99–119 (2008)
26. Burke, E.K., Petrovic, S., Qu, R.: Case-based heuristic selection for timetabling problems. Journal of Scheduling 9(2), 115–132 (2006)
27. Petrovic, S., Yang, Y., Dror, M.: Case-based selection of initialisation heuristics for metaheuristic examination timetabling. Expert Systems with Applications 33(3), 772–785 (2007)
28. Kolisch, R.: Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. European Journal of Operational Research 90(2), 320–333 (1996)
29. Sprecher, A., Kolisch, R., Drexl, A.: Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem. European Journal of Operational Research 80(1), 94–102 (1995)

Adapting Numerical Representations of Lung Contours Using Case-Based Reasoning and Artificial Neural Networks

Julien Henriet[1,*], Pierre-Emmanuel Leni[1], Rémy Laurent[1], Ana Roxin[1], Brigitte Chebel-Morello[2], Michel Salomon[2], Jad Farah[3], David Broggio[3], Didier Franck[3], and Libor Makovicka[1]

[1] CE UMR 6249 CNRS, Université de Franche-Comté, France
{henriet,leni,laurent,roxin,makovicka}@univ-fcomte.fr
[2] FEMTO-ST UMR 6174 CNRS, Université de Franche-Comté, France
{morello,salomon}@univ-fcomte.fr
[3] French Institute of Radiological Protection and Nuclear Safety, France
{farah,broggio,franck}@irsn.fr

Abstract. In case of a radiological emergency situation involving accidental human exposure, a dosimetry evaluation must be established as soon as possible. In most cases, this evaluation is based on numerical representations and models of subjects. Unfortunately, personalised and realistic human representations are often unavailable for the exposed subjects. However, accuracy of treatment depends on the similarity of the phantom to the subject. The EquiVox platform (Research of Equivalent Voxel phantom) developed in this study uses Case-Based Reasoning principles to retrieve and adapt, from among a set of existing phantoms, the one to represent the subject. This paper introduces the EquiVox platform and Artificial Neural Networks developed to interpolate the subject's 3D lung contours. The results obtained for the choice and construction of the contours are presented and discussed.

Keywords: Adaptation, Interpolation, Case-Based Reasoning, Artificial Neural Network, 3D personalised phantoms.

1 Introduction

In case of accidental exposure to radiation, a dosimetry evaluation must be established for each potential victim (subject) as soon as possible. In most cases, this evaluation is based on available 3D voxel Phantoms, numerical models created from medical images to represent the imaged subject with maximum realism. Examples of voxel phantoms for dosimetric assessment following internal contamination or external exposure can be found [1] [2]. However, even when medical images are available, the subject's specific phantom is not always accessible since its construction is delicate and time consuming, and in emergency cases such time and effort are unaffordable. Moreover, medical

[*] Corresponding author.

images are avoided so as to prevent any additional exposure to radiation. Thus, existing models are used even if their characteristics differ from the subject's biometrical data. Dosimetry assessment accuracy and the resulting decontaminating medical action is nevertheless highly dependent on the similarity between phantom and subject. Hence, the actual work aims at assisting the physician in choosing and customizing the most similar phantom from the existing and available ones. Case-Based Reasoning (CBR) is a problem solving method that uses similar solutions from similar past problems in order to solve new problems [3]. The EquiVox platform uses the CBR approach to find the most similar phantom(s) within any set of phantoms and then attempts to adapt them to the characteristics of the target case (the subject). EquiVox adaptation tool uses Artificial Neural Networks (ANN) [4] to adapt the stored phantoms to the subject. A large number of phantoms can be found in literature [5] [6] and radiation protection is also divided into numerous sub-domains. Indeed, some phantoms are commonly used by experts for external radiotherapy, and others are used by other physicians for evaluation of internal doses received. In fact, each expert has his own collection of 10 to 20 phantoms. When physician's usual phantoms are all too distant from the subject, the expert must create a new one. Indeed, using iterative 3D dilations and contractions, physicians modify the contours of the 3D organs of their phantoms until they correspond to those of the subject. Then, they put them together and obtain the final phantom on which the computations will be based [6]. Thus, the adaptation rules are guided by their experience and knowledge. The main challenge of EquiVox is to reproduce the same transformation process automatically, without human intervention. Another requirement of EquiVox is to be able to use any set of phantoms and to help the physician to capitalise on them. We also hope that such a platform will be used to automatically create a well-fitting phantom for each subject in order to increase the accuracy of dose calculations. At this step of the implementation, we relied on phantoms usually used by a team of experts for pulmonary anthroporadiometry which consists of evaluating the internal dose inhaled.

2 The EquiVox Application

A large number of CBR designed for Health Science (CBR-HS) can be found [7]. Combinaisons with Artificial Intelligence tools can also be found in [8] [9] [10]. EquiVox is a CBR-HS system that uses an AI tool (ANN) during its adaptation phase. The ambition of EquiVox can be seen as giving parameters and tools so as to create prototypical cases [11] or samples for underrepresented classes of subjects [12]. In addition, EquiVox provides an adaptation tool which can create missing samples. Nevertheless, its adaptation knowledge acquisition is automatic since based on ANN trainig, thus not very intelligible as highlighted by M. D'Aquin *et al.* in [13].

Figure 1 presents the technologies that were used and the data flows over the EquiVox architecture. All the phantoms are stored in Rhino3D files [14]. Their characteristics are stored in a database (data flow #0 in Figure 1), the lung contours are extracted (data flow #1) and then transmitted to the ANN training module (data flow #2) which creates the ANN (data flow #3). When a new phantom is required (flow #4), the target case description is transmitted to the retrieval module (data flow #5) which determines the similarity and confidence indices taking into account the source case (data flow #6). If required by the experts, the lung adaptation module sends the characteristics of the source cases (data flow #7) to the ANN interpolation module (data flow #8) which loads the trained ANN (data flow #9) and the coordinates of the contour of the lungs in question (data flow #10) in order to create interpolated contours suited to the target case (data flow #11). Then, the experts can edit and modify manually (create the other organs) the adapted solution of the target case (i.e. the interpolated 3DLC, flow #12) and eventually retain it if the entire 3D phantom is satisfying (flow #13). The adaptation module of EquiVox is not complete yet. Since lungs are the first organs that are designed by experts, we focused on their adaptation while the EquiVox retrieval phase is able to compare the entire phantoms. Thus, the adaptation module of EquiVox deals with the Lung Contours in 3 Dimensions (3DLC). Other studies have been begun to focus on the adaptation of the other organs. Thus, the revision process (flow #12) can only be performed manually at this state of the work.

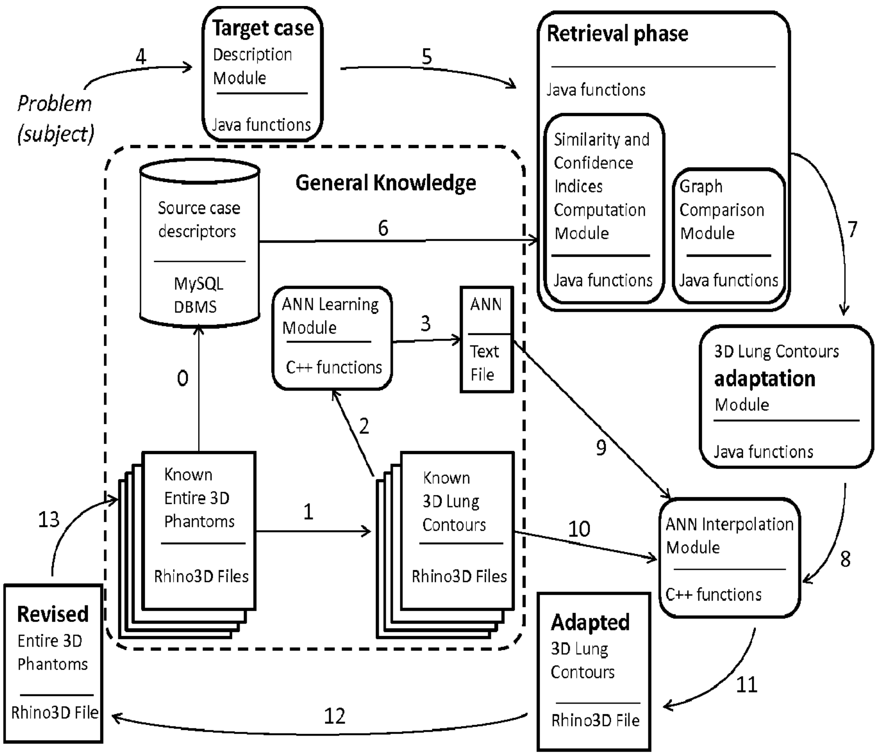


Fig. 1. Data flows over the EquiVox architecture

2.1 Case Modelling

When radiation overexposure occurs, a dosimetric report must be established for all subjects. For each subject, the experts' first task is to choose the most accurate 3D phantom considering the information known about the subject. Each phantom has its own characteristics and is chosen by comparing the subject's available measurements and information with his/her characteristics. The phantom is thus chosen by analogy. As explained, the experts choose the phantom according to the characteristics of the subject. We exhausted the list of useful characteristics provided by the physicians of the French Institute of Radiation and Protection (IRSN).

Thus, in EquiVox, a case *i* is: $case_i = \{pb_i, sol_i\}$.

The problem part of case *i* is described as a set of *r* descriptors: $pb_i = \{d_1^i, \dots, d_r^i\}$.

Each expert has his own set of *n* phantoms $\Omega = \{P_i, \dots, P_n\}$.

Each P_i is the solution part of a case and represents the contours of m organs: $sol_i = P_i = \{P_i^1, \dots, P_i^m\}$.

Each organ O is a set of q points joined by a Delauney mesh [15]: $P_i^O = \{C_1^{i,o}, \dots, C_q^{i,o}\}$ where $C_j^{i,o}$ denotes the 3D coordinates of point j of organ O of phantom P_i. $O \in$ *{lungs, heart, liver, sternum, ribs, scapulae, spine, breast, skin, oesophagus}.*

Finally, a case i is equal to $\{\{d_1^i, \dots, d_r^i\}, P_i\}$. We will note t as target case.

2.2 Retrieval Phase

The purpose of this phase is to sort the phantoms of the EquiVox case-base according to information concerning the subject, even if incomplete. Hence, the number of known descriptors influences the level of confidence in the proposed EquiVox ranking. Thus, along with the similarity index (S_i), a confidence index (C) is assessed to associate the probable error with the retrieved solution. In addition, some descriptors may be very important for some types of calculations while others may be totally neglected. Since the purpose of EquiVox is to retrieve and adapt phantoms, whatever their use, our platform must take into account the importance of each descriptor. Thus, the descriptors were weighted, taking into account their importance and influence. As presented in Equations (1) and (2), these weights $\{\omega_1, \dots, \omega_r\}$ are quantitative values associated to each descriptor, amplifying or reducing the differences between t and i. They thus stress on the relative influence that one measure represents in comparison to the others. In fact, when a new problem occurs, some of the subject's characteristics may be unavailable. Thus, a Boolean value δ_k is associated to each d_k. δ_k is equal to 0 if the value of d_k is unknown, and to 1 otherwise. Hence, a classical algorithm for similarity calculation was used, namely the K-nn Algorithm that enables a weight to be applied to the descriptor values. The S_i value is equivalent to the sum of the distances between the descriptors of i and t, each weighted accordingly. It is given by the following equation:

$$S_i = \frac{\sum_{k=1}^{r} \delta_k . \omega_k . \left[\frac{\Delta_k - |d_k^i - d_k^t|}{\Delta_k}\right]}{\sum_{k=1}^{r} \delta_k . \omega_k} \tag{1}$$

where Δ_k is the difference between the maximum and the minimum known values that the descriptor d_k can take. The S_i value is always between 0 and 1. The greater the similarity of i to t, the closer the S_i value is to 1.

Since S_i only takes into account the known values of t, the confidence index C must be taken into account to define the calculation uncertainty. The more values we know, the higher the confidence index. Indeed, if the subject's age is the only known criteria, the similarity value calculated is totally insignificant. So C takes into account the number of known values according to the following formula:

$$C = \frac{\sum_{k=1}^{r} \delta_k . \omega_k}{\sum_{k=1}^{r} \omega_k} \tag{2}$$

2.3 Adaptation of 3D Lung Contours

Once a matching case is retrieved, the expert can decide either to use the phantom of the most similar source cases, or require the EquiVox platform to generate a new phantom, adapting the source cases to the target one. Indeed, if some available phantom measurements are too different from those of the subject, the expert may decide to adapt one of them or even to create a new phantom which may be reused for other problems later. Thus, when the expert requires the generation of a new phantom, the contours of the m organs are expected. Actually, the first organs experts create in such a personalised process are the lungs. The positions and volumes of the other organs are deduced from the lungs. Thus, we first considered the adaptation of 3D Lung Contours (3DLC).

Solution Space Modelling for 3D Lung Contours. As presented in the case modelling part, the lung contours of phantom P_i are defined in 3D by a set of q points joined by a Delaunay mesh: $P_i^{lung} = \{C_1^{i,lung}, \dots, C_q^{i,lung}\}$ where $C_j^{i,lung}$ denotes the 3D coordinates of point j: $C_j^{i,lung} = \{x_j^{i,lung}, y_j^{i,lung}, z_j^{i,lung}\}$. For all the phantoms, the same number of points defines the 3D contours of the lungs: $q = 26723$. The points have been plotted in the same order and in the same Cartesian coordinate system. Thus, the task of the lung contour-adaptation phase of EquiVox consists of interpolating the 3D coordinates of the points of t in the same order and in the same Cartesian coordinate system. A Delaunay mesh can then be applied so as to create the contours of the lungs of t.

Adaptation Rules. Not all the descriptors that identify the phantoms contained in EquiVox are useful for the adaptation of the lungs. Precisely, it has been proven by I. Clairand *et al.* that the height of a person prevails for the geometry and volume of his or her lungs [16]. Thus, when experts decide to create the lung contours of a subject, they choose the lung contours of the stored phantom whose height is the closest without taking into account any other characteristic. The adaptations are usually done manually, applying mathematical transformations (2D and 3D contractions and dilations [6]). These transformations are carried out through 3D modelling tools (such as Rhinoceros [14]). In addition, these transformations are only driven by experience, trials and errors, and may take many hours or more. The delay also increases with the number of subjects whereas the problem resolution delay may be limited. Indeed, in the case of massive irradiation for example, when a disaster such as a nuclear explosion occurs, dosimetric reports are required for hundreds of people of different sizes. In fact, the creation of new lung contours requires a fast data-driven method, and since there is no physical law to govern its design, the expert is not able to explicit a rule for the transformation of the lung contours. Cordier *et al.* [17] defined *Influence functions* that link problem descriptor variations to solution descriptor variations, and *Dependency functions* that indicate problem descriptors which impact solution ones ($Dependency(C_i^o,d_j) = TRUE$ if C_i^o depends of d_j, $Dependency(C_i^o,d_j) = FALSE$ otherwise). Thus, in the case of EquiVox, noting h the descriptor of the subject's height, $Dependency(C_i^{lung},h) = TRUE$ and an interpolation tool will be used as *Influence function*.

Method. Since the mesh and the number of points are not variable, the adaptation must be carried out on the point coordinates of the lung contours, point by point. Since no formal equation exists, we must discover through a learning method the rules that transform the coordinates of the points on one lung contour into other coordinates. Consequently, data-driven methods using inductive reasoning are the most suitable approaches; ANN and Fuzzy-ANN respond to these requirements. We chose ANN as the tool for this step, assuming this could serve as the basis for further work with Fuzzy-ANN if the first results were not convincing. We explored the possibility of using perceptrons with one hidden layer trained with a backpropagation-based method.

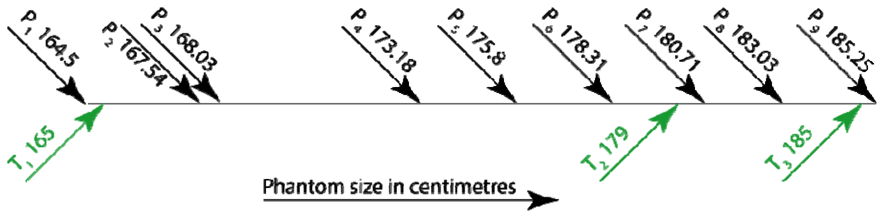


Fig. 2. Phantom heights of the available 3D lung contours

To interpolate the 3D lung contours, the height is required. Actually, this is one of the descriptors of the EquiVox target and source cases. Let us note h_i the descriptor corresponding to the height of the case i and h_t, the height of the target case t. Each $C_i^{t,lung}$ of t is interpolated from $C_j^{i,lung}$, h_i and $\Delta_h=h_i - h_t$ where i is the case for which $|\Delta_h|$ is the smallest. In Figure 2, the 9 heights of the 3DLC P_1 to P_9 used for the training are reported on the axis. Other 3DLCs were also drawn for the test set. The 3 heights of these 3DLCs T_1, T_2 and T_3 are also reported on the same axis. All the thorax organs are represented in P_1 to P_9 whereas only the lungs were drawn in T_1, T_2 and T_3. Actually, the 9 3DLCs (P_1 to P_9) of the training set come from entire phantoms whereas T_1, T_2 and T_3 are 3DLCs that were not extracted from phantoms, but designed manually and espacially for the interpolation tests. The heights of T_1, T_2 and T_3 were carefully chosen so as to cover the entire domain: T_1 is just taller than the smallest phantom (P_1), T_3 is just smaller than the tallest one (P_9), and T_2 represents an average target case. Training ends when the difference between the expected and the obtained values is minimised. W. Hsieh [18] distinguished four algorithms based on the back-propagation method:

- The BFGS method (Broyden-Fletcher-Goldfarb-Shanno) is a quasi-Newton method, which approximates the value of the Hessian matrix of the second derivatives of the function to be minimised;
- The L-BFGS method (Limited memory – BFGS) is an adaptation of the BFGS method which optimises the computational resources to use. Both of these methods must be coupled with a Wolfe linear search in order to determine an optimal step size between two iterations;
- The Rprop (Resistant backpropagation) method proposes a first order algorithm but its complexity increases linearly with network topology;
- The iRpropPlus method is one of the fastest and also one of the most accurate algorithms. This evolution of the Rprop method allows cancelling some synaptic weight updates in the neural network if a negative effect is observed.

Table 1. ANN configuration providing the best preliminary results

Phantom height [cm]	Required precision	Best Learning method
178.31	10^{-6}	BFGS
180.71	10^{-6}	BFGS
183.03	10^{-6}	BFGS

These methods were previously implemented and tested in the EquiVox adaptation phase of 3DLC. Different required precisions were also tested. The coordinates of 10 points were randomly extracted from the 3DLC of P_1 to P_9 and a cross validation was performed. Table 1 shows the algorithm that gave the best interpolations is the one with BFGS as backpropagation method and a precision equals to 10^{-6}. Then, the chosen ANN configuration was compared to a polynomial (Newton, of degree 2) and a Spline interpolation method. The Newton interpolation function proposed by J. Ponce and R. Brette in [19] and the Spline one proposed by Scilab [20] were implemented with Scilab 5.3.2. For each method, a cross-validation for the same 10 points was undertaken using the same 3DLC of P_1 to P_9. Figure 3 presents the mean distances between interpolated and expected coordinates. This figure shows that the polynomial interpolation produced the greatest errors among the three tested interpolations. A factor nearly equal to 10 can be observed between the polynomial interpolation and that of the Spline or the ANN. The Spline and the ANN interpolations gave closer errors. Nevertheless, for all the tested cases, the ANN interpolation errors were inferior to the Spline ones 6 times and equal only once. These results prove the superiority of the ANN interpolations over the other methods since the ANN interpolation gave a more accurate result in all the tested cases.

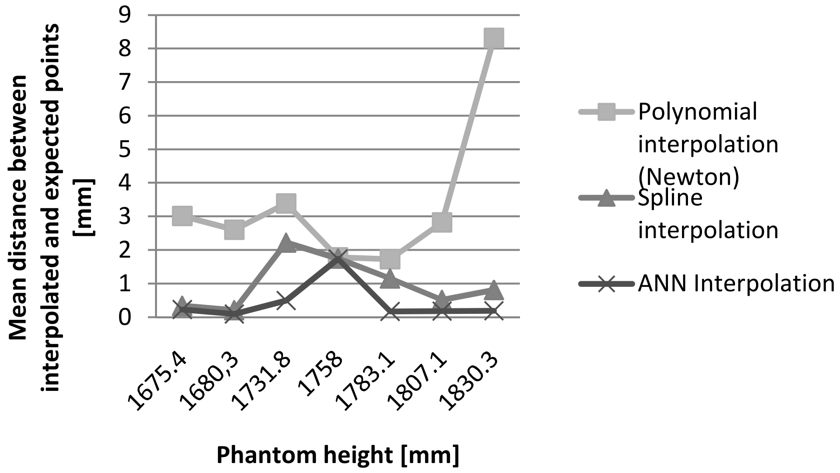


Fig. 3. Mean distances obtained between interpolated and expected coordinates for 10 points and 3 interpolation algorithms

Actually, during the training phase of ANN, learning sets are generally divided in two parts: some of the elements are used to learn while others are used to validate. During this step, the number of neurons of the hidden layers is also determined. Since the number of 3DLCs of our learning set is limited, we wanted to study the impact of some 3DLC in the learning. Thus, we defined two main configurations and four possibilities for each.

Table 2. Learning, validation and test sets tested

	Learning set	Validation set	Test set
Possibility #1	$\{P_1, P_3, P_4, P_5, P_6, P_7, P_8, P_9\}$	$\{P_2\}$	$\{T_1, T_2, T_3\}$
Possibility #2	$\{P_1, P_2, P_4, P_5, P_6, P_7, P_8, P_9\}$	$\{P_3\}$	$\{T_1, T_2, T_3\}$
Possibility #3	$\{P_1, P_2, P_3, P_4, P_5, P_7, P_8, P_9\}$	$\{P_6\}$	$\{T_1, T_2, T_3\}$
Possibility #4	$\{P_1, P_2, P_3, P_4, P_5, P_6, P_8, P_9\}$	$\{P_7\}$	$\{T_1, T_2, T_3\}$

For the first configuration the constraint over the ANN input $h_i - h_t = \Delta_h > 0$ was added and for the second one $\Delta_h < 0$ was required. Then for each configuration, we explored the possibility to extract one particular phantom of the learning set and to include it in the validation set. For each possibility, the test set was always the same: $\{T_1, T_2, T_3\}$. Table 2 shows the different possibilities that were tested.

3 Results

The Equivox platform has been implemented and tested on a *Personal Computer* equipped with an Intel Core i3 CPU, 2.53 GHz, and 4 GiB RAM. The source case descriptors are stored in a mySQL database management system (DBMS). Two programming languages were used: Java and C. The retrieval phase, the GUIs, and the storage phase modules developed by our team in Java call C++ programs also developped by our team for the adaptation phase. The phantoms were drawn using Rhino3D. The ANN learning was performed in C++ on the supercomputer facilities of the *Mésocentre de calcul de Franche-Comté*, that contains 74 nodes based on Intel processors (4 or 6 cores) and 12 to 96 GB of ram. Each learning phase is monothread, so several learning phases could be simultaneously executed on one node.

3.1 EquiVox Case Base

The EquiVox case base used for the tests contained 24 whole 3D phantoms with 3D organ contours and characteristics. These phantoms were manually designed from the ICRP standard female phantom [5] for pulmonary anthroporadiometry computations by the team of internal dose evaluation of IRSN [6]. These 3D phantoms were developed to cover as well as possible the diversity in the female population: thoracic phantoms of cup sizes ranging from A to F and chest girth from 85 to 120 (European Standard Clothing Units) [21]. These phantoms were developed for *in vivo* lung counting optimisation where volume and weight precisions are available for the following structures: lungs, heart, liver, sternum, ribs, scapulae, spine, breasts, skin, and oesophagus. The following external measurements are also available: age, sex, height, weight, cup size, and chest girth (chest and under-bust circumferences). Thus, all these female phantoms and characteristics formed the 24 source cases of the tested EquiVox case base. The experts determined a list of 14 descriptors having varying degrees of influence in the choice of phantom for this type of calculation. These descriptors are age, height, weight, sex, wether the subject smokes or not, thorax volume, lung volume, extrathoracic thickness, fat-muscle proportion, under-bust circumference, wrist diameter, chest circumference, heart volume, and the subject's origin (target case) / phantom (source case). For the adaptation phase tests, the 3D lung contours of these 24 phantoms were considered and extracted. In fact, there are 9 distinct 3D lung contours reported on Figure 2. These 9 3DLCs were used to create the 24 entire phantoms. For example, a phantom with a 90B thorax and one with a 90C have the same 3D lung contours since breast and lung volumes and contours are not correlated at all. In addition, three 3DLCs corresponding to other heights were created by the same process: T_1, T_2 and T_3 whose heights were reported on Figure 2. Actually, the process consists in taking a 3DLC which height is the closest to the target one. Then a scaling factor based on the heights differences is computed and gives the dimensions and volumes of the target lungs. Finally, 3D dilations and contractions are performed according the vertical axis first, the horizontal ones until volumes and dimensions correspond to the ones computed with the scalling factor.

3.2 EquiVox Retrieval Phase Performance

In order to evaluate the performance of the EquiVox retrieval phase, the measurements of 80 different female subjects randomly selected from the CAESAR database

[22] (Civilian American and European Surface Anthropometry Resource database) were considered as target cases descriptions. The latter is a database of over 2000 optical scans of Italian and Danish male and female subjects. Some of their measurements (age, sex, origin, and weight) are also stored with these scans and the spatial resolution enabling calculation of chest girth, cup size, and the height of each subject. IRSN experts determined 5 sets of subject characteristics which influence the pulmonary anthroporadiometry dose computations. The weights ω_k of the associated descriptors from the set influencing the phantom choice the most were set at 4, whereas the weights of those with no influence on that type of computation were set at 0. It is the task of the expert to determine the weights. In the case of *in vivo* counting, it is known that the chest circumference and lung volumes are the most important parameters [23]. Hence, their associated weights were given the highest value: 4. Moreover, in this example, the weights associated to the internal volumes were set at 0. For each target case, we compared the source case the expert would have chosen to the classification proposed by the EquiVox retrieval phase. For 75 target cases, the experts and the EquiVox retrieval phase chose the same source case first. Thus, 5 times, the EquiVox retrieval phase put the source case chosen by the experts in second place. Consequently, in 93.75% of the cases, EquiVox chose the most accurate source case regarding the target case description. The 5 target cases, for which the EquiVox retrieval phase missed the most accurate solution, can be explained by the influence of all other informed descriptors (age, height, weight, etc.). In fact, the difference between the values of these descriptors in these 5 target cases adds up and leads to a low similarity index. For these 5 cases, applying weight values higher than 4 may correct the problem since it would minimize the influence of the least important descriptors. In addition, when no descriptor weighting was assigned ($\omega_k = 1 \; \forall \, k \in \{1, \dots, 14\}$), the EquiVox retrieval phase put the most accurate source case in first place only 54 times.

3.3 Performance Adapations of Lung Contours

As explained in the previous part of this paper, we tested two main configurations for EquiVox adaptation (one considering the phantom' heights inferior to the target' one and one considering the phantom' heights superior to the target'one) and four possibilities for each configuration. Table 3 shows the results obtained with the first configuration (when $\Delta_h > 0$). For the interpolation of T_1, the best results were obtained when P_3 was in the validation set and the worst with P_6 in it instead. For T_2, the most accurate adaptation was obtained when P_7 was in the validation set and the least one with P_6. Concerning T_3, including P_7 in the validation set gave the best interpolations whereas including P_3 gave the worst. Generally, we can remark that P_6 always provided the highest errors and none always gave the best interpolation accuracy. Finally, we can note very important differences between best and worst deviations: the best interpolations were more than twice more accurate than the worst ones.

Table 3. Distances between interpolated and expected points with the first configuration ($\Delta_h > 0$)

3DLC	Deviation [mm]	Possibility (phantom of the validation set) #1 (P_2)	#2 (P_3)	#3 (P_6)	#4 (P_7)
T_1	Mean	1.8	**1.2**	*3.1*	1.5
	Standard	0.7	**0.8**	*1.2*	0.5
T_2	Mean	2.1	1.5	*3.4*	**1.3**
	Standard	0.8	0.8	*1.2*	**0.4**
T_3	Mean	0.9	*2.5*	1.7	**0.5**
	Standard	0.3	*1.2*	0.6	**0.2**

Usually, experts use phantoms described with 1.8 mm by 1.8 mm by 4.8 mm voxels. Regarding this constraint, the best adaptations were satisfying whereas the worst could infer some errors at the dosimetric calculations. Table 4 shows that the best results with the second configuration (when $\Delta_h < 0$) were obtained with the same learning set and validation set for all the tested 3DLCs: $\{P_1, P_2, P_4, P_5, P_6, P_7, P_8, P_9\}$ as learning set and $\{P_3\}$ as validation set. Nevertheless, the worst results were interpolated with *Possibility #3* for $T_1$and T_2, and *Possibility #1* for T_3. Furthermore, the best interpolation computed for $T_1$was less satisfying than the others since the mean error is superior to the voxel dimensions commonly used by experts of radiation protection. Table 4 shows that the best results with the second configuration (when $\Delta_h < 0$) were obtained with the same learning set and validation set for all the tested 3DLCs: $\{P_1, P_2, P_4, P_5, P_6, P_7, P_8, P_9\}$ as learning set and $\{P_3\}$ as validation set. Nevertheless, the worst results were interpolated with *Possibility #3* for $T_1$and T_2, and *Possibility #1* for T_3. Furthermore, the best interpolation computed for $T_1$was less satisfying than the others since the mean error is superior to the voxel dimensions commonly used by experts of radiation protection. Higher differences can be observed between best and worst interpolations of $T_1$and T_2 with this configuration than with the first one: the best interpolations were respectively four and three times more accurate than the worst ones. On the contrary, the difference between best and worst interpolations of T_3 and this configuration were less important than the one with the other. A partial explanation is the distance variations of T_1, T_2 and T_3 from the adapted 3DLC: when $\Delta_h > 0$, T_3 (185cm) was adapted from P_8 (183.03cm) ($\Delta_h = 1.97$cm), whereas for $\Delta_h < 0$, T_3 was adapted from P_9 (185.25cm) ($\Delta_h = -0.25$cm). On the contrary, T_1 (165cm) was adapted from P_1 (164.5cm) when $\Delta_h > 0$ ($\Delta_h = 0.5$cm), and from P_2 (167.54cm) when $\Delta_h < 0$ ($\Delta_h = -2.54$cm); and similarly, T_2 (179cm) was interpolated from P_6 (178.31) ($\Delta_h = 0.69$cm) or P_7 (180.71) ($\Delta_h = -1.71$cm). We can notice that the best interpolations were usually obtained when using the phantom whose height is the closest. As a remark, the adaptations performed with ($\Delta_h > 0$) as additional constraint were generally more accurate than the ones performed with ($\Delta_h < 0$). In addition, whatever the tested configuration was, the learning and validation sets had a great impact on the interpolation accuracies and important differences can be observed. Finally, we can notice that interpolations of T_3 were always twice better than interpolations of the other 3DLCs.

Table 4. Distances between interpolated and expected points with the second configuration ($\Delta_h < 0$)

3DLC	Deviation [mm]	Possibility (phantom of the validation set) #1 (P_2)	#2 (P_3)	#3 (P_6)	#4 (P_7)
T_1	Mean	3.4	**1.9**	*8.4*	3.2
	Standard	1.7	**0.7**	*2.6*	1.1
T_2	Mean	2.4	**1.7**	*5.4*	1.8
	Standard	1.0	**0.6**	*1.7*	0.7
T_3	Mean	*1.2*	**0.8**	0.9	0.9
	Standard	*0.7*	**0.2**	0.3	0.3

4 Discussion

In addition, the necessary delay for an ANN learning can be long (many days), but it is not to be taken into account since it is done when a new source case is capitalised, thus before the emergency situation. Consequently, in emergency situation, only the time of the ANN interpolation and the files creation delay are to be taken into account. During the performed tests, the interpolation and the Rhino3D file creation delays were negligeable since the entire process varied from 1 to 3 seconds. Figure 4 shows some interpolated lungs and their accuracies. Figure 4a presents the most accurate lungs interpolated (T_3 with the validation set #4 and $\Delta_h > 0$) and Figure 4b the worst one (T_1 with the validation set #3 and $\Delta_h < 0$). Each point is colored according to its interpolation error, from blue (the lowest) to red (the highest). Since the interpolation deviations were inferior to the commonly used voxels dimensions of radiation protection experts, the best interpolations for each 3DLC were suitable. Actually, as it is visible in Figure 4, the best results we obtained allow interpolating lung contours with a suitable precision for radiation protection reports. Nevertheless, the interpolation accuracy should be increased for other domains like radiotherapy, where physicians and medicine experts also use such models as a basis for dosimetric reports. Therefore, it emphasizes the importance of the configuration and the 3DLC chosen for each set, since the inclusion of one 3DLC in the validation set can generate an accuracy twice higher or more than another. In addition, including one 3DLC in the validation set introduced a bias for some interpolations and, at the same time, improved the accuracy of another target case (it was the case for P_3 with $T_1$and T_3 when Δ_h was positive for example). Indeed, EquiVox case base is relatively young and limited. Thus, its adaptation phase is limited by the number of known 3DLCs. The results presented in this study show that some 3DLC can introduce bias in the adaptation tool. These results confirm and quantify the general drawback of using interpolation as means of adaptation in CBR systems [24]: imperfections are introduced in adapted solutions.

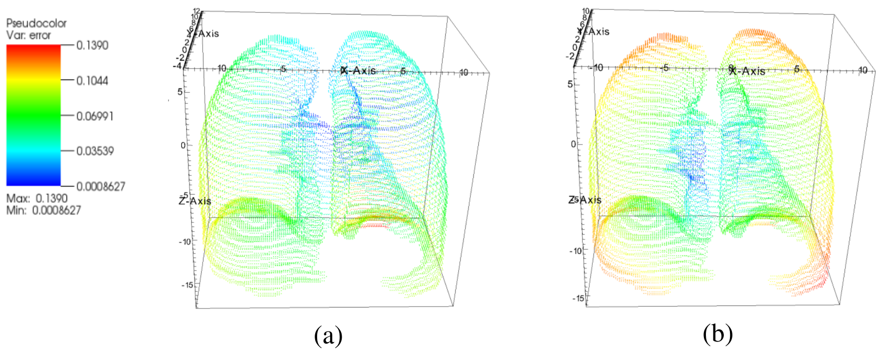


Fig. 4. Representation of an interpolated lung with: (a) T_3 with the validation set #4 and $\Delta_h > 0$, and (b) T_1 with the validation set #3 and $\Delta_h < 0$

Consequently, two ways of improvement are now considered for Equivox. The first one consists in capitalising phantoms and 3DLCs and so to ease progressively the imperfections of the solutions; the ANN interpolations, based on learning sets more and more important, will become better and better. Nevertheless, a second option can be explored, which depends on the association of vectors to the learning set, to optimise interpolation accuracies and to determine, a priori, the best learning set/validation set for each target case.

5 Conclusion

The EquiVox platform was developed for emergency situations, when a fast and reliable decision is required in order to choose the best 3D phantom to perform dosimetry calculation and establish a dosimetric report. The choice is made using the CBR approach based on the feedback from previous similar experiences. EquiVox helps the experts in choosing the most similar 3D phantom by means of the computation of indices for similarity and confidence. The similarity index defines the equivalence between the target case and the source case, whereas the confidence index highlights the uncertainty in the similarity calculation. The tests performed on an average set of target cases gave an efficiency of 93.75% in the application case of *in vivo* female counting for pulmonary anthroporadiometry. Furthermore, an adaptation strategy for 3D Lung Contours (3DLC) was implemented and discussed. This strategy was based on Artificial Neural Networks. Different configurations based on different sets of 3DLC for learning and validation were tested and analysed through the interpolations of three new lung contours. The results show the importance of the choice of the 3DLC repartition between the learning and validation sets: whereas the best interpolations met the requirements of experts, it was not always the case for the worst ones. Some of the interpolation errors were related to the imperfections that can be contained in the source case solutions. Thus, further work will focus on the elaboration of an adaptation algorithm capable of taking into account the confidence that can be associated to a source case solution. In other words, our goal is to propose a tool that creates rules for the adaptation of target cases using this confidence indice. Moreover, we will also extend the EquiVox adaptation to other organ contours of thorax. Finally, if these perspectives concern the reliability of EquiVox, other works and studies will have to be performed in order to guarantee its safety.

Acknowledgements. The authors want to thank *INSERM* (*Institut National de la Santé et de la Recherche Médicale*), *LCC* (*Ligue Contre le Cancer*), *Pays de Montbéliard Agglomération* and *SFRP* (*Société Française de Radioprotection*) for their financial helps.

References

1. Broggio, D., Zhang, B., de Carlan, L., Desbrée, A., Lamart, S., le Guen, B., Bailloeuil, C., Franck, D.: Analytical and Monte Carlo assessment of activity and local dose after a wound contamination by activation products. Health Phys. 96, 155–163 (2009)
2. Huet, C., Lemosquet, A., Clairand, I., Rioual, J.B., Franck, D., de Carlan, L., Aubineau-Lanièce, I., Bottollier-Depois, J.F.: SESAME: a software tool for the numerical dosimetric reconstruction of radiological accidents involving external sources and its application to the accident in Chile in December 2005. Health Phys. 96, 76–83 (2009)
3. Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann Publishers (1993)
4. McCulloch, W., Pitts, W.: A logical calculus of ideas immanent in nervous activity. Bulletin of Mathematical Biophysics 5, 115–133 (1943)
5. ICRP89. Basic anatomical and physiological data for use in radiological protection. International Commission on Radiological Protection Publication 89 (2002)
6. Farah, J., Broggio, D., Franck, D.: Examples of Mech and NURBS phantoms to study the morphology effect over in vivo lung counting. Radiation Protection and Dosimetry Special Issue 144, 344–348 (2011)
7. Bichindaritz, I.: Case-Based Reasoning in the Health Sciences: Why It Matters for the Health Sciences and for CBR. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 1–17. Springer, Heidelberg (2008)
8. Diaz, F., Fdze-Riverola, F., Corchado, J.M.: Gene-CBR: A Case-Based Reasoning Tool for Cancer Diagnosis using Microarray Datasets. In: Computational Intelligence, pp. 254–258 (2006)
9. El Balaa, Z., Strauss, A., Uziel, P., Maximini, K., Traphoner, R.: FM-Ultranet: A Decision Support System Using Case-Based Reasoning Applyied to Ultrasono-graphy. In: McGinty, L. (ed.) Workshop Proceedings of the Fifth International Conference on Case-Based Reasoning, pp. 37–44. NTNU, Trondheim (2003)
10. Monati, S.: Case-Based Reasoning for Managing Non-Compliance with Clinical Guidelines. In: Wilson, D.C., Khemani, D. (eds.) Proceedings of Case-Based Reasoning in Health Science Workshop, ICCBR, Belfast, pp. 325–336 (2007)
11. Bichindaritz, I.: Prototypical Cases for Knowledge Maintenance in Biomedical CBR. In: Weber, R.O., Richter, M.M. (eds.) ICCBR 2007. LNCS (LNAI), vol. 4626, pp. 492–506. Springer, Heidelberg (2007)
12. Little, S., Colatino, S., Salvetti, O., Perner, P.: Can Prototype-Based Classification be a good Method for Biomedical Applications? Trans. MLDM (MLDM) 2(1), 44–61 (2009)

13. D'Aquin, M., Lieber, J., Napoli, A.: Adaptation Knowledge Acquisition: A Case Study for Case-Based Decision Support in Oncology. In: Computational Intelligence, pp. 161–176 (2006)
14. McNeel. Rhinoceros Modeling tools for designers, `http://www.rhino3d.com`
15. Christensen, G.E.: Deformable shape models for anatomy. Washington University. PhD Thesis (1994)
16. Clairand, I., Bouchet, L.G., Ricard, M., Durigon, M., Di Paola, M., Aubert, B.: Improvment of internal dose calculations using mathematical models of different adult heights. Phys. Med. Biol. 45, 2771–2785 (2000)
17. Cordier, A., Fuschs, B., Mille, A.: Engineering and learning of adaptation knowledge and Case-Based Reasoning. In: Proceedings of the 15th International Conference on Knowledge Engineering and Knowledge Management, EAKW 2006, pp. 303–317. Springer (2006)
18. Hsieh, W.: Learning Methods in the Environmental Sciences - Neural Networks and Kernels. Cambridge University Press (2009)
19. Ponce, J., Brette, R.: Polynomial interpolation. Introduction to scientific computing and its applications, `http://audition.ens.fr/brette/calculscientifique/2006-2007/lecture2.pdf`
20. Digiteo. Scilab Home Page, `http://www.scilab.org`
21. (CEN), European Committee for Standardization: Size designation of clothes: -part 1. Terms definitions and body measurement procedure. EN 13402-1 (ISO 3635: 1981 modified) (2001)
22. Robinette, K.M.: CAESAR. measures up. Ergonomics in Design 8(3), 17–23 (2000)
23. Kramer, G.H., Burns, L.C.: Evaluation of the effect of chest wall thickness, tissue composition and photon energy on the quantity muscle equivalent chest-wall-thickness by Monte Carlo simulation. Radiat. Prot. Dosim. 82, 115–124 (1999)
24. Chatterjee, N., Campbell, J.A.: Interpolation as a means of fast adaptation in case-based problem solving. In: Bergmann, R., Wilke, W. (eds.) 1st German Workshop on Case-Based Reasonning, Kaiserslautern, Germany, pp. 65–74 (1997)

Adaptation in a CBR-Based Solver Portfolio for the Satisfiability Problem

Barry Hurley and Barry O'Sullivan

Cork Constraint Computation Centre,
Department of Computer Science, University College Cork, Ireland
{b.hurley,b.osullivan}@4c.ucc.ie

Abstract. The satisfiability problem was amongst the very first problems proven to be NP-Complete. It arises in many real world domains such as hardware verification, planning, scheduling, configuration and telecommunications. Recently, there has been growing interest in using portfolios of solvers for this problem. In this paper we present a case-based reasoning approach to SAT solving. A key challenge is the adaptation phase, which we focus on in some depth. We present a variety of adaptation approaches, some heuristic, and one that computes an optimal Kemeny ranking over solvers in our portfolio. Our evaluation over three large case bases of problem instances from artificial, hand-crafted and industrial domains, shows the power of a CBR approach, and the importance of the adaptation schemes used.

1 Introduction

The satisfiability (SAT) problem is defined as follows: given a propositional formula, $\phi = f(x_1, \ldots, x_n)$, over a set of variables $x_1, \ldots, x_n$, decide whether or not there exists a truth assignment to the variables such that ϕ evaluates to true.

SAT problem instances are usually expressed in a standard form, called conjunctive normal form (CNF). A SAT problem, in this form, is expressed as a conjunction of *clauses*, where each clause is a disjunction of *literals*; a literal is either a variable or its negation. The following SAT formula is in CNF:

$$\phi = (x_1 \vee x_3 \vee \neg x_4) \wedge (x_4) \wedge (x_2 \vee \neg x_3).$$

This formula comprises three clauses: the first a disjunction of literals x_1, x_3 and $\neg x_4$; the second involves a single literal x_4; the third is a disjunction of literals x_2 and $\neg x_3$. The SAT problem ϕ is satisfiable because we can set x_1, x_2 and x_4 to *true*, satisfying the first, third and second clauses, respectively.

SAT problems occur in a variety of domains such as hardware verification, security protocol analysis, theorem proving, scheduling, routing, planning, digital circuit design and artificial intelligence [1]. Deciding whether a SAT problem is satisfiable or not is usually performed by either systematic search, based on backtracking, or local search. Because the general problem is NP-Complete, systematic search algorithms have exponential worst-case run times, which has the effect of limiting the scalability of these methods. If a SAT problem is unsatisfiable, local search algorithms, while scalable, cannot prove unsatisfiability.

Over the past decade there has been a significant increase in the number of satisfiability solving systems that have been developed. It is recognised that different solvers are better at solving different problem instances, even within the same problem class [2]. It has been shown that the best on-average solver can be out-performed by a portfolio of possibly slower on-average solvers because of complementarities amongst them, i.e. a slow on-average solver might have best performance on a particular instance. Three specific approaches that use contrasting approaches to portfolio management for the constraint satisfaction problem (CSP), SAT and quantified Boolean formula (QBF) are CPHYDRA, SATZILLA and AQME, respectively. CPHYDRA is a portfolio of constraint solvers that exploits a case base of problem solving experience [3]. CPHYDRA combines case-based reasoning with the idea of partitioning CPU-TIME between components of the portfolio in order to maximise the expected number of solved problem instances within a fixed time limit; CPHYDRA is an earlier piece of work in our research programme on portfolios, but does not consider alternative approaches to adaptation, which we study here. SATZILLAbuilds run time prediction models using linear regression techniques based on structural features computed from instances of the Boolean satisfiability problem [4]. Given an unseen instance of the satisfiability problem, SATZILLA selects the solver from its portfolio that it predicts will have the fastest running time on the instance. The AQME system is a portfolio approach to solving quantified Boolean formulae, i.e. SAT instances with some universally quantified variables [5].

The objective of the work reported in this paper is to study a simple case-based reasoning approach to a portfolio for the SAT problem. We present three large case bases of problem-solving experience with a large number of modern SAT solvers in three distinct domains, including one comprising almost 1200 industrial problems (Section 2), which we have made available online. We focus primarily on the problem of adaptation, having retrieved a suitable set of similar experiences involving problems similar to the one we wish to solve (Sections 3 and 4). Our results (Section 5) demonstrate that a case-based reasoning approach would perform close to oracle performance on the domains we evaluate, exhibiting a potential killer application domain for case-based reasoning. These results are consistent with the belief held in the SAT community that experience plays a key role in selecting a good solver for a problem instance.

2 Building Case-Bases for SAT Solving

We summarise the representation, cases and similarity measure used in our three case bases for SAT solving. Our case bases relate to three domains: industrial instances, hand-crafted instances and randomly generated instances.

Feature Representation. We employed the same set of SAT instance features as those used in SATZILLA.[1] SATZILLA is a successful algorithm portfolio for

[1] http://www.cs.ubc.ca/labs/beta/Projects/SATzilla/

Problem Size Features:
1. **Number of clauses**: denoted c
2. **Number of variables:** denoted v
3. **Ratio:** c/v

Variable-Clause Graph Features:
4-8. **Variable nodes degree statistics:** mean, variation coefficient, min, max and entropy.
9-13. **Clause nodes degree statistics:** mean, variation coefficient, min, max and entropy.

Variable Graph Features:
14-17. **Nodes degree statistics:** mean, variation coefficient, min and max.

Balance Features:
18-20. **Ratio of positive and negative literals in each clause:** mean, variation coefficient and entropy.
21-25. **Ratio of positive and negative occurrences of each variable:** mean, variation coefficient, min, max and entropy.
26-27. **Fraction of binary and ternary clauses**

Proximity to Horn Formula:
28. **Fraction of Horn clauses**
29-33. **Number of occurrences in a Horn clause for each variable:** mean, variation coefficient, min, max and entropy.

DPLL Probing Features:
34-38. **Number of unit propagations:** computed at depths 1, 4, 16, 64 and 256.
39-40. **Search space size estimate:** mean depth to contradiction, estimate of the log of number of nodes.

Local Search Probing Features:
41-44. **Number of steps to the best local minimum in a run:** mean, median, 10th and 90th percentiles for SAPS.
45. **Average improvement to best in a run:** mean improvement per step to best solution for SAPS.
46-47. **Fraction of improvement due to first local minimum:** mean for SAPS and GSAT.
48. **Coefficient of variation of the number of unsatisfied clauses in each local minimum:** mean over all runs for SAPS.

Fig. 1. A summary of the features used to describe SAT instances in our case base. These are the same features used in SATzilla [4].

SAT, i.e. a system that uses machine learning techniques to select the fastest SAT solver for a given problem instance. That system uses a total of 48 features, summarised in Figure 1 [4]. These features can be summarised under nine different categories: problem size features; variable-clause graph features; variable graph features; balance features; proximity to Horn formula; DPLL probing features; and local search probing features. The first category of features are self explanatory, and simply relate to the number of variables and clauses in the SAT instance. The next two categories relate to two different graph representations of a SAT instance. The variable-clause graph is a bipartite graph with a node for each variable, a node for each clause, and an edge between them whenever a variable occurs in a clause. The variable graph has a node for each variable and an edge between variables that occur together in at least one clause. The balance features are self explanatory and relate, primarily, to the distribution of positive and negative literals within the SAT instance. Another category measures the proximity to a Horn formula. This captures how close the SAT instance is to an important polynomial class of SAT that can be solved using the standard inference method used in all systematic SAT solvers (i.e. unit propagation). The DPLL probing features are related to statistics that a standard systematic search algorithm gathers while testing the difficulty of the instance [6]. The local search features are the non-systematic analogue of the latter category.

Cases. We built three case bases from the training data used by the SATzilla system [4].[2] Each case in the case base represents one SAT problem instance and the individual performance of a set of solvers when applied to it. For each benchmark instance in the dataset, there is a record of whether each solver solved

[2] SATzilla data: `http://www.cs.ubc.ca/labs/beta/Projects/SATzilla/`

Table 1. The number of problem instances and solvers in each of our three case bases

Case base	**# Instances**	**# Solvers**
Handcrafted (HAN)	1181	19
Industrial (IND)	1183	19
Random (RAN)	2308	27

the instance within a specified cut-off time (1 hour), the time taken by each to solve the instance, and whether the solver crashed during execution. Each solver is run independently of each other. Thus, each case is a pair, (F, S), where F is a set of feature values and S is a set of pairs encoding the performance of each solver on the query instance.

The problem instances were originally taken from the benchmark suites associated with the annual International SAT Competition.[3] We combine the instances from each year of the SAT competition into a single combined dataset. Each instance is assigned to one of three categories: handcrafted (HAN), industrial (IND) and randomly generated (RAN) problem instances. The instances are additionally separated into what SATzilla classified as satisfiable and unsatisfiable instances. For our evaluation, instances from these two classifications were combined into a single dataset. Table 1 gives the resulting size of each case base.[4] Table 1 also shows the number of solvers in each. Note that the random category contains eight additional solvers that are not available in the dataset for the handcrafted and industrial categories.

Similarlity Metric. The features that encode the cases are all numeric. For the purposes of this paper we assume that the similarity between two cases is computed using the unweighted normalised Euclidean distance. Feature values are normalised to the interval $[0, 1]$. Specifically, for the ith feature of case γ, we compute the normalised value $\eta(\gamma[i])$ of feature value $\gamma[i]$ as follows:

$$\eta(\gamma[i]) = \frac{\gamma[i] - min(i)}{max(i) - min(i)}.$$

where $min(i)$ and $max(i)$ are the minimum and maximum values respectively for feature i across all cases.

When evaluating a test case, one finds the k cases with highest similarity (smallest Euclidean distance) in the case base. The challenge is then, given the set of performance data for each solver, how should one adapt this experience to the current problem instance. We frame this problem as a label ranking task [7], which we will discuss in greater detail in the following section.

[3] `http://www.satcompetition.org`

[4] The case bases are available at `http://osullivan.ucc.ie/datasets/iccbr2012/`

3 Adaptation Strategies

The specific task of adaptation in our portfolio context is to decide which solver should be used to solve a given instance. We will consider a setting in which we are allocated one hour to solve an instance. The objective is, given a set of problem instances, to solve as many of them as possible within the cutoff in the shortest time. In other words, we lexicographically order these two objectives: maximise the number of solved instances, and tie-break by running time, which we prefer to minimise. In our setting, each nearest neighbour can be seen as giving an ordering over the performance of the available solvers. We can interpret this order as a ranking, possibly time-weighted.

In traditional classification, we are interested in assigning one or more labels from a finite set of labels, to a case. In contrast, label ranking deals with assigning a total ordering of labels to a case. This ranking of labels can be much more useful than assigning a single label, e.g. rank aggregation methods have been used to combine query results from multiple search engines [8]. In this paper, the labels represent different algorithms in our portfolio solver and the ranking of labels represents the expected order of run time on an instance.

Label ranking may also be seen as a generalization of multi-label classification. Instead of classifying a case with a subset of the classes, we instead assign a totally ordered ranking of the classes. Multi-label ranking is the task whereby in addition to producing a total ordering of the labels for an instance, the task is to also identify a partition of the labels into relevant and irrelevant labels [9,10]. This introduces an additional layer of complexity to the task. Methods for learning pairwise preferences between labels have been proposed [7]. It has been shown that case-based label ranking compares well against model-based approaches [11]. An in depth survey of additional label ranking methods is given in [12].

We consider a variety of voting-based approaches for label ranking and consensus ranking; we refer the reader to the literature for further details of the various methods [13]. We will use examples throughout, based on the sample data presented in Table 2. In this table, we present an example of the retrieval set from our case-based system, but do not present the running times. Instead we simply order the solvers by running time.

Kendal-Tau Distance. To compare two rankings, we define a function to compute the distance between them. The Kendal-Tau distance between two rankings A and B is the number of discordant pairs. Let L be the set of all labels and let A and B be two complete rankings of these labels. Formally:

$$\text{KT-distance}(A,B) = \sum_{c,d \in L,\, c \neq d} \begin{cases} 1 & \text{if } A \text{ and } B \text{ rank } c \text{ and } d \text{ in a different order} \\ 0 & \text{otherwise}. \end{cases}$$

Example 1. Using the example rankings given in Table 2, the Kendal-Tau Distance between rankings A and B is:

$$\text{KT-distance}(A,B) = 3$$

Table 2. An example list of label rankings, which we will use as a running example. Each ranking contains a total ordering of the labels a, b, c, d. The operator $\prec$ can be read as 'faster than'.

Name	**Label Ranking**
A	$a \prec b \prec c \prec d$
B	$c \prec b \prec a \prec d$
C	$b \prec a \prec d \prec c$
D	$a \prec c \prec d \prec b$
E	$b \prec a \prec c \prec d$

This is because the pairs of candidates (a, b), (a, c) and (b, c) are ranked differently by the two rankings. A and B both rank the other pairs, e.g. (c, d), in the same order. □

Kemeny Consensus Ranking. The Kemeny Score of a ranking R is the sum of all the Kendal Tau distances from R to all rankings among the votes V.

$$\text{Kemeny-Score}(R, V) = \sum_{v \in V} \text{KT-distance}(R, v)$$

Example 2. Let $R = \langle a \prec c \prec b \prec d \rangle$. If we take all rankings from Table 2 as the votes, then the Kemeny Score of R is 9. This is the sum of:

$$\begin{array}{lll} \text{KT-distance}(R, A) = 1 & \text{KT-distance}(R, B) = 2 & \text{KT-distance}(R, C) = 3 \\ \text{KT-distance}(R, D) = 1 & \text{KT-distance}(R, E) = 2 & \end{array}$$

□

The Kemeny Consensus is the ranking of the labels that minimises the Kemeny Score. This may also be referred to as the optimal Kemeny ranking. It is the ranking which minimises, among the votes, the number of disagreements on the pairwise preference between every pair of labels. Aggregating multiple rankings into a single optimal Kemeny ranking is NP-hard [8].

Example 3. For the votes given in Table 2, the optimal Kemeny ranking is $\langle b \prec a \prec c \prec d \rangle$ with a Kemeny Score of 7. All other possible permutations of the labels have a higher Kemeny Score than this. In this case the Kemeny Optimal ranking matches one of the rankings in the votes, but this may not necessarily be the case. □

The Kemeny Consensus ranking is said to satisfy the Condorcet criterion. This states that if a candidate is preferred by most voters to any other candidate, then it should be ranked first in the aggregation ranking. It expresses no condition on the remainder of the positions, however.

Borda Voting. Borda voting is a polynomial time approximation scheme for the Kemeny Consensus ranking. Each of the k nearest neighbours votes for each label in the order of which that solver performed on that case. A label in position p receives $n - p + 1$ points based on its position in the ranking. The points for each candidate are summed up. The ranking is produced by sorting these tallies in decreasing order. Borda voting does not satisfy the Condorcet criterion.

Example 4. For vote B in Table 2, candidates c, b, a and d would receive 4, 3, 2 and 1 points respectively. If we sum up the points from all the votes in this table, the a would have a score of 16 points, b of 15, c of 12 and d of 7. The resulting ranking would be $\langle a \prec b \prec c \prec d \rangle$. □

Weighted Borda Voting. Weighted Borda voting takes extra information about each neighbour into account. The vote for a particular solver is multiplied by the weight in one of the two weighting schemes we consider. In distance-weighted Borda voting (DW-BV), the weight W_D is given as $W_D = \frac{1}{1+d}$ where d is the Euclidean distance between the neighbour and the test instance.

In time-weighted Borda voting (TW-BV) the weight W_T is given as $W_T = \frac{\text{cutoff}-t}{\text{cutoff}}$, where 'cutoff' is the cut-off execution time limit and t is the time taken for the solver on a given neighbour. Time-weighted Borda voting gives a large weight to solvers that take very little time to solve the instance and a weight of zero to any that timeout or do not solve the instance. This suits our goal of choosing the solver that will perform fastest for a given instance.

Copeland Voting. Copeland voting looks at every pair of labels (a, b) and counts the difference between the number of votes that prefer a to b and those that prefer b to a. The label with the higher number of preferences gets one added to its score. The label with the lower number of preferences gets one deducted from its score. The resulting ranking of the labels is obtained by sorting on their respective scores.

Example 5. Given the votes in Table 2, the label ranking produced by Copeland voting would be $\langle b \prec a \prec c \prec d \rangle$. The scores for each label would be: (a, 1), (b, 3), (c, -1), (d, -3). □

Bucklin Voting. Bucklin voting is a means of choosing the label with the best median ranking. The algorithm first attempts to select the label that has a majority of first preference votes. The number of first preferences for each label is counted across the votes. If one of the labels has a majority, then that label is the winner. If no label has a majority, then the second preference votes are added to the first. Again, if there is a label that has a majority of votes, then that label is the winner. There may be multiple labels with a majority. In this case, the winner is the one with the highest vote tally.

Coomb's Voting. Coomb's voting is similar to Bucklin voting in that it first attempts to select the label that has the majority of first preference votes. If no label has a majority, a separate election is held between the labels that are ranked last in the votes. The label with the most last-place votes is then removed from all votes. A tally of the first preference votes is taken again and this is repeated until there is a label with a majority.

Instant Runoff Voting. In Instant Runoff voting (IRV), we again stop if there is a label with a majority of first place votes. If not, then the label with the fewest first preference votes is eliminated. This label is removed from each of the votes. For each vote where the eliminated label held a first place preference, the next preference votes are added to their respective label's tally. This is repeated until there is a candidate with a majority of votes. This voting scheme is similar to Coomb's voting except instead of eliminating the label that is ranked last, we eliminate the label that has the fewest first preference votes.

Best Average Score. Among the data for the k nearest neighbour instances is the run time for each solver on that neighbour instance. The Borda voting and distance weighted Borda voting methods above do not take this valuable data into account when performing their aggregation. Another strategy to get a ranking of the solvers from this data is to order them by their average score across these k instances. This gives us an ordering of the solvers by their performance, averaged across the k nearest neighbours. We refer to this aggregation as ordering by Best Average Score.

Very Best Ranking. The Very Best Ranking (VBR) is the ranking produced by an oracle, who knows the best ranking for each instance. We use this ranking as a benchmark to compare the rankings produced by the aggregation methods.

4 An Exact Method for Optimal Kemeny Ranking

The ranking methods presented in the previous section, with the exception of the optimal Kemeny and VBR rankings, are heuristics. In this section a Mixed Integer Programming (MIP) model is presented for computing the optimal Kemeny ranking from the k nearest neighbors of a query SAT instance as an optimization problem. This model was implemented using the combinatorial optimization system Numberjack[5] using SCIP as the underlying MIP solver.[6]

Let L be the set of all labels. Let V be the set of votes from each of the k nearest neighbors. We encode each ranking as a list where each label takes the value of the number of labels ranked higher than it. For example, if we are given a ranking of the labels $c \prec a \prec d \prec b$, then this would be converted to $\langle 1, 3, 0, 2 \rangle$

[5] Available under LGPL from `http://numberjack.ucc.ie/`
[6] SCIP: `http://scip.zib.de/`

because a has 1 candidate ahead of it, b has 3, and so on. This simplifies the process of finding the index of a label within a ranking for the MIP model. We define the MIP model as follows:

- R is the array of the rank indices in the Kemeny Consensus ranking. R_i states the number of labels that are ranked before label i in the aggregation ranking. The domain of values that each position in R can take is therefore $0 \dots m-1$. This array contains all the decision variables.
- We add the constraint that the values taken by the variables in R are all different because only one candidate can occupy each position.
- For each pair of labels i and j, we have a binary variable r_{ij} which is encoded to take the value 1 if i is ranked higher than j in the target ranking R, 0 otherwise.
- For each pair of labels i and j in each vote V_k we have a binary variable v_{kij} which takes the value 1 if label i is ranked higher than label j in vote V_k, 0 otherwise.
- For each pair of labels i and j in each vote V_k we have the binary variable D_{kij} which is the exclusive-or between r_{ij} and v_{kij}. This means D_{kij} will take the value 1 iff R ranks i and j in a different order to V_k.
- The Kendal Tau distance to vote V_k from R is KT_k, which is the sum over all D_{kij} for every pair of candidates i and j.
- The Kemeny Score of the target ranking R is $\sum KT_k$. We attempt to minimise this value.

For a set of votes among 19 labels, this MIP model is able to solve the difficult aggregation problem in a matter of seconds. Consider that a greedy naive algorithm for commuting the Kemeny Optimal ranking may need to examine every possible permutation of the labels, which is $O(n!)$. It must compute the Kemeny Score for each permutation and choose the ranking that minimises this function. This approach quickly becomes infeasible.

5 Evaluation

We present an evaluation of both the quality of our adaptation strategies for ranking solvers by run time (Section 5.1), and the performance of our case-based reasoning-based solver portfolio for SAT (Section 5.2). We use the case bases described in Section 2. In terms of the quality of the rankings, we show that rankings that consider running time, rather than relative position in the rank, give better performance. This is somewhat unsurprising, but it is interesting to see that the effort spent in finding the optimal Kemeny ranking is not worthwhile.

Of much greater significance is our demonstration that our CBR portfolio out-performs all of its constituent solvers by a considerable margin. In fact, the superiority of the CBR approach is observed regardless of the adaptation scheme used. Again, rankings that consider time are superior to all others, and compare well in terms of performance against the oracle (VBR) that always selects the best solver for a particular SAT instance.

Methodology. In all experiments we used a 10-fold cross validation approach, studying each of our three case bases (Section 2) separately. We report averages, where appropriate. We always seek five nearest neighbours (5-NN), having observed that setting k to this value gave good typical-case performance. For the purpose of this paper, unweighted normalised Euclidean distance is used as a similarity metric throughout. All adaptation methods use the same distance measure, therefore each are tasked with aggregating the same set of neighbours.

5.1 Evaluation of the Adaptation Schemes

Given a ranking of the solvers, using a particular adaptation scheme, and their respective execution times, we can plot the cumulative execution time of each solver against its position in the ranking. Let $s(i)$ be the solver ranked in position i of a ranking, and $t(\alpha)$ be the time taken by solver α to solve the instance. The plot of the cumulative time of the solvers in a ranking is given by:

$$f(x) = \sum_{i=1}^{x} t(s(i)).$$

If the solvers are ordered in strict order of increasing run time, the area under the curve in this plot will be minimised. On the other hand, the ranking which is as poor as possible will have maximum area. We compare each of our adaptation strategies that produce a ranking in this way. Figure 2 shows an example plot of the curve for each of the label rankings produced by an adaptation method.

We performed paired t-tests to compare two label ranking methods on the basis of the area under the curves in our ranking plots. Such a paired t-test was performed between every pair of adaptation methods on every instance across the 10 splits in each dataset category. Table 3 gives the complete table of these results showing the 95% confidence interval and the p-value. In this table a confidence interval with negative lower and upper bounds, which is highlighted in bold, signifies that the ranking on the left is statistically significantly better than the ranking on the right.

On hand-crafted and industrial problems, which are really the most interesting from a practical viewpoint, the best-average-solver (BAS) and three variants of Borda voting out-perform all other methods; the statement is almost also true in the random category. It is clear, and not unsurprising, that the methods that take running time into account, out-perform all others. The Kemeny ranking never out-performs another method.

While comparing the rankings is interesting in itself, the more important question is how effective are these rankings in a CBR-based algorithm portfolio for SAT. We study this below.

5.2 Evaluation of the CBR-Based Solver Portfolio for SAT

We implemented a variant of our basic CBR-based solver portfolio using each of our adaptation schemes in turn. We name these using the acronym of the

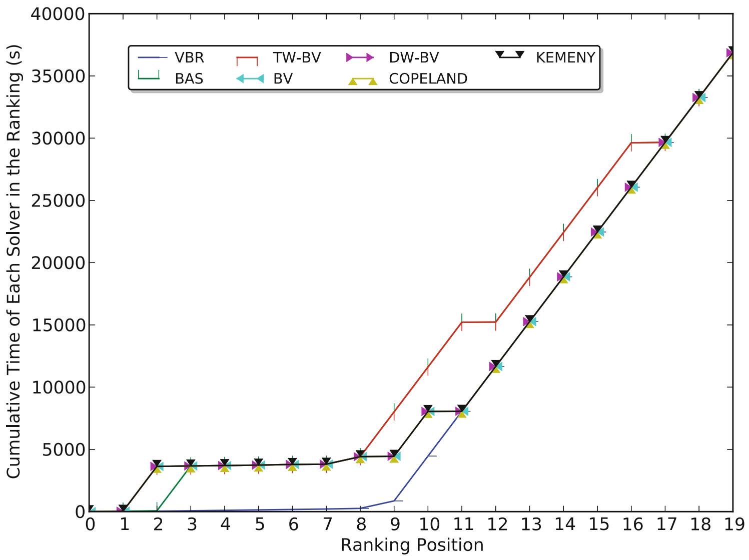


Fig. 2. An illustration of the curve produced by accumulating the time taken by each solver in a ranking. Each line represents a curve produced by the rankings from each of the adaptation methods that produce a ranking.

adaptation scheme used. Given a SAT instance, the portfolio system will apply the relevant adaptation scheme to the set of cases retrieved using our 5-NN method. The highest ranked solver is selected. We record the total number of instances solved by the chosen solvers and the cumulative time summed over all solved instances, given a cut-off time of 1 hour per instance, in a 10-fold cross validation setting. This setup is very similar to that of the International SAT Competition.

We compare this to the Very Best Ranking (VBR), which chooses the best solver for the instance given. We report the average number of instances solved and the average run time, with standard deviation in both cases. In our results tables (Tables 4, 5 and 6) we sort the variants in terms of number of instances solved, and then by run time. The VBR, the oracle, is therefore always ranked at the top. Due to space constraints, these leader boards only show the top 15 positions.

The overall result is that the best SAT solvers are out-performed in every problem class by each of the CBR-based portfolios. The CBR portfolio compares very well against the oracle (VBR) in each category. For example, in the random problem category, the CBR portfolio solves 33% more instances than the best SAT solver on its own, and solves within 5% of the instances solved by the oracle.

Table 3. Table of paired t-tests comparing the area under the curve for each of the methods. Results with a negative interval and p-value below the threshold of 0.05 are highlighted in **bold**. This means that the first method is statistically significantly better than the second method for that dataset. Results with a positive interval and p-value below the threshold of 0.05 are highlighted in *italics*. This means that the second method is statiscally significantly better than the first method for that dataset.

		Handcoded			**Industrial**			**Random**		
		95% Conf. Int.		p-value	95% Conf. Int.		p-value	95% Conf. Int.		p-value
VBR	BAS	**-8615.4**	**-6366.5**	**0.000**	**-7186.6**	**-5286.6**	**0.000**	**-23538.6**	**-19300.2**	**0.000**
VBR	TW-BV	**-9425.4**	**-7119.7**	**0.000**	**-7857.2**	**-5864.8**	**0.000**	**-27900.0**	**-23160.4**	**0.000**
VBR	DW-BV	**-14736.7**	**-11755.3**	**0.000**	**-10613.7**	**-8388.4**	**0.000**	**-53803.9**	**-46149.2**	**0.000**
VBR	BV	**-14902.9**	**-11866.3**	**0.000**	**-11065.8**	**-8671.7**	**0.000**	**-53795.7**	**-46126.8**	**0.000**
VBR	COPELAND	**-17103.5**	**-13394.1**	**0.000**	**-12239.6**	**-9423.4**	**0.000**	**-55608.2**	**-46523.2**	**0.000**
VBR	KEMENY	**-17173.4**	**-13463.0**	**0.000**	**-11854.5**	**-9303.8**	**0.000**	**-70833.4**	**-61940.3**	**0.000**
BAS	TW-BV	**-1196.0**	**-367.2**	**0.000**	**-876.0**	**-372.8**	**0.000**	**-5223.4**	**-2998.3**	**0.000**
BAS	DW-BV	**-7054.4**	**-4455.7**	**0.000**	**-4141.7**	**-2387.2**	**0.000**	**-31689.3**	**-25425.0**	**0.000**
BAS	BV	**-7218.2**	**-4569.1**	**0.000**	**-4600.1**	**-2664.1**	**0.000**	**-31675.3**	**-25408.4**	**0.000**
BAS	COPELAND	**-9420.1**	**-6095.6**	**0.000**	**-5787.9**	**-3401.8**	**0.000**	**-33551.5**	**-25741.1**	**0.000**
BAS	KEMENY	**-9499.7**	**-6154.9**	**0.000**	**-5396.9**	**-3288.2**	**0.000**	**-48835.7**	**-41099.2**	**0.000**
TW-BV	DW-BV	**-6203.8**	**-3743.0**	**0.000**	**-3514.4**	**-1765.7**	**0.000**	**-27354.7**	**-21538.0**	**0.000**
TW-BV	BV	**-6368.2**	**-3855.9**	**0.000**	**-3965.6**	**-2049.9**	**0.000**	**-27347.5**	**-21514.5**	**0.000**
TW-BV	COPELAND	**-8552.0**	**-5400.5**	**0.000**	**-5143.7**	**-2797.1**	**0.000**	**-29182.7**	**-21888.2**	**0.000**
TW-BV	KEMENY	**-8630.7**	**-5460.7**	**0.000**	**-4761.6**	**-2674.6**	**0.000**	**-44543.8**	**-37169.4**	**0.000**
DW-BV	BV	**-260.6**	**-16.7**	**0.026**	**-558.1**	**-177.3**	**0.000**	-136.7	167.2	0.844
DW-BV	COPELAND	**-2639.3**	**-1366.3**	**0.000**	**-1839.8**	**-821.0**	**0.000**	-2622.9	444.6	0.164
DW-BV	KEMENY	**-2719.4**	**-1425.0**	**0.000**	**-1595.1**	**-561.1**	**0.000**	**-18057.6**	**-14763.0**	**0.000**
BV	COPELAND	**-2459.8**	**-1268.6**	**0.000**	**-1361.5**	**-564.0**	**0.000**	-2625.3	416.4	0.155
BV	KEMENY	**-2543.1**	**-1324.2**	**0.000**	**-1251.7**	**-169.1**	**0.010**	**-18060.9**	**-14790.3**	**0.000**
COPELAND	KEMENY	-222.4	83.6	0.374	-339.2	843.8	0.403	**-16663.1**	**-13979.1**	**0.000**

Table 4. Leader board for the Handcrafted category of problem instances

Approach		Solver Name	Nr. Solved	Cumulative Time on Solved Instances (s)
Oracle	1	VBR	114.9 (± 1.4)	24703.4 (± 4972.5)
CBR	2	TW-BV	110.5 (± 2.2)	25668.9 (± 5855.8)
	3	BAS	109.5 (± 2.7)	26147.9 (± 7106.6)
	4	DW-BV	109.3 (± 1.8)	26239.2 (± 6779.6)
	5	BV	109.2 (± 1.8)	25561.7 (± 6541.9)
	6	IRV	108.0 (± 1.8)	23872.8 (± 6121.4)
	7	COOMBS	107.9 (± 2.5)	24553.9 (± 6317.8)
	8	KEMENY	107.8 (± 2.4)	24163.2 (± 6683.3)
	9	BUCKLIN	107.6 (± 2.5)	22892.7 (± 6783.0)
	10	COPELAND	107.6 (± 2.5)	24060.7 (± 6140.9)
SAT	11	minisat20SAT07	88.1 (± 4.1)	33700.5 (± 9455.9)
	12	mxc08	86.2 (± 3.9)	30312.6 (± 9620.1)
	13	march_dl2004	85.1 (± 3.5)	22245.3 (± 7770.6)
	14	picosat846	84.0 (± 3.6)	28713.1 (± 8167.2)
	15	minisat2.0	82.5 (± 4.3)	32019.3 (± 7527.4)

Table 5. Leader board for the Industrial category of problem instances

Approach		Solver Name	Nr. Solved	Cumulative Time on Solved Instances (s)
Oracle	1	VBR	113.1 (± 2.5)	24561.0 (± 5164.6)
CBR	2	BAS	110.3 (± 3.3)	30003.9 (± 4674.8)
	3	TW-BV	109.8 (± 3.0)	27742.0 (± 4430.4)
	4	KEMENY	105.4 (± 3.7)	26500.6 (± 5287.4)
	5	DW-BV	105.1 (± 3.4)	25699.3 (± 4611.6)
	6	BV	105.0 (± 3.5)	26364.8 (± 4243.5)
	7	COPELAND	104.5 (± 4.1)	26650.9 (± 5209.8)
	8	COOMBS	104.2 (± 3.9)	26758.2 (± 6411.9)
	9	IRV	103.6 (± 3.7)	26317.2 (± 5540.3)
	10	BUCKLIN	102.6 (± 4.5)	25574.9 (± 5617.2)
SAT	11	mxc08	101.8 (± 3.9)	30144.6 (± 6091.5)
	12	picosat846	96.4 (± 3.7)	29688.2 (± 6551.8)
	13	rsat20	93.8 (± 4.8)	34573.7 (± 6666.6)
	14	minisat20SAT07	89.8 (± 3.2)	31467.8 (± 8382.5)
	15	minisat2.0	87.2 (± 2.4)	34332.8 (± 7641.0)

Table 6. Leader board for the Random category of problem instances

Approach		**Solver Name**	**Nr. Solved**	**Cumulative Time on Solved Instances (s)**
Oracle	1	VBR	227.6 (± 1.4)	28960.0 (± 5745.6)
CBR	2	BAS	216.7 (± 3.2)	30463.4 (± 5284.6)
	3	TW-BV	211.8 (± 2.6)	25250.1 (± 4005.0)
	4	COOMBS	206.2 (± 3.1)	24303.7 (± 4554.8)
	5	IRV	206.1 (± 3.9)	25405.7 (± 5249.5)
	6	COPELAND	205.8 (± 3.1)	24590.6 (± 5769.9)
	7	DW-BV	205.6 (± 3.5)	24019.9 (± 4382.9)
	8	BV	205.4 (± 3.5)	23753.5 (± 4606.5)
	9	BUCKLIN	203.2 (± 3.9)	24111.5 (± 5774.5)
	10	KEMENY	194.0 (± 4.5)	27713.1 (± 4651.7)
SAT	11	march_dl2004	149.8 (± 6.3)	38318.5 (± 4934.1)
	12	gnoveltyplus	148.1 (± 6.0)	21884.0 (± 6398.8)
	13	SATenstein_T7	146.8 (± 6.4)	23124.2 (± 3713.5)
	14	ranov	146.0 (± 6.4)	19454.8 (± 4909.5)
	15	SATenstein_swgcp	142.7 (± 7.5)	16795.1 (± 5327.9)

Both BAS and TW-BV portfolios perform consistently well, which would not be obvious a-priori in this setting in which it is most important to solve instances within a cut-off. Once again, the Kemeny ranking is not competitive amongst the CBR-based portfolios.

These results are consistent with the expectations of experts in the field of SAT. It is regarded as a challenge to be able to select a good performing solver for a given instance, and the choice is heavily reliant on the experience of the user who makes this choice. Therefore, this domain is perfect for CBR, and the results demonstrate that it is also a very useful technique to use here.

6 Conclusions and Future Work

In this paper we studied a variety of adaptation schemes for a family of CBR-based algorithm portfolios for the SAT problem. Our results demonstrate that the choice of adaptation scheme is important for performance with schemes that consider run time rather that relative ranking gives superior performance.

We demonstrated that a CBR approach to this task is competitive, and out-performs individual high-performing SAT solvers in a wide variety of problem domains. A feature of the domain of SAT, and constraint solving in general, is that experience is important. This paper demonstrates that CBR has a lot to offer the SAT community.

Acknowledgement. This work is partly supported by Science Foundation Ireland Grant 10/IN.1/I3032 and FP7 FET-Open Grant 284715.

References

1. Biere, A., Heule, M.J.H., van Maaren, H., Walsh, T. (eds.): Handbook of Satisfiability. Frontiers in Artificial Intelligence and Applications, vol. 185. IOS Press (February 2009)
2. Gomes, C.P., Selman, B.: Algorithm Portfolios. Artificial Intelligence 126(1-2), 43–62 (2001)
3. O'Mahony, E., Hebrard, E., Holland, A., Nugent, C., O'Sullivan, B.: Using Case-based Reasoning in an Algorithm Portfolio for Constraint Solving. In: Proceedings of AICS (2008)
4. Xu, L., Hutter, F., Hoos, H.H., Leyton-Brown, K.: SATzilla: Portfolio-based Algorithm Selection for SAT. Journal of Artificial Intelligence Research, 565–606 (June 2008)
5. Pulina, L., Tacchella, A.: A Multi-engine Solver for Quantified Boolean Formulas. In: Bessière, C. (ed.) CP 2007. LNCS, vol. 4741, pp. 574–589. Springer, Heidelberg (2007)
6. Davis, M., Logemann, G., Loveland, D.: A Machine Program for Theorem Proving. Communications of the ACM 5(7), 394–397 (1962)
7. Fürnkranz, J., Hüllermeier, E.: Pairwise Preference Learning and Ranking. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) ECML 2003. LNCS (LNAI), vol. 2837, pp. 145–156. Springer, Heidelberg (2003)
8. Dwork, C., Kumar, R., Naor, M., Sivakumar, D.: Rank Aggregation Methods for the Web. In: Proceedings of the 10th International Conference on World Wide Web, WWW 2001, pp. 613–622. ACM, New York (2001)
9. Brinker, K., Fürnkranz, J., Hüllermeier, E.: A Unified Model for Multilabel Classification and Ranking. In: Proceedings of the 2006 European Conference on Artificial Intelligence, ECAI 2006, pp. 489–493. IOS Press (2006)
10. Brinker, K., Hüllermeier, E.: Case-based Multilabel Ranking. In: Proceedings of the 20th International Joint Conference on Artificial intelligence, IJCAI 2007, pp. 702–707. Morgan Kaufmann Publishers Inc., San Francisco (2007)
11. Brinker, K., Hüllermeier, E.: Case-Based Label Ranking. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 566–573. Springer, Heidelberg (2006)
12. Gärtner, T., Vembu, S.: Label Ranking Algorithms: A Survey. In: Johannes Fürnkranz, E.H. (ed.) Preference Learning. Springer (2010)
13. Pacuit, E.: Voting Methods. In: Zalta, E.N. (ed.) The Stanford Encyclopedia of Philosophy, Winter 2011 edn. (2011)

A Local Rule-Based Attribute Weighting Scheme for a Case-Based Reasoning System for Radiotherapy Treatment Planning

Rupa Jagannathan and Sanja Petrovic

Automated Scheduling, Optimisation and Planning Research Group
School of Computer Science, University of Nottingham, Nottingham, UK
{psxrj,sanja.petrovic}@nottingham.ac.uk

Abstract. This paper presents a novel local rule-based weighting scheme to determine attribute weights in a case-based reasoning system for radiotherapy treatment planning in brain cancer. A novel method of generating IF THEN rules to assign local weights to case attributes used in the nearest neighbour similarity measure is presented. The rules are prescreened using the data mining evaluation measures of confidence and support. Unique rules are then selected from the set of prescreened rules using an instance weighting algorithm that is based on a novel concept called the random retrieval probability of a training case, which is introduced to give an indication of the validity of the feedback obtained from a successful retrieval with respect to a particular training case. Experiments using real world brain cancer patient data show promising results.

Keywords: Case-based reasoning, radiotherapy treatment planning, local weights, weighted nearest neighbor similarity measure, rule selection.

1 Introduction

The weighted nearest neighbour method (wNN) is a popular similarity measure commonly used in case-based reasoning (CBR) systems [1]. The accuracy of the wNN algorithm depends highly on finding appropriate attribute weights [2]. Most inference engines use global weights, i.e. the weight of an attribute remains constant over the run of the algorithm or over the domain [3]. However, the significance of an attribute can vary across its instance space and also across the instance space of the other attributes. It has been shown that in human reasoning the importance of an attribute changes depending on the context or the values of other attributes as explained by Aha and Goldstone [4]. This kind of human reasoning can be transferred to artificial intelligence methodologies. In the CBR system under development, a local weighting scheme is used, where each attribute of the wNN similarity measure is weighted based on the values of the attributes in the target case.

In our work, the effect of an attribute value on its significance and that of the other case attributes is determined using a wrapper method, which utilises feedback on the performance of the system to make design decisions. A set of IF THEN rules that assigns the local attribute weights is generated based on performance feedback of the retrieval mechanism on training cases. The rules are pre-screened using the product of two data mining rule evaluation measures called *support* and *confidence* [5].

Wrapper methods are popular in attribute weighting algorithms since they incorporate the actual inference or classification engine [6], take into account the correlation between features and are simple to design [7]. A disadvantage of wrapper methods, however, is that they depend heavily on the training data available. Training data is often scarce or biased, in particular if the cases consist of real world data. In this case, a successful or correct retrieval does not always indicate that the weights used are truly representative of the significance of the attribute since correct retrieval depends not only on the design of the similarity measure but also on the availability of relevant cases in the case base. For this reason, we have designed a novel instance weighting algorithm that gives an indication of the validity of the feedback obtained from a successful retrieval. This factor, which we have coined the random retrieval probability is used to select a rule from the set of pre-screened rules.

The developed local rule-based weighting scheme is used in a CBR system for radiotherapy treatment planning for brain cancer. CBR has been widely applied in clinical applications. Begum et al. provide a good overview of CBR systems in the health sciences [8]. However, so far, the use of CBR in radiotherapy treatment planning has only been very sparsely studied [9, 10]. Radiotherapy is a form of cancer treatment in which tumour cells are destroyed by subjecting them to ionizing radiation. However, since excessive radiation adversely affects all cells, including healthy tissue and critical organs, a detailed treatment plan is required for each patient that describes exactly how a patient is irradiated in order to deliver a tumouricidal radiation dose over the tumour region while minimising the radiation received by healthy tissue and critical organs in the vicinity of the tumour. Oncologists use their subjective experience and expert clinical knowledge to generate the plan parameters using a trial and error approach that can take from a few hours to several days in complicated cases. The clinical decision support system for radiotherapy planning that we are developing uses case-based reasoning to capture this kind of non-quantifiable knowledge to aid oncologists in the computation of plan parameters. Since the manual planning process is highly intuitive and empirical determining the case attributes and their weights is not straightforward. Attributes that appear to capture the manual planning process have been tentatively identified in consultation with radiotherapy planners but the relevance and significance of these attributes to the automated decision support system has to be determined. This motivated the research presented in this paper.

The CBR system is developed in collaboration with the City Hospital, Nottingham University Hospitals NHS Trust, UK. Real-world data on brain cancer patients are used in the experiments.

The paper is organized as follows. A brief overview of the relevant literature and background to this work is provided in section 2. Section 3 describes the CBR system for radiotherapy planning for brain cancer. The weighted nearest neighbour similarity measure is presented in section 4. Section 5 discusses the local weighting scheme developed for the CBR system. The rule generation process, the pre-screening of rules using the rule evaluation measures of support and confidence and the final rule selection using the random retrieval probability are described. Experimental results on real world brain cancer patient cases comparing the performance of the nearest neighbour similarity measure using global weights with local weights are presented in section 6. Section 7 concludes this paper and discusses future research directions.

2 Related Work

Global attribute weights have been widely studied in CBR and classification systems. A comprehensive review of weighting algorithms can be found in [3, 4]. Though most CBR systems use global weights, local weighting schemes can offer better retrieval accuracy. According to Ricci and Avesani [11], the use of a local similarity metric (called Asymmetric Anisotropic Similarity Metric), where features in a nearest neighbour algorithm are selected based on the values of other features improves both the accuracy of the similarity computation and also requires fewer cases in the case base to obtain the same accuracy. Bonzano et al. [12] compare the effectiveness of local and global weights applied to their CBR system for Air Traffic Control and find that local weights achieve a lower error rate. Howe and Cardie [13] argue that using a different attribute weight for each case instance might not always be applicable. They implement weighting on a coarser scale, where the attribute weights of a nearest neighbour algorithm are local with respect to a class. Similarly, in this work, attributes are weighted based on the class or cluster that their values belong to.

A drawback of local weights can be that the correlation or interaction between attributes is not always taken into account [14]. The interaction between attributes however is not always easy to model or determine. Our algorithm generates rules to assign local weights to attributes based on the attribute values of the target case by using performance feedback from the CBR system on training data. Since the rules are generated by considering all attributes of a case at the same time, they inherently take into account any correlation between them. The rules are prescreened and selected using rule evaluation measures. Rule evaluation methods have been widely studied in machine learning. Lavrac et al. [15] provide an overview of commonly used rule evaluation measures such as rules accuracy, sensitivity, specificity, coverage and satisfaction. Ishibuchi et al. [16] demonstrated that using the product of two rule evaluation measures, called the confidence and support vastly improved efficiency of rule selection. They obtained good classification results on different data sets. In this work, we used the product of the confidence and support as described by Ishibuchi et al. to prescreen generated rules.

An additional disadvantage of using local weights is that if the training data available is limited then the local weights might be spread too thin [2]. Also the information obtained from feedback using the training data could be biased if the training data does not cover all possible cases. In particular the nearest neighbor algorithm is sensitive to noisy data [17]. For this reason, we have developed an algorithm that uses instance weighting to extract more information from the available data. The algorithm takes into account the probability of a retrieval instance being successful due to a bias in the training data rather than the appropriateness of the chosen attribute weights in the similarity measure. We introduce a factor, called the average random retrieval probability that gives an indication of the quality of the feedback obtained from retrieval instances that were used to generate a rule. This factor is used to select relevant rules from the set of prescreened rules.

3 A CBR System for Radiotherapy Treatment Planning for Brain Cancer

This section describes the CBR under development and the case attributes.

The medical physicists at the City Hospital create the treatment plans primarily based on the relative location of the tumour (planning target volume or PTV) and organs at risk (OAR) structures outlined by the oncologist on the patient images. The OAR form the group I attributes and usually include the brain, lens, spinal cord, optical nerve and chiasm in the case of brain cancer cases. Roentgen [9], which is a CBR system for lung and thorax cancer, uses patient geometric descriptors as case attributes. A similar idea has been implemented in our CBR system. We have identified six geometrical descriptors, called group II attributes, which attempt to capture geometric information about the tumour and the spatial relationship between the tumor and OAR. The output of the CBR system consists of the treatment plan parameters. Currently, we consider the number of beams used to irradiate the patient and the beam angles. The group II attributes are listed below. The range of values of attributes *E, V, R, Dt* and *E* (explained below), found in the available training data, is given in table 2.

1. **Angle, *A*:** This is the angle, given in degrees from [0,360] interval, between the line connecting the centre of the PTV and the origin of the DICOM image coordinate system and the line connecting the centre of the OAR and the origin.
2. **Distance, *E*:** The distance is defined as the minimum edge to edge distance connecting the outline of the PTV and the OAR and it is given in mm.
3. **Volume, *V*:** The attribute refers to the volume of the PTV, given in mm^3.
4. **Body – PTV volume ratio, *R*:** This is the ratio of the PTV volume to the volume of the patient body.
5. **Body – PTV distance, *Dt*:** This attribute denotes the minimum edge to edge distance in mm between the outline of the PTV and the outline of the body.
6. **PTV – OAR Spatial Relationship, *P*:** This attribute defines the relative position of the PTV with respect to the OAR. It contains six labels: left, right, posterior, anterior, superior and inferior, which take binary values.

Figure 1 shows the main components of our CBR system. The case base contains patient cases of previously treated brain cancer patients. The cases in the case base consist of the case description in the form of group I and group II case attributes and the treatment plan parameters. Given a target case, the inference engine first filters out from the case base a group of cases, which are comparable to the target case with respect to group I attributes (in other words they consider the same OAR as the target

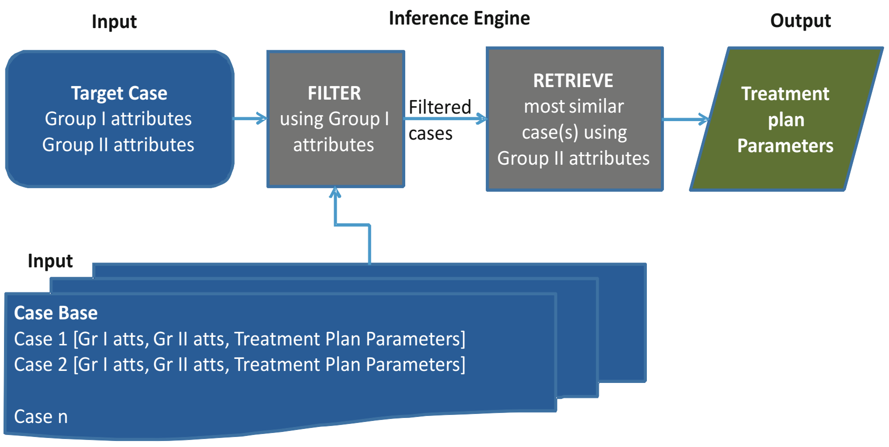

Input
Inference Engine
Output
Target Case
Group I attributes
Group II attributes
FILTER
using Group I attributes
Filtered cases
RETRIEVE
most similar case(s) using Group II attributes
Treatment plan Parameters
Input
Case Base
Case 1 [Gr I atts, Gr II atts, Treatment Plan Parameters]
Case 2 [Gr I atts, Gr II atts, Treatment Plan Parameters]
Case n

Fig. 1. Components of the CBR system for radiotherapy treatment planning

case). The filtered cases from the case base are then made available for similarity calculation. The similarity with respect to group II attributes between the target case and every case of the filtered case base is computed and the most similar case and its treatment plan are retrieved. The solution of the retrieved case is used as the solution of the target case. Two separate treatment plans are retrieved successively in two phases. In the first phase, a treatment plan suggesting the beam number is retrieved and in the second phase a treatment plan suggesting the beam angles is retrieved from the filtered case-base with plans with corresponding beam number suggested in the first phase (for details, please see [18]). The retrieved treatment plans can be presented to the medical physicists as a starting point for planning or can be passed on to the adaptation stage of the CBR system.

4 The Weighted Nearest Neighbour (WNN) Similarity Measure

The weighted nearest neighbour (wNN) similarity measure computes the aggregate similarity between two cases as the weighted sum of the individual attribute similarities. The attribute similarity is often expressed as a function of the distance between attribute values of two cases. A commonly used distance function in the wNN similarity measure is the Euclidian distance. Let C_T be the target case and C_C be a case from the case-base. The distance between C_T and C_C with respect to attribute l, where $l = A, E, V, R, Dt, P$, is given by expression (1):

$$Distance(C_T, C_C) = \sqrt{\textstyle\sum_l w_l \left(v_{T,l} - v_{C,l}\right)^2} \tag{1}$$

where $v_{T,l}$ and $v_{C,l}$ are the values of attribute l of C_T and C_C, respectively and w_l denotes the weight of attribute l. The similarity is then given by expression (2):

$$Similarity(C_T, C_C) = 1 - Distance \tag{2}$$

The global attribute weights are determined using a wrapper method, in which the weights determination is guided by feedback from the retrieval system. The training cases consist of cancer patients treated in the past. Therefore, the solution of these cases, that is the treatment plan parameters, beam number and beam angles, are known. The aim is to find attribute weights, which result in a low average retrieval error on the training cases.

The beam number error, E_{BN}, denotes the error or difference in the beam number between the retrieved treatment plan and the expected value (obtained from the original treatment plan of the target case). If the number of beams is exactly the same in both the retrieved treatment plan and the known treatment plan of the target case, the retrieval is deemed successful, i.e. $E_{BN} = 0$. The beam angles error, E_{BA}, denotes the average error in the beam angles between the retrieved treatment plan and the expected values (obtained from the original treatment plan of the target case). A successful retrieval is empirically set as $E_{BA} <= 30$ degrees, which is deemed as an acceptable starting point to fine-tune the angles either by a medical physicist or in the adaptation stage of the CBR system.

To compute optimal global attribute weights on our training cases we used a k-fold cross validation technique, where k is the number of folds. The training cases are divided into k sets. Each set of training cases is consecutively made the target set and the other (k -1) sets of cases are used in the case base. The similarity is calculated between the target cases and each case in the case base using expression (1) and (2). The difference or error in the beam parameters, i.e. E_{BN} and E_{BA}, is calculated between the treatment plan of the retrieved case and the original treatment plan of the target case. The results are averaged over k sets. To allow for variation in the distribution of training cases in the sets, this experiment is repeated three times; each time the k sets contain different cases. Again, the results are averaged. The entire experiment is repeated for each combination or set of weight values, where weights take values from the set {0, 0.5, 1}. In addition to the error value, the variance in error, V_{BN} and V_{BA}, is used as well, in order to avoid overfitting. The normalized error and variance value are averaged for each weight vector. The smallest average value, EV, of normalized error and variance indicates a good attribute weights vector.

We also studied how the number of folds k in the cross validation affects the error and variance. As expected, the variance increases considerably as the value of k increases but with little improvement in the retrieval. We, therefore, used k=4 to determine the results. The results showed that the attribute weight vectors resulting in the lowest value of EV with respect to the beam number were different to the lowest value of EV with respect to the beam angles, which is why two treatment plans are retrieved for each target case. In each retrieval, the appropriate beam number or beam angles weights are used. Table 1 shows the weight sets with respect to the beam number and beam angles retrieval that resulted in the smallest EV value and the best error-variance trade-off.

Finally, we also used k-fold cross validation to study the effect of the number of retrieved cases K, on the error, where K=1, 2, 3. We found that the best combination of error and variance was obtained when K =3 with respect to the beam number and K=1 with respect to the beam angles error. For more details please refer to [19].

Table 1. Attribute weights used for Beam number and Beam angles retrieval

	w_A	w_E	w_V	w_R	w_{Dt}	w_P
Beam Number	0.5	1	0.5	1	1	1
Beam Angles	1	1	1	1	0.5	0.5

5 Local Weights

In the previous section, we used global attribute weights in the similarity measure. Context sensitive attributes are weighted based on their own values and also on the values of other attributes of the target case. Plotting single attribute weights versus the resultant beam number and beam angles error, E_{BN} and E_{BA}, does not show a clear or direct relationship. However, as described by staff at the City Hospital, it is conceivable that the importance of attributes changes with respect to their own value or the value of other attributes. For instance, medical physicists pay special attention if the tumour volume (PTV) is small, and therefore the number of possible beam directions that both avoid the OAR and irradiate the PTV is limited. So if the target case has a small PTV, it makes sense to assign a large weight to the PTV to reflect the importance of a small PTV. Similarly, according to medical physicists, if the distance between the PTV and OAR is small, then the angles and position between the PTV and OAR becomes more important and consequently in a target case with a small distance value we could weight the angle and/or position between the PTV and OAR highly. In this manner, rules could be formulated such as *"IF V is small THEN $w_V = 1$"* or *"IF E is small THEN $w_A = 1$ AND $w_P = 1$".*

Previously, we plotted attribute weight values against the resultant average error both with respect to beam number and beam angles using a limited set of cases [18]. When visually examining the graphs, the error showed a variation with respect to the attribute weights. Rules were formulated that reflected attribute weights showing small error on the training data. Preliminary experiments show promising results [18].

Here, we present a more accurate and objective method that we have designed for our current work to learn the weight assignment rules based on specific evaluation criteria of the retrieval error obtained from training data. The attribute values of the training cases are first assigned to two groups or clusters *Large* or *Small* using the k-means clustering algorithm (as described in section 5.1). Then the rules are generated based on feedback about the retrieval performance on the training cases as described in section 5.2. The rules are pre-screened using two rule evaluation measures known as support and confidence (described in section 5.3). Finally, for each combination of attribute values clusters, a rule is selected based on the random retrieval probability described in section 5.4. A flowchart delineating clustering, rule generation, pre screening and selection is provided in figure 2.

5.1 Clustering

Since Howe and Cardie [13] suggested using different local weights for each attribute value is impractical and can lead to overfitting, we categorize the values of each attribute found in our case base into two groups, *Large or Small.* The local weights are then assigned to attribute value clusters rather than attribute values. The first step in determining the impact of attribute values on the six case attributes *A, E, V, R, Dt* and *P* is to define what attribute values constitute a large or small value for an attribute. The range of clusters *Large* and *Small* for each attribute is obtained from the training cases in the case base. Each case from the case base is assigned to one of the two clusters (*Large* or *Small*) based on the attribute value of the PTV-OAR distance *E* (large *E* or small *E*), PTV volume *V* (large *V* or small *V*), PTV-OAR volume ratio *R* (large *R* or small *R*) and body-PTV distance *Dt* (large *Dt* or small *Dt*). The PTV-OAR angle *A* and the position *P* are not strictly monotonically increasing, which makes it difficult to assign them to the groups *Large* and *Small.* For this reason, their effect on the case attributes is currently not considered in this study. The effect of the other four attributes on *A* and *P,* however, is taken into account. This means that each case is a member of four clusters based on the values of attributes *E, V, R*, and *Dt.*

Clustering can be defined as the unsupervised classification of objects, where unlabeled data is separated into discrete clusters [20]. Objects in one cluster are similar, while objects in different clusters are dissimilar. A widely used clustering technique is the k-means algorithm, which is popular due to its ease of implementation, simplicity and efficiency. The k-means algorithm groups objects by minimizing the squared Euclidian distance between the mean of each cluster and the objects in the cluster [21].

The clusters are determined for each attribute separately. In this work, we use the k-means function from the MATLAB statistical toolbox. This function iteratively partitions the input data with the aim of minimizing the total sum over all clusters obtained by summing the distances of each object within a cluster to the cluster centroid. The input to the k-means clustering function consists of the values of an attribute of all cases in the case base. The distance between objects in a cluster and the cluster centroid is set to be the Euclidian distance. The k-means function is run with 5 replicates. In each replicate a different centroid point, randomly selected from the training data is used as starting point and then the partitions with the lowest total sum are determined. We also use an online update phase, which ensures that the solution is a local minimum. In other words moving any single object to a different cluster would increase the total sum of distances.

The clusters that are created are not equal in size. This however is acceptable since equal sized clusters result in less distinction between the clusters. However, we have restricted our work to only two clusters (*Large* and *Small*) per attribute since using more clusters results in the size of clusters being exceedingly small due to the limited available data for training. The ranges of attribute values found among the training cases and the centroid of the clusters *Large* and *Small* as determined by the k-means algorithm for each attribute are shown in table 2. Each case is now represented by a vector of attribute clusters [E_{Cl}, V_{Cl}, R_{Cl}, Dt_{Cl}], where the subscript *CL* = [*Large, Small*] denotes the cluster that its attribute value belongs to.

Table 2. Range of values and centroids of clusters *Large* and *Small* for attributes *E, V, R* and *Dt*

	Attribute Range	**Centroid**$_{Small}$	**Centroid**$_{Large}$
E	2.0mm - 99.8mm	18.95	53.02
V	24.9mm^3 -729.8mm^3	194.35	417.70
R	6.4 - 255.9	19.68	158.93
Dt	0.06mm - 53.7mm	9.38	44.70

5.2 Generation of Rules

Once the case attributes *E, V, R*, and *Dt* of all cases are assigned to clusters *Large* and *Small*, the next step is to determine what effect a *Large* or *Small* attribute value has on the significance of the case attributes. The effect is formulated in the form of IF THEN rules. First, the candidate IF THEN rules are generated and then a number of rules are selected based on rule evaluation measures. A rule R_q, where $q=1,2,..n_A$, n_A = number of rules or antecedents, can be expressed in the following form:

$$\text{IF } [E_T, V_T, R_T, Dt_T] = A_q \text{ THEN } [w_{A,q}, w_{E,q}, w_{V,q}, w_{R,q}, w_{Dt,q}, w_{P,q}] = W_q$$

Where, $[E_T, V_T, R_T, Dt_T]$ denotes the attribute cluster vector of the target case, A_q = $[E_q, V_q, R_q, Dt_q]$ is the antecedent of rule R_q, and weight vector, $W_q = [w_{A,q}, w_{E,q}, w_{V,q}, w_{R,q}, w_{Dt,q}, w_{P,q}]$ is the consequent of rule R_q. The weights, $w_{l,q,}$ can take values from {0, 0.5, 1}, where $l = A, E, V, R, Dt, P$.

Let n_A be the number of possible antecedents and n_W be the number of possible consequents that can be formulated. Since each antecedent vector consists of four attributes (*E, V, R* and *Dt*), which can take one of two possible values (*Large* or *Small*), $n_A = 2^4 = 16$. Therefore, we require 16 rules for weights assignment. Since the weights can take values from the set: [0, 0.5, 1], the number of consequents available for six attributes (*A, E, V, R, Dt* and *P*) is $n_W = 3^6 = 729$. This means that the number of rules that are generated by a combination of antecedents and consequents is: $n_R=n_A * n_W = 11664$.

Given an antecedent A_q, $q=1,2,..n_A$, all training cases that are compatible with antecedent A_q are identified and then used as the set of target cases $S_{A,q}$ during cross validation of rule q. In order to determine the retrieval error obtained with each rule, i.e. each antecedent-consequent combination, we use the leave-one-out strategy commonly employed in case-based reasoning systems. Each of the training cases in the identified set $S_{A,q}$ is consecutively made the target case and the remaining cases constitute the case base. For each target case, the most similar case in the case base is retrieved using expressions (1) and (2), where the consequent W_r, $r = 1,2,.. n_{W,}$ supplies the attribute weights vector $[w_{A,r}, w_{E,r}, w_{V,r}, w_{R,r}, w_{Dt,r}, w_{P,r}]$. The retrieval error with respect to E_{BN} and E_{BA} is computed for each antecedent-consequent combination. A rule R_q is deemed feasible with respect to a target case, if the antecedent A_q matches the attribute values of the case and if the weights used in the similarity measure, supplied by consequent W_r result in a successful retrieval. During beam number retrieval, a successful retrieval occurs if the beam number in the retrieved plan is the same as the beam number in the known treatment plan of the target case, i.e. if E_{BN} =0. During beam angles retrieval, a successful retrieval occurs if the average difference in beam angles in the retrieved plan and the known treatment plan of the target case is smaller than the preset threshold, i.e. E_{BA} <= 30deg. The rules, which result in successful retrieval on the training cases constitute the feasible rules.

5.3 Pre-screening Using Rule Evaluation Measures

It may happen that more than one consequent is associated with the same antecedent. From the set of feasible rules we need to find a limited set of 16 rules, which will uniquely assign a consequent or weight vector to each antecedent or attribute values vector. In order to determine the most appropriate and relevant rules evaluation measures are used as constraints. Two rule evaluation measures commonly used in data mining are the *confidence* and *support* of a rule. [16].

If D is a set, containing m training cases, then $D(A_q)$ is the number of cases, which are compatible with antecedent A_q and $[D(A_q) \cap D(W_q)]$ is the number of cases that are compatible with both antecedent A_q and consequent W_q. In other words, $[D(A_q) \cap D(W_q)]$ represents the number of cases with attribute values E_q, V_q, R_q, Dt_q in which the retrieval was successful when weights $w_{A,q}$, $w_{E,q}$, $w_{V,q}$, $w_{R,q}$, $w_{Dt,q}$, $w_{P,q}$ were used in the wNN similarity measures described by expressions (1) and (2).

The confidence, *con,* measures the validity of rule A_q. It is the percentage of all cases compatible with antecedent A_q that are also compatible with consequent W_q.

$$con\left(A_q \Rightarrow W_q\right) = \frac{|D(A_q) \cap D(W_q)|}{|D(A_q)|} \tag{3}$$

The support, *sup,* measures the coverage of rule R_q. It is the percentage of all training cases, which are compatible with both antecedent A_q and consequent W_q.

$$sup\left(A_q \Rightarrow W_q\right) = \frac{|D(A_q) \cap D(W_q)|}{\mathrm{m}} \tag{4}$$

Though the confidence and support can directly be used as evaluation measures, according to Ishibuchi and Yamamoto [16] the confidence criterion selects rules, which cover only a small number of compatible training cases but have a low retrieval error. The support criterion selects rules based on many compatible training cases but could result in a high retrieval error. They found that they obtained a good trade-off between generalisability and retrieval error on various different data sets when using the product *CSP* of the confidence and support.

$$CSP\left(A_q \Rightarrow W_q\right) = \mathrm{con}\left(A_q \Rightarrow W_q\right) \times \mathrm{sup}\left(A_q \Rightarrow W_q\right) \tag{5}$$

For each antecedent, the rules with the highest *CSP* are selected. Then, among the pre-screened rules, one rule is selected for each antecedent using the random retrieval probability (*PRR*) rule evaluation measure described in section 5.4.

5.4 Rule Selection Based on Instance Weighting Using Random Retrieval Probability

From the prescreened rules, a single rule per antecedent has to be selected. This is done by using a novel instance weighting algorithm that gives an indication of the quality of information gained from a retrieval instance. Not every successful retrieval indicates that a rule accurately describes the relationship between attribute significance and weights and will obtain good results outside the training phase when using unseen cases. If the case base is small or biased the average retrieval error based on the training cases can be skewed if the solution parameter values are not equally distributed. For example, let us assume that a large majority of treatment plans in the case base use four beams and let us further assume that the target case happens to have four beams as well. This will result in a low retrieval error even though the number of beams in practice might not usually be 4 as is indicated by the training cases. In that situation a low retrieval error might not be indicative of the performance of the retrieval mechanism but might just mean that the probability of retrieving a case with the correct solution is uncharacteristically high. Therefore, we require a way to quantify the validity of a successful retrieval as opposed to a random retrieval. In a random retrieval, a case is retrieved at random from the case base without calculating the similarity of cases. The random retrieval probability (*RRP)* of an instance refers to the probability of a random retrieval being successful. In other words, what is the likelihood of successful retrieval if a random case is retrieved from the case base (instead of the most similar case) given a particular target case? If *RRP* is small the information we infer from the instance when using the weighted similarity measure is valid. If *RRP* is high, then we do not know if the retrieval is successful due to correct weights used in the similarity measure or due to the fact that the likelihood of a randomly successful retrieval is high.

In our CBR system, we define the random retrieval probability *RRP*, which considers cases available after filtering based on OAR of the target case. For a given target case, let C_{Right} denote the number of cases in the filtered case base, where E_{BN} =0 in the beam number retrieval or $E_{BA} \leq$ 30deg in the beam angles retrieval. Let C_{Wrong} denote the number of cases, where $E_{BN} \neq 0$ or $E_{BA} > 30$ degrees. Then for a given target case, the random retrieval probability of a retrieval instance is given by:

$$RRP = \frac{C_{Right}}{C_{Right} + C_{Wrong}} \qquad (6)$$

where the number of cases in the filtered case base is given by $\left(C_{Right} + C_{Wrong}\right)$. For each rule (i.e. combination of antecedent and consequent) the number of successful retrieval instances over the training cases is noted. *RRP* of each successful retrieval instance is averaged to represent the average *RRP* of a rule. The average random retrieval probability of a rule constitutes the final rule evaluation measure to select a unique consequent for each antecedent. For a given antecedent, the consequent (from the set of prescreened rules) with the lowest average *RRP* is chosen. If more than one consequents result in the same lowest *RRP* value, then one of those consequents is chosen arbitrarily.

```
//Clustering
FOR each training case
    Assign attribute values $v_E, v_V, v_R, v_{Dt}$ to clusters Large or Small using k-means algorithm
//Rule generation
FOR each antecedent $A_q = [E_q, V_q, R_q, Dt_q]$ FROM 1 TO $n_A$
    FOR each consequent $W_r = [w_{A,r}, w_{E,r}, w_{V,r}, w_{R,r}, w_{Dt,r}, w_{P,r}]$ FROM 1 TO $n_W$
            FOR each target case $C_T$ FROM 1 TO $n$
                IF $[E_T, V_T, R_T, D_{T]} = A_q$ //antecedent matches target case attribute clusters
                Antecedent counter $cnt_{A,q} = cnt_{A,q} + 1$
        //Rule Pre screening
        Retrieve most similar case to target case using wNN similarity measure with
        $[w_{A,r}, w_{E,r}, w_{V,r}, w_{R,r}, w_{Dt,r}, w_{P,r}] = W_r$
        IF retrieved case has $E_{BN} = 0$ // during beam number retrieval [OR]
        IF retrieved case has $E_{BA} < 30$degrees // during beam angles retrieval
            retrieval = successful
            Rule counter $cnt_{Aq,Cr} = cnt_{Aq,Cr} + 1$
            Random retrieval probability of instance: $RRP_T = C_{Right}/(C_{Right} + C_{Wrong})$
    Confidence con = $cnt_{Aq,Cr} / cnt_{A,q}$
    Support sup = $cnt_{Aq,Cr} / n$
    Confidence support product $CSP = con * sup$
    //Rule Selection
    Average $RRP$ of rule $R_q = \sum_{T=1,2,..n} RRP_T / n$
FOR each antecedent $A_q$ FROM 1 TO $n_A$
    select consequent $W_q$ with highest CSP and lowest RRP.
    Rule $R_q : A_q \Rightarrow W_q$
```

Fig. 2. Flowchart showing clustering, rule generation, prescreening and selection

6 Experimental Results

We tested the rule generation and evaluation algorithm for local weights assignment in the CBR system under development for radiotherapy treatment planning. The data consists of real brain cancer patients from the City Hospital. For generating the rules, we randomly selected 64 of these brain cancer cases as training cases, while a further 22 randomly selected cases constituted the test cases. From the set of 11664 possible rules that can be formed using all combinations of antecedents and consequents, 4340 rules resulted in successful beam number retrieval while 5096 rules resulted in successful beam angles retrieval. The two rule evaluation measures, i.e. the support-confidence product *CSP* and the average random retrieval probability *RRP*, were used to select 16 weights assignment rules for beam number retrieval and 16 rules for beam angles retrieval. The 22 test cases (with known treatment plans) were used as target cases and the most similar case was retrieved for each using the wNN similarity measure defined in expressions (1) and (2) with local weights assigned to the attributes of the target cases using the appropriate rules. The retrieval errors E_{BN} and E_{BA} for each target case were computed.

The average retrieval errors for E_{BN} and E_{BA} were compared with the average retrieval errors obtained using the global weights given in table 1.

Table 3 shows the average beam number error, E_{BN}, and the average beam angles error, E_{BA}, obtained using global weights (as described in section 4) and local weights over the test cases. The success rate refers to the percentage of test cases in which the retrieval was successful, that is $E_{BN} = 0$ and $E_{BA} \leq 30$ degrees.

We can see that the success rate increases markedly using local weights in comparison to global weights for both beam number and beam angle retrieval. The improvement in error and success rate is statistically significant for the beam angles retrieval with a p value of 0.02 and a confidence level of 98%. The improvement in error and success rate in the case of beam number retrieval has a p value of 0.08 and corresponding confidence level of 92%. The lower improvement in success rate seen in beam number retrieval than in beam angles retrieval is possibly due to attribute values possibly having a lower influence on the significance of case attributes with respect to the beam number. The results will be verified again once more test cases become available. However, the success rate obtained using the wNN similarity measure in the retrieval mechanism using local weights is deemed as a good starting point for adaptation, which is planned to be the next stage of a CBR system.

Table 3. Comparison of average beam number error, $E_{BN,}$ and beam angles error, $E_{BA,}$ using global weights and local weights in the wNN similarity measure

Weighting Method	**Average Error**	**Success rate**
Beam Number Retrieval		
Global weights	0.36	68.2%
Local weights	0.27	77.3%
Beam Angles Retrieval		
Global weights	32.28 degrees	59.1%
Local weights	25.04 degrees	72.7%

7 Conclusion

In this work, we present a local weighting scheme in which the attribute weights are assigned by rules based on the value of the case attributes. The rules are generated using a supervised learning approach in which feedback about the retrieval success of the wNN similarity measure on training cases is used to guide the weights determination. The rules are prescreened using the data mining rule evaluation measure of the product of the confidence and support. A unique rule for each antecedent is then selected based on the average random retrieval probability. Statistical experiments carried out using real world brain cancer patient cases show an improvement in the success rate of the retrieval mechanism when using local weights assigned using the generated rules instead of using global weights, in particular for beam angle retrieval. Another advantage of this algorithm is that since, clustering and rule generation are done offline using the archived cases in the case base, they do not affect the retrieval time. Therefore, when presented with a target case, the algorithm runs quite quickly using the pre generated weight assignment rules. The clusters and rules, however, can be updated when a large number of cases has been added to the case base.

Currently, the attribute values are assigned to the crisp classes *Large* or *Small*, obtained using the k-means clustering algorithm. However, since the attribute values are continuous the boundaries of each class are artificially generated. In the future, we plan to use fuzzy sets to obtain a more accurate representation of how large or small an attribute value is and how its value affects the significance of attribute weights.

The work presented in this paper concentrates on the retrieval mechanism. The next stage in a CBR is the adaptation module, which adapts the solution of the retrieved case with respect to the specific requirements of the target case. In radiotherapy treatment planning, adaptation can be done by adjusting the beam configuration according to the geometric displacement in the location of the tumor and OAR structures of the target case compared to the retrieved case or based on an evaluation of the resulting radiation dose profile. Two possible approaches for adaptation include rule-based adaptation and case-based adaptation.

References

1. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory 13, 21–27 (1967)
2. Bonzano, A., Cunningham, P., Smyth, B.: Learning Feature Weights for CBR: Global versus Local. In: Lenzerini, M. (ed.) AI*IA 1997. LNCS, vol. 1321, pp. 417–426. Springer, Heidelberg (1997)
3. Wettschereck, D., Aha, D.W., Mohri, T.: A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. Artificial Intelligence Review 11, 273–314 (1997)
4. Aha, D.W., Goldstone, R.L.: Concept Learning and Flexible Weighting. In: Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society, pp. 534–549. Erlbaum (1992)
5. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules in Large Databases. In: Proceedings of the 20th International Conference on Very Large Data Bases, pp. 487–499. Morgan Kaufmann Publishers Inc. (1994)
6. Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics 23, 2507–2517 (2007)
7. Wettschereck, D., Aha, D.: Weighting Features: Case-Based Reasoning Research and Development. In: Aamodt, A., Veloso, M.M. (eds.) ICCBR 1995. LNCS, vol. 1010, pp. 347–358. Springer, Heidelberg (1995)
8. Begum, S., Ahmed, M.U., Funk, P., Ning, X., Folke, M.: Case-Based Reasoning Systems in the Health Sciences: A Survey of Recent Trends and Developments. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 41, 421–434 (2011)
9. Berger, J.: Roentgen: Radiation Therapy and Case-Based Reasoning. In: Proceedings of the Tenth Conference on Artificial Intelligence for Applications, pp. 171–177 (1994)
10. Petrovic, S., Mishra, N., Sundar, S.: A novel case based reasoning approach to radiotherapy planning. Expert Systems with Applications 38, 10759–10769 (2011)
11. Ricci, F., Avesani, P.: Learning a Local Similarity Metric for Case-Based Reasoning. In: Aamodt, A., Veloso, M.M. (eds.) ICCBR 1995. LNCS, vol. 1010, pp. 301–312. Springer, Heidelberg (1995)

12. Bonzano, A., Cunningham, P., Smyth, B.: Using Introspective Learning to Improve Retrieval in CBR: A Case Study in Air Traffic Control. In: Leake, D.B., Plaza, E. (eds.) ICCBR 1997. LNCS, vol. 1266, pp. 291–302. Springer, Heidelberg (1997)
13. Howe, N., Cardie, C.: Examining Locally Varying Weights fo Nearest Neighbour Algorithms. In: Leake, D.B., Plaza, E. (eds.) ICCBR 1997. LNCS, vol. 1266, pp. 455–466. Springer, Heidelberg (1997)
14. Stanfill, C., Waltz, D.: Toward memory-based reasoning. Commun. ACM 29, 1213–1228 (1986)
15. Lavrač, N., Flach, P.A., Zupan, B.: Rule Evaluation Measures: A Unifying View. In: Džeroski, S., Flach, P.A. (eds.) ILP 1999. LNCS (LNAI), vol. 1634, pp. 174–185. Springer, Heidelberg (1999)
16. Ishibuchi, H., Yamamoto, T.: Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining. Fuzzy Sets and Systems 141, 59–88 (2004)
17. Brighton, H., Mellish, C.: Advances in Instance Selection for Instance-Based Learning Algorithms. Data Mining and Knowledge Discovery 6, 153–172 (2002)
18. Jagannathan, R., Petrovic, S., McKenna, A., Newton, L.: A Novel Two Phase Retrieval Mechanism For A Clinical Case-Based Reasoning Systems For Radiotherapy Treatment Planning. International Journal on Artificial Intelligence Tools (2012)
19. Jagannathan, R., Petrovic, S., McKenna, A., Newton, L.: A Novel Two-Phase Retrieval Mechanism for a Clinical Case-Based Reasoning System for Radiotherapy Planning. In: Petrovic, D., Syntetos, A. (eds.) OR53, pp. 17–22. OR Society, Nottingham (2011)
20. Rui, X., Wunsch, D.: II: Survey of clustering algorithms. IEEE Transactions on Neural Networks 16, 645–678 (2005)
21. Jain, A.K.: Data clustering: 50 years beyond K-means. Pattern Recognition Letters 31, 651–666 (2010)

Learning and Reusing Goal-Specific Policies for Goal-Driven Autonomy

Ulit Jaidee[1], Héctor Muñoz-Avila[1], and David W. Aha[2]

[1] Department of Computer Science & Engineering Lehigh University Bethlehem, PA 18015
[2] Navy Center for Applied Research in AI
Naval Research Laboratory (Code 5514) Washington, DC 20375
ulj208@lehigh.edu, munoz@cse.lehigh.edu,
david.aha@nrl.navy.mil

Abstract. In certain adversarial environments, reinforcement learning (RL) techniques require a prohibitively large number of episodes to learn a high-performing strategy for action selection. For example, Q-learning is particularly slow to learn a policy to win complex strategy games. We propose GRL, the first GDA system capable of learning and reusing goal-specific policies. GRL is a case-based goal-driven autonomy (GDA) agent embedded in the RL cycle. GRL acquires and reuses cases that capture episodic knowledge about an agent's (1) expectations, (2) goals to pursue when these expectations are not met, and (3) actions for achieving these goals in given states. Our hypothesis is that, unlike RL, GRL can rapidly fine-tune strategies by exploiting the episodic knowledge captured in its cases. We report performance gains versus a state-of-the-art GDA agent and an RL agent for challenging tasks in two real-time video game domains.

Keywords: Case-based learning, reinforcement learning, goal-driven autonomy.

1 Introduction

Reinforcement learning (RL) algorithms attempt to optimize an agent's behavior when interacting in an environment. The agent's behavior is defined by a policy π, which maps for every state s and action a the value $\pi(s,a)$ of selecting a in s. The optimization function is frequently defined as the summation of future rewards, which in our context of adversarial games can be defined as the difference in score between the agent and its opponents. Thus, an RL agent seeks to maximize this difference.

A difficulty arises in situations where the adversaries frequently change their strategies or, equivalently, when each adversary has a fixed and unique strategy and adversaries are frequently changed without divulging their identity to the RL agent. In such situations, the RL agent will learn a "maximum common denominator" strategy that is optimal regardless of the opponent. Unfortunately, as evidenced in our empirical study (Section 6), either (1) this strategy performs poorly or (2) learning it requires a prohibitively large number of episodes.

To address this difficulty, we propose to use case-based reasoning (CBR) techniques to augment RL's cycle by acquiring and reusing cases that capture episodic knowledge about an agent's (1) expectations, (2) goals to pursue when discrepancies exist (i.e., where these expectations are not met), and (3) actions for achieving these goals in given states. Agents that reason with expectations, discrepancies, and goals are the focus of *goal reasoning*.

In this paper, we introduce GRL (Goal Reasoning Learner), a case-based Goal-Driven Autonomy agent. Goal-Driven Autonomy (GDA) is a reflective model of goal reasoning that controls the focus of an agent's planning activities by dynamically resolving unexpected discrepancies in the world state (Molineaux *et al.*, 2010).Unlike previous work integrating RL and CBR and work on GDA (see Section 2), GRL embeds GDA in an RL cycle by learning and reusing the three kinds of cases mentioned above. GRL is the first GDA agent capable of learning and reusing goal-specific policies. Our hypothesis is that, as a result of this capability, GRL can fine-tune strategies by exploiting the episodic knowledge captured in its cases. We report performance gains versus a state-of-the-art GDA agent and an RL agent for challenging tasks in two real-time gaming domains.

Section 2 discusses related work. Sections 3 and 4 then describe GRL's GDA instantiation and its detailed algorithm. Section 5 presents an example of GRL's behavior. We present our empirical studies in Section 6 and conclude in Section 7.

2 Related Work

GDA agents use a four-step strategy to respond competently to unexpected situations in their environment: (1) detect any discrepancy between the observed state and the expected state(s), (2) explain this discrepancy, (3) formulate a goal to resolve it (if needed), and (4) manage this new goal along with its pending goals (Molineaux *et al.*, 2010; Muñoz-Avila *et al.*, 2010). In step 3, these agents use a variety of models to formulate new goals. For example, Intro (Cox, 2007) uses explanation patterns represented as *cause* $\rightarrow$ *effect* rules such that, if a state is judged to be a discrepancy and it maps to the effects of a rule, then Intro will select the negation of that rule's cause as its new goal. ARTUE (Molineaux *et al.*, 2010) uses rule-based reasoning for goal formulation and ranking (i.e., pending goals are maintained in a priority list). Its rules encode expert knowledge in a manner similar to Intro's rules, but ARTUE adds a more robust process by encoding planning dependencies in a truth-maintenance system. EISBot (Weber *et al.*, 2010b) instead uses a case-based model to formulate goals, where a case $c_i=(c_{i,1},\ldots,c_{i,n})$ is an expert-provided sequence of states for accomplishing a task, and states are represented as a vector of numeric values. Given current state c, EISBot retrieves a most similar state $c_{i,j}$ in its case base along with $c_{i,j+w}$, where w is the length of its planning window. It computes the difference $c_{i,j+w}$-$c_{i,j}$ and adds this to c to define its new goal. In contrast to these GDA agents, GRL *learns* its goal formulation knowledge.

T-ARTUE (Powell *et al.*, 2011) is an extension of ARTUE that interactively learns goal formulation knowledge; it can query the user to ask for new goals or confirm their formulation, and the user can provide feedback on these decisions. In contrast, GRL *automatically* learns goal formulation knowledge and new goals.

Most GDA agents (e.g., CB-gda (Jaidee *et al.*, 2010)) are given knowledge about state expectations, discrepancies, goals to achieve, and the means to achieve the goals (e.g., the plans or the policies). While some prior work has focused on learning some of these, GRL is the first GDA agent to learn ***all*** of them simultaneously. Finally, in contrast to most prior GDA work, GRL learns and reasons with *stochastic* expectations, which we define in Section 3.

Agents can compute state expectations using action models (i.e., their preconditions and effects) and the current state. Bouguerra *et al.* (2008) use description logics to model and infer expectations after executing a plan, which is particularly useful for partially observable environments. For example, an agent might observe John entering a vehicle at a location *A* and the vehicle later arriving at location *B*, where its occupants departed. Given this, it could infer that John arrived at *B*. GDA agents vary in how they compute expectations, including using a model of abstract explanation patterns (Cox, 2007), or by defining discrepancy detectors to trigger when state expectations fail (Weber *et al.*, 2010a). Unlike these (and most other) GDA agents, GRL *learns* its action models for computing state expectations. The only related agent is LGDA (Jaidee *et al.,* 2011a), which learns action models it uses to compute expectations but it assumes that the policies and goals are given as input. In contrast, ***GRL identifies new goals,*** and ***it learns and reuses goal-specific policies***.

There is substantial interest in integrating CBR and RL, as exemplified by Derek Bridge's ICCBR-05 invited talk on potential synergies between CBR and RL (Bridge, 2005), the SINS system that solves problems in continuous environments (Ram & Santamaria, 1997), and CBRetaliate, which stores and retrieves Q-tables (Auslander *et al.*, 2008). Most previous contributions focused on improving the performance of an agent by exploiting synergies among CBR and RL or by enhancing the CBR process by using RL (e.g., to improve similarity metrics). More recently, researchers have studied ways in which CBR can improve reinforcement learning. This includes reducing the memory requirements of RL (Dilts & Munoz-Avila, 2010), using cases as a heuristic to speed up the RL process (Bianchi *et al.,* 2009) and using cases to approximate state value functions in continuous spaces (Gabel & Riedmiller, 2005; 2007). GRL falls in this latter category; it uses CBR to fine-tune strategies by exploiting the episodic knowledge captured in the cases while embedded in the RL cycle. In this context, GRL's novelty is that it automatically identifies goals, learns policies specific to those goals, learn expectations about the action's outcomes, and reasons when a discrepancy occurs.

3 Case-Based Goal-Driven Autonomy with GRL

Like other GDA agents, GRL conducts a meta-process for online planning (Nau, 2007). Figure 1 illustrates GDA's information flow, which naturally embeds the standard RL model (Sutton & Barto 1998), where the objective is to maximize the expected return. The return is a function of the rewards obtained. For example, the return can be defined as the summation of the future rewards. Like RL, GRL executes an action *a* in the environment and observes from the environment the next state *s′* and a reward *r*. Unlike RL, which selects the next action based on the current policy (i.e., a mapping $S \rightarrow 2^{A\times[0,1]}$ from states to a probability distribution over the actions), GRL selects an action based on the current goal *g*, the policy *π* for that goal, and the current state *s*. A crucial challenge is that, in many environments, there is no optimal policy for all situations. For example, in an adversarial game, a policy might be optimal versus one opponent but not others. This in part motivates why GDA agents reason with expectations, discrepancies, and goals.

GRL learns and reasons with (1) an ***Expectation Case Base*** (ECB), which is a mapping $S \times A \rightarrow 2^{S\times[0,1]}$ from (state, action) pairs to a probability distribution over the expectations, (2) a ***Goal Formulation Case Base*** (GFCB), which is a mapping $G \times D \rightarrow 2^{G\times[0,1]}$ from (goal, discrepancy) pairs to a distribution over the expected values for formulated goals, and (3) a ***Policies Case Base*** (Π), which is a set of goal-policy pairs (*g*, *π*). We discuss the relation between *g* and *π* below.

GRL's ***Discrepancy Detector*** compares state observations with expectations *X*. If a discrepancy (i.e., an unexpected observation) $d \in D$ is found, then it is passed to the Goal Formulator. The discrepancy may warrant a change in the current goal. The ***Goal Formulator*** generates a goal $g \in G$ given a discrepancy and next state *s′*. The ***Goal Manager*** checks for opportunities to learn new goals that are not in *G*. In Section 4 we clarify how and when new goals are learned. Finally, ***Policy Learner*** learns policies for new goals and refines the policies of existing goals.

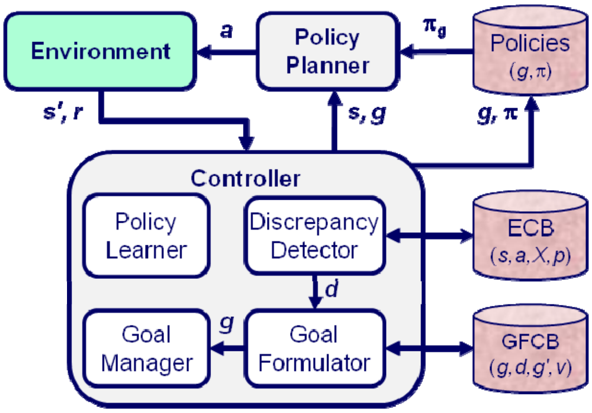


Fig. 1. Information flow in GRL

We now provide formal definitions for GDA elements. The ***expectations*** of an action *a* when executed in state *s* is a collection of states $S_{a,s} \subseteq S$. Each expectation $x \in S_{a,s}$ will occur with a non-zero conditional probability $p(s_{t+1}=x \mid s_t=s,\ a_t=a)$ (assuming the Markov property). A ***discrepancy*** occurs whenever executing *a* yields state $s' \notin S_{a,s}$. We assume that the explanation for a discrepancy reflects GRL's incomplete domain knowledge. Hence, it revises its expectations knowledge whenever a discrepancy occurs, as explained in Section 4.

The representation of a discrepancy depends on the state representation. Here we assume a state is represented as a vector $s=(v_1,\ldots,v_n)$, where v_i is a value of a feature f_i. We represent a discrepancy as a vector of Boolean values $d=(b_1,\ldots,b_n)$, where b_i is true iff the values in position i of the actual and expected states are the same.

Any state can be a goal. We are interested in particular goals that we call ***trajectory goals***, which are defined as follows. First, when an agent follows a policy it generates a ***trajectory***, which is the sequence of states that it visits. A ***trajectory goal relative to a policy*** π is any state along a trajectory produced by π from the start state to a terminal state. All pairs $(g,\pi) \in \Pi$ are such that g is a trajectory goal relative to π. Our policies behave like weak solutions (Ghallab *et al.*, 2004) in that there is no guarantee that a particular goal will be reached because policies are learned incrementally (i.e., when executing the policy a trajectory might be generated that does not contain the goal). Hence, many iterations may be needed before strong solutions are obtained (which guarantee that certain goals are always reached).

4 The GRL Algorithm

We now present GRL, which incrementally learns expectations, goal formulation knowledge, and goal-specific policies. GRL uses Q-learning as its RL algorithm. Q-learning is frequently used as the prototypical RL algorithm due to its bootstrapping capabilities, which enables it to estimate state-action values based on other state-action values estimates. As a result, it tends to converge to optimal policies faster than other RL methods (Sutton & Barto, 1998).

GRL receives as input the start state s_0, a waiting time Δ, the Policy Case Base Π, the Expectation Case Base (ECB), the Goal Formulation Case Base (GFCB), the actions A, and some parameters. The parameters α and γ are the step-size and discount-rate parameters for Q-learning. Parameters ε_1 and ε_2 are for the ε-greedy selection of action and goals, respectively. Parameters c_a and c_b are used to learn new goals as will be explained later, and t is a threshold used to determine when two goals are similar to one another. GRL runs one episode of a game and returns updated values for Π, ECB, and GFCB.[1]

GRL executes an iterative decision making cycle with the following steps: (1) identify discrepancies when they arise, (2) decide which goals to achieve to resolve any such discrepancies, and (3) perform actions to accomplish these goals. Simultaneously, GRL learns knowledge about state expectations, discrepancies, goals to achieve, and the actions to achieve these goals (e.g., goal-specific policies).

GRL has three phases: In Phase 1, which occurs during an episode, GRL uses and updates ECB and GFCB. Phases 2 and 3 occur immediately after an episode ends. In Phase 2 new goals are identified and in Phase 3 goal-specific policies are updated.

[1] We assume episodic tasks in which the instances eventually end. This is an effect of the underlying Q-learning. If the task did not terminate, then we would need to use mechanisms such as eligibility traces (Sutton & Barto, 1998).

```
GRL($s_0$, $\Delta$, $\Pi$, ECB, GFCB, $A$, $\alpha$, $\gamma$, $\varepsilon_1$, $\varepsilon_2$, $c_a$, $c_b$, $t$) =
// Phase 1: Online execution and updating
1:  $s \leftarrow x \leftarrow s_0$; $a \leftarrow l_a \leftarrow l_b \leftarrow \emptyset$;  $g \leftarrow g' \leftarrow g_0$;  $d \leftarrow d_0$;  $G \leftarrow$ GetGoals($\Pi$)
2:  while episode continues
3:      wait($\Delta$)
4:      $s' \leftarrow$ GetState()                       // Periodically observe the state
5:      $r \leftarrow U(s') - U(s)$                      // Compute the reward
6:      $l_a \leftarrow$ concat($l_a$ ,<$s'$>); $l_b \leftarrow$ concat($l_b$ ,<$s,a,s',r$>)
7:      ECB $\leftarrow$ Update(ECB, $s$, $a$, $s'$)    // Update ECB's distribution
8:      $a' \leftarrow$ Random(|$A$|)                    // Random current action
9:      if $\Pi \neq \emptyset$
10:         $q \leftarrow$ Get(GFCB,$g$,$d$,$g'$)        // Fetch/update Q value
11:         $q' \leftarrow q+\alpha(r+\gamma$ argmax$g_{i\in G}$(Get(GFCB,$g$,$d$,$g_i$))$-q$)
12:         GFCB $\leftarrow$ Update(GFCB,$g$,$d$,$g'$,$q'$)
13:         if $r < 0$                               // Performing poorly?
14:             $d \leftarrow$ CalculateDiscrepancy($s'$, $x$)
15:             if Random(1) $\geq \varepsilon$          // Formulate next goal
16:                 $g'' \leftarrow$Argmax$g_{i\in G}$(Get(GFCB,$g$,$d$,$g_i$))
17:             else $g'' \leftarrow$Random(|$G$|)
18:             $g \leftarrow g'$ ; $g' \leftarrow g''$
19:             $\pi \leftarrow \Pi(g')$                  // Retrieve a new policy
20:         if Random(1) $\geq \varepsilon_2$
21:             $a' \leftarrow$ Argmax$a_{i\in A}$(Get($\Pi$, $\pi$, $s'$, $a_i$))
22:     $x \leftarrow$ Argmax $x_{i\in X}$(Get(ECB,$s'$,$a'$,$x_i$))
23:     Execute($a'$)                                // Execute current action
24:     $a \leftarrow a'$; $s \leftarrow s'$
// Phase 2: Goal extraction
25:   $G' \leftarrow$ TopFrequency($l_a$,$c_a$,$c_b$) ; $G \leftarrow \emptyset$
26:   for-each $g' \in G$                            // Iterate over the most frequent goals
27:     $H \leftarrow \emptyset$
28:     for-each ($g$, $\pi$) $\in \Pi$                  // Attempt to group $g'$ with an existing goal
29:         if Similarity($g$, $g'$) $\geq t$ then $H \leftarrow H \cup \{g\}$
30:     if $H = \emptyset$ then $H \leftarrow \{g'\}$      // $g'$ is a new goal
31:     $G \leftarrow G \cup H$
//Phase 3: Policies revision
32:  for-each $g \in G$
33:     if $\Pi \neq \emptyset$ then $\pi \leftarrow \Pi(g)$ else $\pi \leftarrow$ nil
34:     if $\pi$ = nil then $\pi \leftarrow$ New($g$); $\Pi \leftarrow \Pi \cup \{(g, \pi)\}$
35:     for-each <$s,a,s',r$> $\in l_b$
36:         $q \leftarrow$ Get($\Pi$,$\pi$,$s$,$a$)
37:         $q' \leftarrow q + \alpha(r + \gamma$argmax$a_{i\in A}$(Get($\Pi$,$\pi$,$s'$,$a_i$)) $- q$)
38:         $\pi \leftarrow$ Update($\pi$, $s$, $a$, $q'$)
39:     $\Pi \leftarrow$ Update($\Pi$, $g$, $\pi$)
40:  return $\Pi$, ECB, GFCB
```

In Phase 1 (Lines 1-24), GRL applies and updates ECB and GFCB. It first initializes s and x to the initial state s_0, action a to the null action, lists l_a and l_b to empty, g and current goal g' to the dummy goal g_0, discrepancy d to dummy value d_0, and G to the set of goals that can be accomplished by Π (Line 1) (i.e., policies are annotated with the goals they accomplish). During an episode (Lines 2-24), GRL periodically waits (Line 3) and then observes the current state s' (Line 4), calculates the reward r (line 5), and concatenates $<s'>$ to l_a and $<s,a,s',r>$ to l_b (Line 6) for use after the game episodes concludes. It then updates the distribution of expected states when taking action a in s (Line 7) and generates the current action a' randomly (Line 8). This guarantees that, if Π is empty, then GRL still has an action to perform. Otherwise (Line 9), it retrieves GFCB's estimated q value for formulating goal g' given (g,d) (Line 10), updates the new q' value using Q-learning (Line 11), and records it in the GFCB (Line 12). If the agent is performing poorly (Line 13), it then calculates the discrepancy between the current and expected states (Line 14) and retrieves a new goal g'' from GFCB using ε-greedy exploration (Lines 15-17), and updates its previous and current goal (Line 18). GRL then retrieves a new policy from Π using goal g' as the index (Lines 19-21). It retrieves from the ECB the expected state x from executing a', executes a', and updates the previous action a, current action a', and previous state s (Lines 22-24).

After an episode completes, GRL's Phase 2 extracts a set of goals to update their policies. It first identifies the set $\boldsymbol{G'}$ of most frequent states that appear in the most recent c_a% of visited states, where the frequency of these states must be at least a threshold value ($c_a \times c_b \times |l_a|$). For example, assume that $l_a = \{s_a,s_b,s_c,s_a,s_a, s_b,s_a,s_a,s_a,s_b\}$, $|l_a|=10$, $c_a = 50\%$, and $c_b = 0.25$. Then the most recent 50% of l_a is $\{s_b,s_a,s_a,s_a,s_b\}$, and state s_a is the most frequent state among these (with frequency 3). The threshold value equals 1.75 (i.e., 0.5×0.25×10), which means GRL will also include state s_b in G' because its frequency is 2. However, if $c_b = 0.1$, then $G'=\{s_a\}$ (Line 25). GRL then adds new goals from Π that are at least as similar to goals in G' as the threshold t (Line 29). Similarity between goals is computed using a linear combination of local similarity metrics, one for each of the state's features (Lopez de Mántaras *et al.*, 2005). More precisely, we assume cases to be vectors of n-dimensional features $X=\{x_1,...,x_n\}$. For computing similarity, we define a collection of local similarity metrics $\text{sim}_i()$, one per feature i, and a collection of weights α_i, which sum to 1. The aggregated similarity metric SIM_{agg} is defined as:

$$\text{SIM}_{\text{agg}}(X,Y) = \Sigma_{i=1,n}\ \alpha_i \cdot \text{sim}_i(x_i,y_i)$$

GRL groups goals by similarity to reduce the size of the Policies Case Base Π. However, if no similar goals exist, then GRL will interpret g' as a new goal (line 30).

In Phase 3, GRL refines or adds new policies. For each goal g in G (Line 32), if Π is not empty, then GRL will retrieve policy $\pi \in \Pi$ for this goal (Line 33). If either Π is empty or the policy associated with g is nil, a new policy π for g is created and π is added to Π (Line 34). It will then apply Q-learning to update π using the recent state transitions and rewards (Line 35-38) and update the (goal, revised policy) in Π (Line 39). Finally, GRL returns all the revised case bases (Line 40).

5 Example

Suppose in the real-time strategy (RTS) game Wargus a GRL-controlled agent is competing against one opponent (Wargus, 2012). Wargus is a combat game where each player controls a variety of units. One of the most challenging aspects in RTS games is that there is no "best" unit type. For example, archers can quickly kill footmen but are particularly vulnerable to knights. In our experiments each player controlled mages, archers, knights, ballistae, and footmen. The objective of this game is to be the first to reach a predefined number of points, which are earned by killing the opponent's units. Some units award more points than others (e.g., killing a knight earns more points than a footman).

Assume each team begins with two footmen and two archers, and that the agent has already played many games. Thus, the case bases Π, ECB, and GFCB have recorded some results. For the Wargus state representation we use $s' = (u_1,u_2,\ldots,u_n, e_1,e_2,\ldots,e_m)$, where $u_i \in \mathbb{Z}$ and $e_j \in \mathbb{Z}$ denote the number of remaining units of type i on our team and e_j denotes the same of type j for the opponent. Usually, n and m are equal (e.g., if the current state equals (2,1,0,2), then our team has 2 footmen and one archer remaining while the opponent has only two archers). Actions in Wargus, denoted as $a' = (b_1,b_2,\ldots,b_k)$, where each b_i is a unit type of the opponent such as {R=archers, F=footmen}, means that units of type i on GRL's team attacks opponent units of type b_i. For example, the action (R,F,F,∅) means that a unit with id 1 attacks an opponent archer, units with id 2 and 3 attack opponent footmen, and units with id 4 do nothing.

Suppose the current state s' is $s_{21} = (1,0,0,1)$ (i.e., GRL's team has only one footman left and the opponent has only one archer) and $r = -2$ (Lines 4-5). In Phase 1, GRL adds $\{s_{21}\}$ to l_a and $\{(s_{20}, a_{20}, s_{21}, -2)\}$ to l_b (Line 6). After updating the appropriate ECB distribution (Line 7), GRL will generate the random action a' and then calculate and update the q value of GFCB (Lines 10-12). Because the reward is negative, GRL will change to a new goal (Line 13). After finding the discrepancy d_{21} = (T,T,T,F) between current state s_{21} and expectation $x_{21} = (1,0,0,0)$, it will choose a new goal g'' in an ε-greedy fashion (Lines 14-18). Using this new goal to retrieve a policy $\pi \in \Pi$, suppose it retrieves (by chance) greedy action $a'_{21} = (\varnothing,R,\varnothing,\varnothing)$ (i.e., send the remaining footman to attack an enemy archer) from policy π (Lines 19-21). GRL then updates the previous state and action, computes expectation x, and executes action a' (Lines 22-24). Suppose that this action eliminates the opponent's units, which ends the game.

Phases 2-3 update Π. First, GRL uses TopFrequency to compute a set of new goals G', and then searches for goals from Π that are similar to any members in G' to create a set G (Lines 25-31). Suppose G'={(100,150,0,0)} (e.g., we have 100 footmen and 150 archers while the enemy has 0 footmen and 0 archers), meaning that GRL won the episode because it destroyed all enemy units. Assume $\Pi = \{((96,96,0,0),\pi_1), ((0,0,10,20),\pi_2), ((150,100,0,0),\pi_3), ((105,104,0,0),\pi_4)\}$. Then $G=G'$ assuming there are no (sufficiently) similar cases in Π. Lines 39-48 will learn a policy π_5, and $\{((100,150,0,0),\pi_5)\}$ will be added to Π. On the other hand, if G'={(100,100,0,0)} and assuming it would be (above-threshold) similar to the first and fourth goals in Π, then G={(96,96,0,0), (105,104,0,0)} and Lines 39-48 will update the policies π_1 and π_4.

6 Empirical Study

We examined the task of winning two adversarial games to investigate the following hypothesis: GRL can significantly outperform a standard RL agent that learns ***only*** policies (i.e., Retaliate (Smith *et al.*, 2007), which uses Q-learning) and an ablated GDA agent that does ***not*** learn policies (i.e., LGDA (Jaidee *et al.*, 2011a), which is given policies representing an opponent's strategies and their goals, and learns only expectations and goal formulation knowledge). In our study, all three learning agents use the same models for states, actions, and rewards.

6.1 Domains and Scenarios

The adversarial games we use are Wargus and DOM. Both are two-player real-time video games: players make asynchronous moves. They exhibit the characteristics that we want to explore in this paper: there doesn't seem to be a universally good strategy for these games. Instead, they exhibit the "rock-paper-scissors" behavior whereby any strategy can be countered. LGDA have demonstrated good performance in DOM and Wargus (Jaidee *et al.*, 2011a; 2011b) while Retaliate has demonstrated good performance in DOM (Smith *et al.*, 2007), so they are good baselines for testing GRL.

We used two maps in our Wargus experiments. The first is a medium-sized map with 64×64 cells and 8 units per player, while the second uses the largest feasible map (128×128 cells) and 32 units per player. We set the games' score limits to be 200 and 1000 points, respectively. In our experiments, we used five hand-coded opponents that order all units of the same type to attack a single type of the agent's units. For example, they might assign knights to attack archers. These opponents differ in their attack order. In testing, no single opponent outperformed all the others. We used these built-in opponents to train the three agents (i.e., Retaliate, LGDA and GRL).

The second domain, DOM, is a domination game in which two opponent players try to capture specified domination locations on a 2-D map. Teams are composed of k bots. The player's actions are k-tuples $(l_1,..l_k)$ indicating the domination location l_i to which each bot b_i is assigned. A player captures a location by simply moving a bot to it. A team receives one point for every five consecutive ticks it "owns" a location. The first team to earn a predefined number of points wins. Each bot starts with a max number of health points, which can be lost in combat, which occurs when two or more opposing bots are within a certain range of each other. When a bot's health is zero, it respawns after a few ticks in a (randomly-selected) respawning location with max health points. Combat losses are determined using a biased random function that computes the health points lost by each competing bot (it favors bots on the team that has more bots within a certain range). In our experiment, we use a map with five domination locations and eight bots per team.

We used the same six hand-coded opponents in DOM we previously used in (Jaidee *et al.*, 2011a), where we used a variety of fixed strategies such as the "half plus one adversary", which attempts to control a majority of locations by sending bots to them whenever they are owned by the competing agent. Another strategy, called "smart opportunistic", sends a different bot to each domination location the team does not own. Among these six adversaries, there are two that are better than all the others, two that are middling performers, and the last two are defeated by all the others.

6.2 Protocol and Results

Agents played *N* episodes, where *N*=20 for Wargus and *N*=340 and 2000 for DOM. The difference in the number N of runs between Wargus and Dom is due to the fact that running DOM games is much quicker. During each training episode, each agent played each of the *M* built-in opponents once (*M*=5 for Wargus and *M*=6 for DOM). During training, the agents GRL, LGDA and Retaliate are learning. We tested GRL against Retaliate and LGDA after each training episode. Because both DOM and Wargus are highly stochastic, games during testing were repeated 10 times. Any knowledge learned during a game in the testing phase was removed after the game ends. Thus, the only knowledge affecting the performance of the agents when competing versus one another was learned during training and any knowledge learned online within that particular game episode.

Figures 2 and 3 summarize the average results. The x-axis plots the number of training episodes, while the y-axis plots the average utility (i.e., score difference of GRL versus another agent).

Experiment 1 (Wargus): In most Wargus episodes (Figure 2), GRL clearly outperformed the other agents, although LGDA sometimes defeated GRL in the medium-size map (Figure 2b). Nevertheless in all cases the differences are statistically significant (p<0.001), as determined by a two-tailed Student's t-Test on the utility scores of GRL versus the scores of another agent (i.e., Retaliate or LGDA). Hence our hypothesis is supported for Wargus, and we can draw three conclusions:

1. There is either no universally good strategy for these games or none can be found by Q-learning even after a large number of episodes.
2. GRL outperformed the Q-learning agent. This highlights the importance for using case-based approaches to learn and reason about expectations, goal formulation knowledge, and goal-specific policies in domains where no universally-best strategy can be elicited by RL.
3. GRL outperformed the LGDA agent. This highlights the importance of identifying new goals and using CBR to learn and reuse goal-specific cases.

We were surprised that GRL outperformed LGDA after only a few episodes because GRL begins with no goals and no policies. In contrast, LGDA begins with policies representing the built-in opponents' strategies and goals for these policies. Upon inspection we found that the opponents' strategies cause their units to form choke points while trying to reach the units they intended to attack. As a result, few units, mostly ranged attack units, actually were effective. Without knowledge about expectations and goals, LGDA rotates among the various opponents' strategies. As mentioned, these end up being ineffectual because it frequently results in choke points. GRL instead initially performs random actions that, on average, cause more of their own units to damage opponent units, which explains the relative results of the first few episodes.

We also investigated why, despite its overall good performance, GRL will occasionally lose games to the opponents in the medium-sized map (e.g., in round 13 versus Retaliate (Figure 2a) and round 20 versus LGDA (Figure 2b)). We found that for this map the score limit was frequently reached even though both teams had several units left. That is, the maximum point threshold was set too low for the number and types of units in the scenario (i.e., killing a high-value unit such as a knight is worth many points, and the game ends sooner when any such unit is killed). This caused high variation in the results because, after a while, several units from both sides will have few health points. In this situation, after a few of these units die the game terminates because the point limit is reached. As a result, depending on the random factor that determines which unit attacks succeeded, units from either side die while others remain with few health points. However, points are only awarded for deaths, and not for low health points. This caused the variance in the results. This was not a factor in the large map because the number of points was set sufficiently high and, although there is fluctuation, GRL did not lose a game on average (Figures 2c and 2d).

Experiment 2 (DOM): Figure 3 summarizes the results with DOM games. In all cases GRL clearly outperformed the other agents, although initially both Retaliate and LGDA outperformed GRL. This is to be expected; GRL initially has no knowledge of which goals to pursue nor how to achieve them. Nevertheless in all cases the difference is statistically significant (p<0.001) across the entire curves, as determined by a two-tailed Student's t-Test for comparing the utility scores of GRL versus those of the other two learning agents). This also supports our hypothesis and allows us draw the same conclusions as mentioned above for Experiment 1.

We investigated why it took so many episodes for GRL to start winning versus Retaliate and LGDA in the DOM game compared to Wargus. This occurred because the state model used by the agents forms a DAG for Wargus, meaning that a state is never visited more than once. As a result, for Wargus, we define the new goal to be the final state (whereas for DOM this is defined as the most frequently visited state). In contrast, the same state can be visited multiple times in DOM. Thus, multiple goals were frequently learned per DOM episode, resulting in many more goals being learned overall. Hence, Π grows faster in the DOM rather than in the Wargus experiments during the initial training episodes. This in turn increases the number of episodes needed to learn useful goal formulation knowledge and good policies. Thus, it takes longer for GRL to outperform the other agents in DOM scenarios.

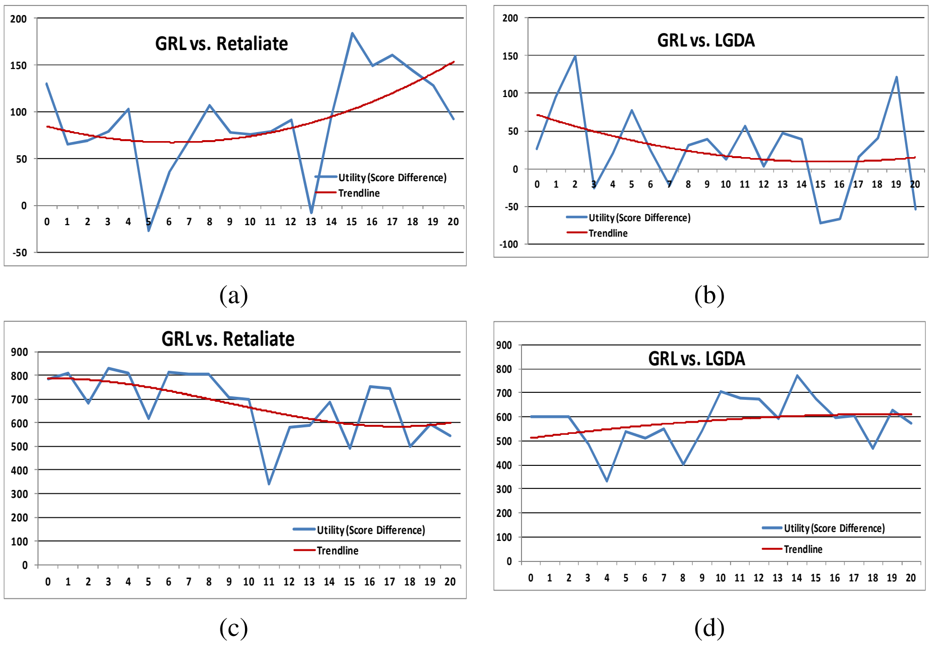


Fig. 2. The results of the Wargus experiments: GRL vs. Retaliate (a) and vs. LGDA (b) on the medium map, and GRL vs. Retaliate (c) and vs. LGDA (d) on the large map. The x-axis plots the number of training episodes, while the y-axis plots the average utility (i.e., score difference of GRL versus another agent).

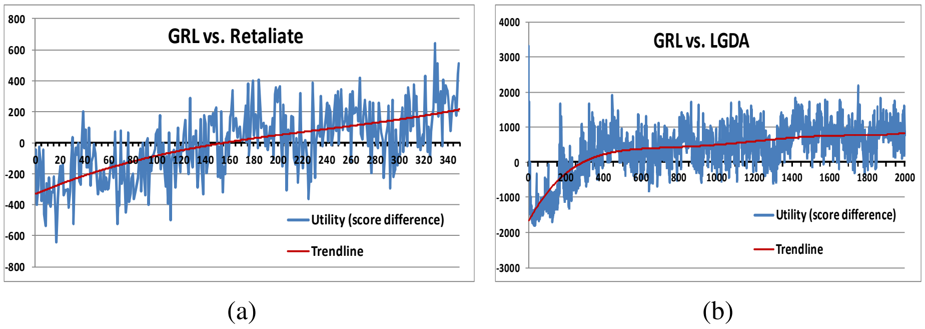


Fig. 3. Results from the DOM experiments: (a) GRL vs. Retaliate and (b) GRL vs. LGDA. The x-axis plots the number of training episodes, while the y-axis plots the average utility (i.e., score difference of GRL versus another agent).

We also investigated why it took so many more episodes for GRL to outperform LGDA compared to Retaliate. Namely, it took around 150 episodes for Retaliate compared to almost 300 for LGDA. This was caused by the two strong hand-coded adversaries, which LGDA was able to leverage. This also explains why, in the first episode, GRL loses to Retaliate by approximately 350 points whereas it loses to LGDA by approximately 1600 points.

7 Conclusions and Future Work

We introduced a goal-driven autonomy (GDA) agent, named the Goal Reasoning Learner (GRL), which uses CBR processes to learn and reuse goal-specific action policies, state expectations, and goal selection knowledge. It is the first GDA agent that learns all of this information, and in particular the first to learn policies. GRL uses a reinforcement learning (RL) process to learn its policies and goal formulation knowledge. GDA agents are designed for complex environments in which unexpected situations can occur, as is the case for complex video game environments. In our empirical study with two such environments (Wargus and DOM), we found that GRL outperforms its RL-only ablation, even though GRL is embedded in the same RL process: it uses the same reward function, has knowledge of the same actions, and uses the same state-action transition model.

Our GDA agent does not perform discrepancy explanation (Molineaux *et al.,* 2010). GDA agents often generate an explanation for a discrepancy and formulate a new goal based on it. GRL bypasses this step by directly linking discrepancies with goals in the GFCB. Cox (2007) proposes a general taxonomy for plausible explanations of a discrepancy. This taxonomy classifies potential categories of explanations (called *meta-explanations*) including: incomplete domain knowledge, incorrect domain knowledge, and noise in the environment. Most research on explanations has been in the context of deterministic expectations, whereas GRL would require modeling stochastic explanations. We plan to study these issues in the future. As discussed in Section 6, learning too many goals caused GRL to require many episodes before it became competitive in the DOM domain. Hence, we also plan to study alternative criteria to learn goals.

Acknowledgements. This work was sponsored by NRL and NSF (#0642882). The views, opinions, and findings contained in this paper are those of the authors and should not be interpreted as representing the official views or policies, either expressed or implied, of NRL or the DoD.

References

Aha, D.W., Klenk, M., Muñoz-Avila, H., Ram, A., Shapiro, D. (eds.): Goal-Directed Autonomy: Notes from the AAAI Workshop (W4), Atlanta, GA (2010), `http://www.cse.lehigh.edu/~munoz/gda/`

Auslander, B., Lee-Urban, S., Hogg, C., Muñoz-Avila, H.: Recognizing the Enemy: Combining Reinforcement Learning with Strategy Selection Using Case-Based Reasoning. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 59–73. Springer, Heidelberg (2008)

Bianchi, R.A.C., Ros, R., Lopez de Mantaras, R.: Improving Reinforcement Learning by Using Case Based Heuristics. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 75–89. Springer, Heidelberg (2009)

Bridge, D.G.: The Virtue of Reward: Performance, Reinforcement and Discovery in Case-Based Reasoning. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS (LNAI), vol. 3620, pp. 1–1. Springer, Heidelberg (2005)

Bouguerra, A., Karlsson, L., Saffiotti, A.: Monitoring the execution of robot plans using semantic knowledge. Robotics and Autonomous Systems 56(11), 942–954 (2008)

Cox, M.T.: Perpetual self-aware cognitive agents. AI Magazine 28(1), 23–45 (2007)

Dilts, M., Muñoz-Avila, H.: Reducing the Memory Footprint of Temporal Difference Learning over Finitely Many States by Using Case-Based Generalization. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010. LNCS, vol. 6176, pp. 81–95. Springer, Heidelberg (2010)

Gabel, T., Riedmiller, M.: CBR for State Value Function Approximation in Reinforcement Learning. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS (LNAI), vol. 3620, pp. 206–221. Springer, Heidelberg (2005)

Gabel, T., Riedmiller, M.: An Analysis of Case-Based Value Function Approximation by Approximating State Transition Graphs. In: Weber, R.O., Richter, M.M. (eds.) ICCBR 2007. LNCS (LNAI), vol. 4626, pp. 344–358. Springer, Heidelberg (2007)

Ghallab, M., Nau, D.S., Traverso, P.: Automated planning: Theory and practice. Morgan Kaufmann, San Mateo (2004)

Jaidee, U., Muñoz-Avila, H., Aha, D.W.: Integrated learning for goal-driven autonomy. In: Proceedings of the Twenty-Second International Conference on Artificial Intelligence. AAAI Press, Barcelona (2011a)

Jaidee, U., Muñoz-Avila, H., Aha, D.W.: Case-based learning in goal-driven autonomy agents for real-time strategy combat tasks. In: Floyd, M.W., Sánchez-Ruiz, A.A. (eds.) Case-Based Reasoning in Computer Games: Papers from the ICCBR Workshop. U. Greenwich, London (2011b)

Lopez de Mántaras, R., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M.L., Cox, M.T., Forbus, K., Keane, M., Aamodt, A., Watson, I.: Retrieval, reuse and retention in case-based reasoning. Knowledge Engineering Review 20(3), 215–240 (2005)

Molineaux, M., Klenk, M., Aha, D.W.: Goal-driven autonomy in a Navy strategy simulation. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence. AAAI Press, Atlanta (2010)

Muñoz-Avila, H., Jaidee, U., Aha, D.W., Carter, E.: Goal-Driven Autonomy with Case-Based Reasoning. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010. LNCS, vol. 6176, pp. 228–241. Springer, Heidelberg (2010)

Nau, D.S.: Current trends in automated planning. AI Magazine 28(4), 43–58 (2007)

Powell, J., Molineaux, M., Aha, D.W.: Active and interactive discovery of goal selection knowledge. In: Proceedings of the Twenty-Fourth Conference of the Florida AI Research Society. AAAI Press, West Palm Beach (2011)

Ram, A., Santamaria, J.C.: Continuous case-based reasoning. Artificial Intelligence 90(1-2), 25–77 (1997)

Smith, M., Lee-Urban, S., Muñoz-Avila, H.: RETALIATE: Learning winning policies in first-person shooter games. In: Proceedings of the Nineteenth Innovative Applications of AI Conference, pp. 1801–1806. AAAI Press, Vancouver (2007)

Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction. MIT Press, Cambridge (1998)

Wargus. Source code for Wargus (2012), `http://wargus.sourceforge.net/` (last checked: January 2012)

Weber, B., Mateas, M., Jhala, A.: Applying goal-driven autonomy to StarCraft. In: Proceedings of the Sixth Conference on Artificial Intelligence and Interactive Digital Entertainment. AAAI Press, Stanford (2010a)

Weber, B., Mateas, M., Jhala, A.: Case-based goal formulation. In: Aha, et al. (eds.) (2010b)

Custom Accessibility-Based CCBR Question Selection by Ongoing User Classification

Vahid Jalali and David Leake

School of Informatics and Computing, Indiana University
Bloomington IN 47408, USA
{vjalalib,leake}@cs.indiana.edu

Abstract. Question selection in Conversational Case-Based Reasoning (CCBR) is traditionally guided by the discriminativeness of questions, to minimize dialog length for retrieval. However, users may not always be able or willing to answer the most discriminative questions. This paper presents Accessibility Influenced Attribute Selection Plus (AIAS+), a method for customizing CCBR question selection to reflect the types of questions the user is likely to answer. Given background knowledge about response probabilities for different questions by different user groups, AIAS+ performs ongoing classification of new users, based on the questions they choose to answer, uses the classifications to predict the likelihood of the user answering particular questions, and applies those predictions to guide question selection. In addition, its question selection process balances questions' information gain against their potential to aid user classification, to enable better selection of following questions. Experiments with simulated users show improvement over three alternative methods. Experiments in synthetic domains illuminate the domain characteristics under which the method is expected to be effective.

1 Introduction

Conversational Case-Based Reasoning (CCBR) is an interactive form of Case-Based Reasoning (CBR) in which the CBR system guides users through an interactive dialog to select cases to retrieve [1]. In order to minimize the number of questions a user must answer, CCBR systems often rank questions based on estimates of how useful the answers will be in rapidly identifying the target case. Such approaches are effective when users generally can answer the presented questions.

However, in some domains, the questions whose answers are most useful for discriminating cases may not elicit answers. Users may lack the requisite domain knowledge to answer some questions; for example, in a product recommendation domain, a user may not be knowledgeable enough to specify some technical features. Likewise, determining a feature value may be prohibitively expensive or risky; for example, in medical diagnosis domain, it is preferable to avoid providing feature values which can only be determined by invasive tests. In addition, the user may consider some features unimportant and prefer that they be ignored in case selection. For example, in product recommendation, a user unconcerned about the appearance of a cell phone might choose not to commit to a color choice, even if color happens to discriminate between cases in the system, in contrast to a user views a cell phone a fashion accessory and considers color to be a crucial feature. Thus it is desirable for a CCBR system to customize its question selection to the characteristics of the current user.

In previous work, we presented a method for accessibility-influenced question selection, AIAS [2], and demonstrated that when the user's ability to answer particular questions varies, and when response probabilities can be estimated accurately, AIAS can decrease CCBR dialog length. Unfortunately, information on a particular user's response probabilities for different features may not be available. For example, in e-commerce contexts, response probabilities for new users may not be available, and even for known users, response probabilities may not be available if the CCBR system is recommending a type of product the user has never before purchased. However, in such situations, if the user can be identified as a member of a relevant group whose characteristics are known, the group profile may be used to provide feature response probabilities for particular information.

This paper proposes and tests AIAS+, a CCBR question selection method which customizes question selection based on ongoing user classification. AIAS+ is provided with profiles of different user groups, with each profile collecting the probabilities of members of its group responding to a question about each case feature. AIAS+ applies a Naive Bayesian classifier to incrementally classify unknown users by group, throughout a dialog. AIAS+ chooses questions based on three factors: The value of the question for discriminating cases, the predicted accessibility of the response to the question, given the current user classification (if any), and the value of the question's response for increasing certainty in the classification of the user, in order to improve future predictions. We test AIAS+ in a number of domains, compared to a baseline non-customized method, AIAS, and an accessibility-based method from the literature which treats all users uniformly, demonstrating its benefits when users have different characteristics. Because the benefits depend on the test domain, we also present results on synthetic domains, studying the circumstances under which AIAS+ is most effective.

2 Related Work

CCBR has been extensively applied to tasks such as help desks [3] and E-Commerce applications [4], and remains an active research area [5]. CCBR question selection methods range from knowledge-based [6,7,8] to statistical measures. Information gain criteria from decision tree learning [9] have been widely used, with the aim of selecting questions to minimize dialog length [10,11,12,13,14]. In some domains there maybe a tradeoff between dialog efficiency and solution quality [15], and costs can be associated with answering questions, to determine whether it is worthwhile to ask additional questions [16].

Carrick et al. [17] address the issue of users' ability to answer questions with a question selection method considering both information quality and estimated cost for answering a question. Kohlmaier et al. [18] propose assigning costs to questions based on a coarse-grain assessment of the amount of effort needed to answer them and the user's interest in the provided questions, distinguishing the categories *answer without help*, *answer with help*, *have no preference*, and *fail to understand*. Their method alters the information gain of questions by assigning a penalty factor for questions whose answers require additional effort. Goker and Thompson [19] propose a method for selecting the next question to ask based on its information gain, while maintaining a model of user preferences (e.g., for specific items, relative attribute importances, and diversity of the suggested items and values). These preferences are used to narrow the set of retrieved cases. Ricci et al. [20] propose a method for reducing the number of questions by using both content features (e.g., hotel rating) and collaborative features (e.g., user nationality). They propose a two step retrieval process with initial retrieval based on the content features provided in the user's query, and the ranking of retrieved cases updated based on collaborative features. Most relevant to this paper is Mirzadeh et al.'s [21] use of entropy and response likelihood for question selection. Their approach captures response likelihood as the popularity of a given feature, calculated as the ratio of user queries containing that feature to the total number of queries. However, they do not distinguish between different user groups or individuals for calculating the popularity of a feature. How to apply such information is the central focus of this paper.

3 User Response Profiles

Our approach depends on simple accessibility profiles, describing accessibilities for particular classes of users and user contexts. This section sketches their main characteristics.

Information contained in profiles: We assume the system is provided with a set of user profiles, each one containing information about the probability of users in the profiled group responding to a CCBR question about each possible case feature. Each user group is associated with a function $G : F \to [0, 1]$, where F is the set of domain features. Given $f \in F$, $G(f)$ is the probability of a member of the group providing an answer to a question about feature f. Note that profiles can capture information about any number of users; a profile describing only one user would result in personalized question selection.

Context-dependence of group membership: For the purposes of a single dialog, we treat each user as belonging to exactly one group. However, even in a single domain, users' responses behavior might differ depending on their tasks. For example, a given user of a hotel recommendation system might sometimes seek hotels for business trips, and at other times, hotels for family vacations, corresponding to interest in different types of features (wifi in the first case, and a playground in the second). Thus our approach enables different profiles to be selected for different dialogs.

Profile generation: If profile information is not available from external sources, it can be mined from records of CCBR dialogs. For example, in a product recommender system for cell phones, past user behavior could be clustered to determine groups of similar shoppers (e.g., those who are familiar with certain technical features, and those who are technically unsophisticated) and the questions they answer, for future use.

4 AIAS+'s Basic Question Selection Approach

AIAS+ ranks questions by balancing three factors: The discriminativeness of the question, the response likelihood for the question—the likelihood that the user will provide an answer to the question—and the expected value of the answer for refining the classification of the user (which in turn can lead to more accurate predictions of response likelihood for future questions). As discriminativeness has been extensively studied in CCBR (see Section 2), the remainder of this paper focuses on response likelihood and user classification, and their integration into the question selection process.

4.1 Calculating Group Membership Probabilities during a Dialog

CCBR systems generally provide multiple questions at each step in a dialog. The probability of a user belonging to a given group can be updated incrementally during a CCBR dialog, based on the questions the user chooses to answer. Let U be the set of all user groups, $O(f)$ be a Boolean value denoting whether a new question about feature f was answered, and let $P(u|O(f))$ represent the probability of the user belonging to group $u \in U$ given $O(f)$. The probability of the user belonging to u can be updated by the following Bayesian formula:

$$\forall u \in U, P(u|O(f)) = \frac{P(O(f)|u) \times P(u)}{P(O(f))} \tag{1}$$

$P(O(f)|u)$ is provided by the group profile, and $P(O(f))$ can be calculated as:

$$P(O(f)) = \sum_{u \in U} P(O(f)|u) \times P(u) \tag{2}$$

4.2 AIAS

Accessibility Influenced Attribute Selection (AIAS) [2] selects questions to ask based both on their features' information gain and the user response likelihood. We denote the information gain from knowing feature f's value as Gain(f), and its response probability as Accessibility(f). Given a current user group assignment (taken to be the most likely user group, as calculated in the previous section), AIAS calculates the value of a question as:

$$AIAS\_question\_value(f) = Gain(f) \times Accessibility(f) \tag{3}$$

We tested AIAS on four sample domains, for selection of cell phones, restaurants, automobiles, and universities, with (respectively) 2, 2, 3, and 4 different user groups for each of those domains, with differing abilities to provide information about different features. In these experiments, AIAS improved average dialog lengths by 29%, 12%, 15% and 36% over the baseline [2].

4.3 A Limitation of AIAS

AIAS depends on knowing the user's group. However, because AIAS selects questions only based on their discriminativeness and accessibility—not on their ability to help classify the user—classification of the user may be delayed, decreasing the benefit of AIAS. To illustrate the problem, consider 3 features (e.g. $f1$, $f2$ and $f3$) of a domain with two user groups $u1$ and $u2$. Assume $f1$, $f2$ and $f3$ have identical information gain, but differing response likelihoods. For users from group u1, the respective response likelihoods are 0.51, 0.0, and 1.0; for users from u2, the respective likelihoods are 0.51, 1.0, and 0.0.

If the system begins the dialog with no prior knowledge about the interacting user's group, the user has a 50% chance of belonging to u1 or u2. In this case, AIAS will ask about f1 first, because it has the highest response likelihood (0.51). However, starting with either of the other features would enable identifying the user's group with 100% certainty, which might enable much better subsequent question selection. Thus information-gathering about the user may be an important factor in question selection.

4.4 Estimating Confidence in Group Memberships

Making good choices about when to seek information about a user depends on assessing the need for that information. If a CCBR system is highly confident of its group membership assessments, it should simply apply them; if it is highly uncertain, more information is needed.

As a heuristic for judging confidence in group membership, AIAS+ uses the variance of the probabilities of membership in the different groups. Given n user groups to distinguish, in the worst case (with no information to favor any group) there is an equal ($\frac{1}{n}$) chance that the user belongs to any one of those n groups. Certainty is maximized when the probability of the interacting user belonging to $n-1$ of the groups is 0 and it is only 1 for a single group. In general, if P(u) denotes the probability assigned to the user belonging to group u, and U denotes the set of all user groups, the variance of user group assumptions can be calculated as:

$$Var(U) = \frac{\sum_{u \in U}(P(u) - \frac{1}{|U|})^2}{|U|} \tag{4}$$

To reflect when knowledge of a feature's value increases confidence in a user's group membership, AIAS+ uses the expected gain in variance from asking for a given feature value, taking into account the probability that the user will answer the question. Let $P(answer(f))$ denote the probability of the user answering a question about feature f, given the system's current classification of the user's group membership. Let U be the set of probabilities denoting group membership assumptions after the user has answered or skipped the current question, U_0 be the set of probabilities prior to processing the question, and $Var(U|answer(f))$ and $Var(U|\neg answer(f))$ denote the variance of updated user group assumptions after feature f is answered or skipped by the user. Then the expected variance gain can be calculated as follows:

$$\begin{aligned} E(VarGain(f)) \equiv\ & (P(answer(f)) \times Var(U|answer(f)) + \\ & (1 - P(answer(f)) \times Var(U|\neg answer(f))) \\ & - Var(U_0) \end{aligned} \tag{5}$$

4.5 AIAS+

The AIAS+ approach selects questions based on information gain, accessibility, and expected variance gain in the set of user group probabilities. Let $Pos(x) = \max(x, 0)$. Then the AIAS+ formula for assigning weights to possible questions to ask is:

$$QuestionScore(f) = Gain(f) \times (Accessibility(f) + Pos(E(VarGain(f)))) \tag{6}$$

Questions are ranked by *QuestionScore*, with the most highly scored question asked first. Note that in the QuestionScore formula, expected variance gains are only considered if they are positive, in order not to penalize features which help the system to correct a misclassified user. Negative expected variance gains for a feature occur when the system is relatively confident about the class to which the interacting user belongs and asking a question about that feature is expected to reduce the system's confidence compared to its current state.

5 Evaluation Protocol

We evaluated AIAS+ by comparing four question selection methods:

1. Entropy-based: Choosing the questions only by their information gain
2. AIAS: Choosing questions based on information gain and accessibility
3. Popularity: Choosing questions based on Mirzadeh et al.'s popularity-based method, which considers accessibility for uniform user groups.
4. AIAS+: Choosing questions based on information gain, accessibility and expected variance gain for differing user groups.

Our evaluation addressed the following questions:

1. How does the dialog efficiency with AIAS+ question selection compare to efficiency with a purely entropy-based approach, AIAS, and Popularity?

2. How are the relative efficiencies affected by the number of questions presented to the user in each step of a dialog?
3. How sensitive is the performance of AIAS to tuning of weightings for information gain, accessibility, and variance gain?

5.1 Data Used

We evaluated AIAS both on standard domains and on domains generated to test the effects of specific domain characteristics. The standard domains used were the Automobile, Flag, and Housing domains [22]. The Automobile domain has 205 instances each with 26 attributes. Flag has 194 instances each with 29 attributes. Housing has 506 instances with 14 features each.

We generated five synthetic domains, D1, D2, D3, D4, and D5, each containing 1000 cases with 20 features, with 5 unique values for each feature, whose values were distributed randomly across different cases based on a uniform distribution for the number of instances of each unique value. Fig. 1 depicts the information gain of the features in the sample domains. The ordering of the features is only significant as it relates to the following accessibility profiles, as described later.

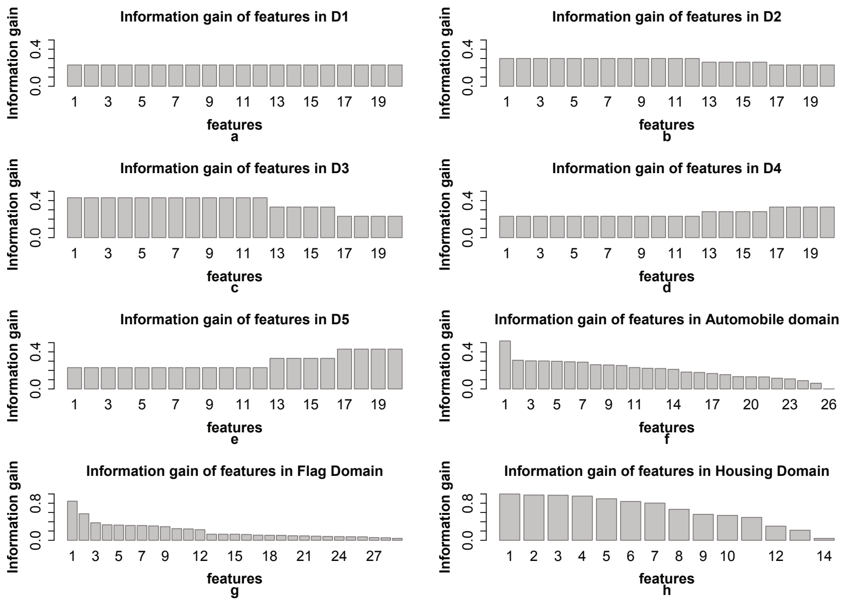


Fig. 1. The information gain of features in the sample synthetic domains

We considered 6 different accessibility distributions, each associated with two different groups depicted in parts a through f of Fig. 2. The distributions are selected to test a range of scenarios. Lines in the graph are for visibility only; all

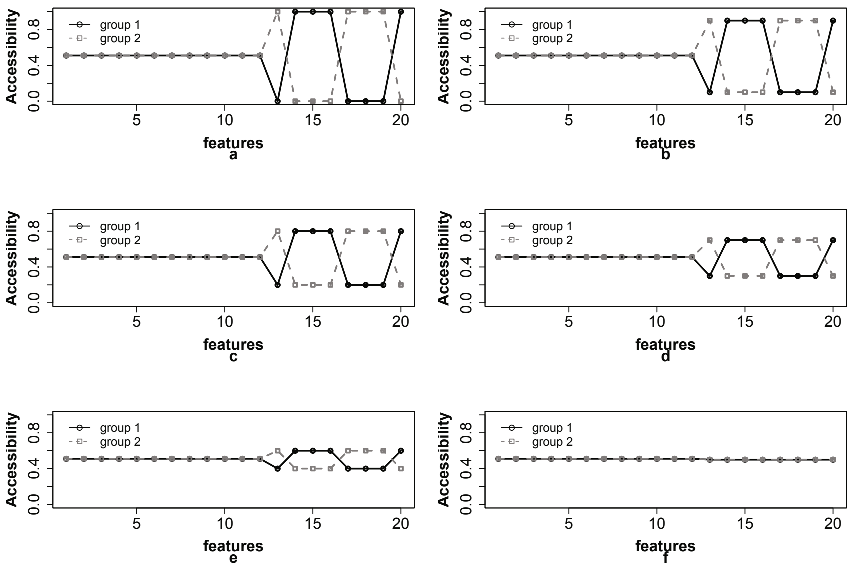


Fig. 2. Candidate accessibility distributions of features in the synthetic domains

points in the profiles are discrete values. For each test, the combination of an information gain profile with an accessibility profile determines the characteristics of the domain during testing.

5.2 Simulating CCBR Dialogs

We used leave-one-in testing [6] to test retrieval performance. To guide retrievals, we implemented a simulated CCBR user which determined stochastically which candidate questions to answer, based on accessibility information provided as a set of probabilities of answering questions about a particular feature.

CCBR systems typically present users with a set of the highest-ranked questions, rather than only a single top question. For tests in which multiple questions are presented to the simulated user in a single step, the simulated user starts with the highest ranked question, determines at random (with probability equal to the feature accessibility) whether it can answer the question, and continues checking questions in rank order until reaching one it can answer or exhausting the presented questions.

For each user group, each case in the case base is used once as a target, but retained in the case base. This process is repeated 100 times for each user group. The average number of questions considered by the simulated user for finding a target case is used as an efficiency indicator for each method. In addition, the average number of skipped questions by the user is considered when studying the efficiency effects of having different number of questions per step of a dialog.

6 Experimental Results

6.1 Question 1: Dialog Length Using Entropy-Based, AIAS, Popularity, and AIAS+ Question Selection

In our first experiment, we compared the performance of the four strategies using the synthetic data sets described in parts a through e of Fig. 1 and accessibility distributions in Fig. 2. We used the average number of questions considered by the simulated user for finding the target case as the performance indicator, with only one question presented to the simulated user at each step.

Fig. 3 shows the percentage of performance difference of AIAS, Popularity and AIAS+ compared the Entropy-based method (considered as baseline) in the five sample synthetic domains for six feature accessibility distributions. In nearly all our experiments, AIAS+ performed at least as well as AIAS (the exceptions are three cases in which AIAS outperformed it by 1%). In almost all cases for which feature accessibilities varied, both AIAS and AIAS+ outperformed the entropy-based approach (the exceptions are four cases in which the entropy-based approach outperformed AIAS by at most 2%). They also performed at least as well as the popularity-based approach, and usually outperformed it. The results suggest the following observations:

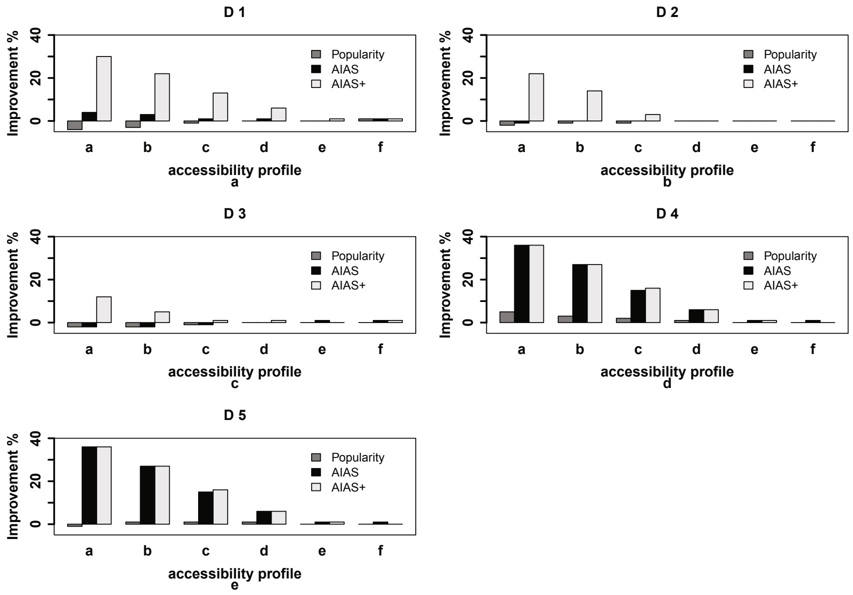


Fig. 3. Percentage of improvement of AIAS+, AIAS and Popularity methods over the Entropy-based method in terms of the average number of considered questions in 5 sample domains for 6 candidate accessibility distributions

– As shown in parts a to c of Fig. 3, AIAS+ outperforms AIAS when there is a competition between two sets of features, one with identical or higher information gains and relatively greater overall response likelihood compared to the features in the second set. But features in the second set have greater expected variance, meaning that they can help the system classify the user's group in fewer steps, and can be worthwhile to pursue. For domains D1, D2 and D3 and the tested accessibility distributions, this feature split holds from the beginning of a dialog, but in practice this situation could occur later as well.
– The less the accessibility of different features varies, the closer the performance of AIAS and AIAS+. This is especially noticeable for accessibility distribution profiles d,e and f in all tested domains.
– Delays in accurate classification of user group result in AIAS performing similarly to the Entropy-based method. However, as shown in parts d and e, AIAS outperforms the Entropy-based method when there is no competition between the information gain and expected variance gain measures, in which case, AIAS+ and AIAS perform identically.
– In all the test domains, if for each user group, many questions are difficult to answer, all three methods will have similar performance, as shown by the last bar (f) for all 5 domains. When the user cannot answer enough questions to identify a unique target case and the dialog terminates after exhausting the candidate questions, all methods have the same average dialog length.
– We note that in domains D1, D2, D3 and to some extent D5, the Popularity method's performance is lower than or equal to that of baseline. This is because Popularity is designed for situations in which all users have the same ability to answer questions. This approach works well for homogeneous user groups, but when there is a dramatic difference between feature accessibilities for the different user groups (e.g. as in accessibility profiles a, b and to some extent c), Popularity yields longer dialogs. This can be seen in parts a, b and c of Fig. 3 for accessibility profiles a, b and c.

6.2 Question 2: The Effect of the Number of Questions Presented Per Step of a Dialog on the Performance of AIAS+

CCBR systems present a ranked set of questions at each step of the dialog, for users to choose one to answer. To study how the number of questions provided at each step of the dialog affects the relative performance of AIAS+, we conducted experiments comparing AIAS+ and the Entropy-based method in the Automobile domain. Two candidate accessibility distributions for two user groups are shown in parts a and b of Fig. 4 and the information gain of features in the Automobile domain is shown at part f of Fig. 1

We measured system efficiency both by the average number of questions considered (answered or skipped) to find the target case and by the average number of skipped questions during a dialog. Fig. 5 shows the percentage of improvement of AIAS+ over the baseline approach for various numbers of questions presented at each dialog step. The simulated user considers the presented questions in the

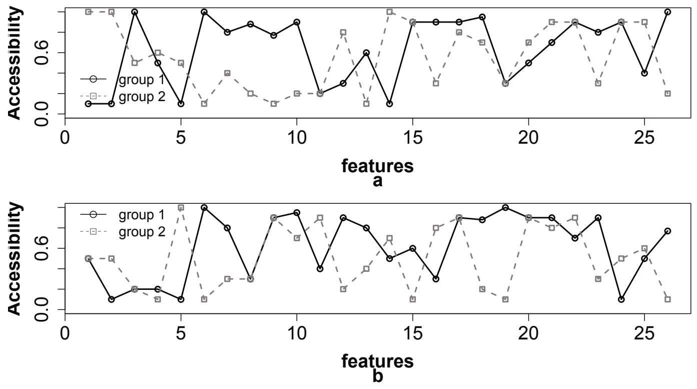


Fig. 4. Candidate accessibility distributions of features in the Automobile domain, used for studying the effect of the number of questions per step of a dialog on the performance of AIAS+

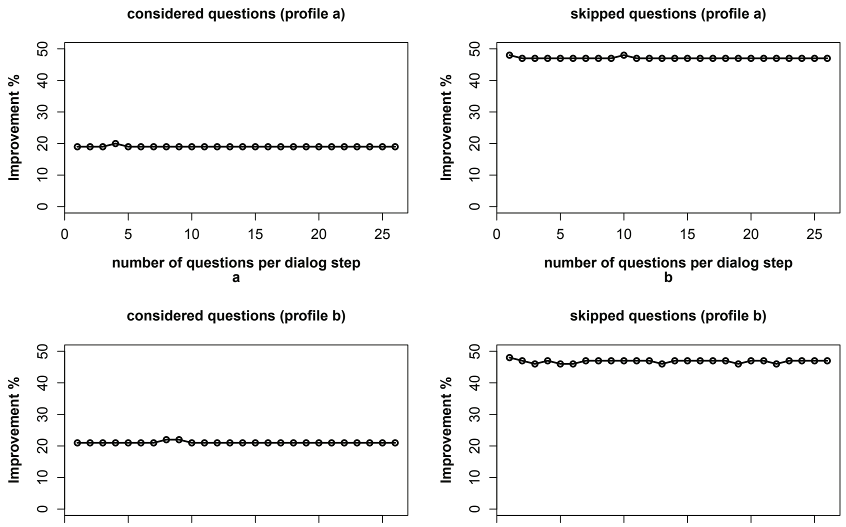


Fig. 5. The effect of having different numbers of questions per dialog step on performance of AIAS+ over baseline in the Automobile domain

order of their ranking, with the highest-ranked question first, and answers the first question it is able to answer. The system's hypotheses about group memberships are updated based on the skipped/answered questions at that step, and a set of new questions are represented to the user. This process continues until either the target case is found or the unknown features in the remaining cases are identical. The experiments cover various number of questions per dialog ranging from 1 to 26 for the Automobile domain.

Fig. 5 compares the performance of AIAS+ to the baseline method in the Automobile domain for the two accessibility distributions tested. Parts a and c of Fig. 5 shows that for both tested accessibility distributions, on average AIAS+ performs 20% better (i.e. 3.14 questions) compared to the baseline in terms of the average number of questions the simulated user considers, where the number of questions presented at each step of the dialog range from 1 to 26.

Parts b and d of Fig. 5 show that for both tested accessibility distributions, AIAS+ on average performs 47% better than the baseline in terms of the average number of questions presented to the user which the user is unable to answer, when the number of questions per step of the dialog ranges from 1 to 26. Our explanation is that AIAS+'s consideration of user response likelihood in ranking questions reduces the chance of skipping several questions in a row. This behavior is similar to AIAS.

6.3 Question 3: Sensitivity of AIAS+ to Component Weightings

AIAS+ depends on balancing considerations of information gain, accessibility, and user classification. Consequently, an important question is how sensitive results are to the specific balance of factors selected. To test the level of sensitivity to specific tunings, we assessed average performance over three domains, with a range of weighting profiles and observed effect on the average number of questions in a dialog. Weighting profiles were generated by the formula:

$$\begin{aligned} BalancedRank(f) \equiv Gain(f) \times (\alpha \times Accessibility(f) + \\ (1-\alpha) \times Pos(E(VarGain(f)))) \end{aligned} \tag{7}$$

When α equals 0.5, rankings correspond to those of AIAS+.

Our experiments used three sample domains (Automobile, Flags and Housing), with feature accessibility distributions for different user groups, as shown at parts a,b and c of Fig. 6. Feature accessibilities were assigned based on the results in Fig. 3, to select accessibilities under which there is a difference in the performance of AIAS+ and AIAS. Fig. 7 shows the improvement of different weighting profiles over AIAS ($\alpha = 1.0$) when α ranges from 0.1 to 0.9 in terms of the average numbers of considered questions.

In the three sample domains, the various tunings of equation 7 ,on average outperform AIAS by 4%, 4% and 8% in the Automobile, Flag and Housing domains respectively. With the optimal tuning, the performance gain compared to AIAS is 5%, 4% and 12% respectively.

Different weightings of equation 7 usually result in approximately the same average performance, provided all three factors are considered; we hypothesize that this reflects that different profiles may perform better for different dialogs. For a given domain and accessibility distributions, slight improvements may be achieved

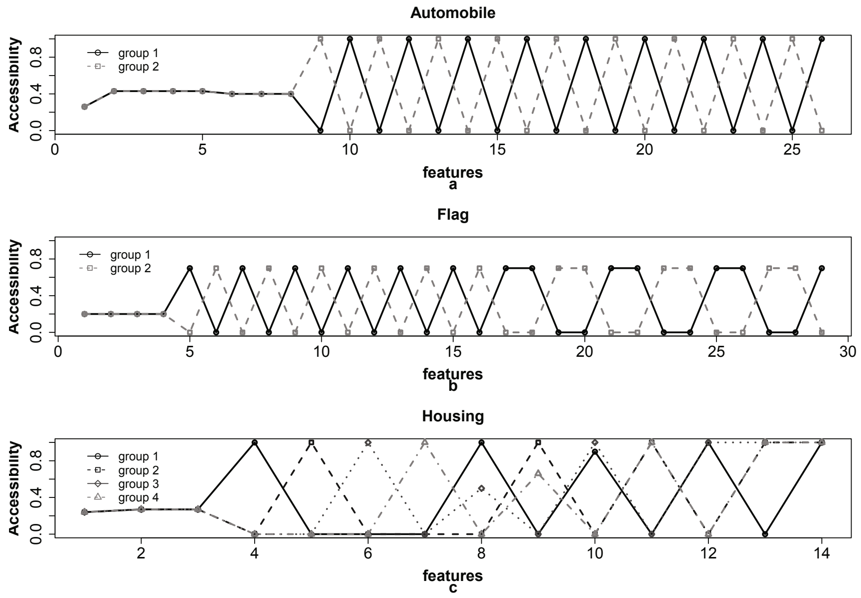


Fig. 6. Candidate accessibility distributions of features in Automobile, Flag and Housing domains

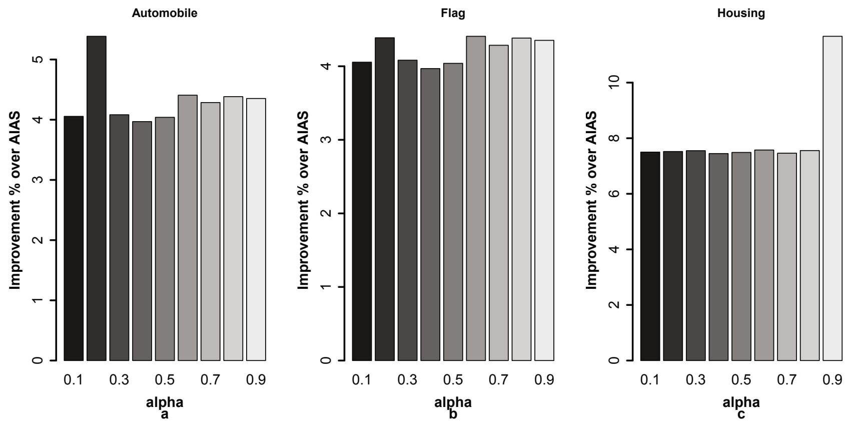


Fig. 7. Balancing the weighting factors

by tuning the weighting compared to the basic AIAS. For example, the optimal α value in the Automobile domain is 0.2, and in the Housing domain is 0.9.

One possible explanation for the insensitivity of performance to tuning is that during a dialog there are few moments when the expected variance gain is sufficiently large to strongly influence question selection. Provided the question selection method takes the expected variance gain into account to some extent, the desired improvement is achieved.

7 Conclusion

This paper presents and evaluates AIAS+, a method for customized question selection in CCBR. AIAS+ improves on our previous work on Accessibility Influenced Attribute Selection by considering not only feature discriminativeness and accessibility, but also the ability of questions to help classify users, in order to customize question selection for their characteristics.

Experiments show the benefits of AIAS+ over both AIAS and non-customized methods. Regardless of the number of questions presented to a user at each step of a dialog, AIAS+ outperforms a pure entropy-based approach. AIAS+ is especially beneficial in reducing the number of questions in a dialog which must be skipped. Finally, the experimental results suggest that AIAS+ generally outperforms AIAS in real domains and that the benefit has little sensitivity to the weights assigned to the component factors.

Interesting questions for future research include how different domains affect the tuning of factors to balance in question selection, the role of feature dependencies in question selection, and strategies for considering larger segments of the dialog than single questions.

References

1. Aha, D., Breslow, L., Munoz-Avila, H.: Conversational case-based reasoning. Applied Intelligence 14, 9–32 (2001)
2. Jalali, V., Leake, D.: Customizing question selection in conversational case-based reasoning. In: Proceedings of the 2012 Florida AI Research Symposium, FLAIRS (in press, 2012)
3. Watson, I.: Applying Case-Based Reasoning: Techniques for Enterprise Systems. Morgan Kaufmann, San Mateo (1997)
4. Shimazu, H., Shibata, A., Nihei, K.: Expertguide: A conversational case-based reasoning tool for developing mentors in knowledge spaces. Appl. Intell. 14(1), 33–48 (2001)
5. Aha, D.W., McSherry, D., Yang, Q.: Advances in conversational case-based reasoning. Knowl. Eng. Rev. 20(3), 247–254 (2005)
6. Aha, D.W., Maney, T., Breslow, L.A.: Supporting Dialogue Inferencing in Conversational Case-Based Reasoning. In: Smyth, B., Cunningham, P. (eds.) EWCBR 1998. LNCS (LNAI), vol. 1488, pp. 262–273. Springer, Heidelberg (1998)
7. Gupta, K.M.: Taxonomic Conversational Case-Based Reasoning. In: Aha, D.W., Watson, I. (eds.) ICCBR 2001. LNCS (LNAI), vol. 2080, pp. 219–233. Springer, Heidelberg (2001)
8. Gu, M., Aamodt, A.: A Knowledge-Intensive Method for Conversational CBR. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS (LNAI), vol. 3620, pp. 296–311. Springer, Heidelberg (2005)
9. Quinlan, J.R.: Induction of decision trees. Machine Learning 1(1), 81–106 (1986)

10. Doyle, M., Cunningham, P.: A Dynamic Approach to Reducing Dialog in On-Line Decision Guides. In: Blanzieri, E., Portinale, L. (eds.) EWCBR 2000. LNCS (LNAI), vol. 1898, pp. 49–60. Springer, Heidelberg (2000)
11. McSherry, D.: Minimizing dialog length in interactive case-based reasoning. In: Proceedings of the seventeenth International Joint Conference on Artificial Intelligence (IJCAI 2001), pp. 993–998. Morgan Kaufmann, San Mateo (2001)
12. Göker, M.H., Roth-Berghofer, T.R., Bergmann, R., Pantleon, T., Traphöner, R., Wess, S., Wilke, W.: The Development of HOMER: A Case-Based CAD/CAM Help-Desk Support Tool. In: Smyth, B., Cunningham, P. (eds.) EWCBR 1998. LNCS (LNAI), vol. 1488, pp. 346–357. Springer, Heidelberg (1998)
13. Shimazu, H.: Expertclerk: Navigating shoppers' buying process with the combination of asking and proposing. In: Proceedings of the Seventeenth International Joint Conference on Articial Intelligence, pp. 1443–1448 (2001)
14. Yang, Q., Wu, J.: Enhancing the effectiveness of interactive case-based reasoning with clustering and decision forest. Computational Intelligence 14(1), 49–64 (2001)
15. McSherry, D.: Conversational case-based reasoning in medical decision making. Artificial Intelligence in Medicine 52(2), 59–66 (2011)
16. McSherry, D.: Increasing dialogue efficiency in case-based reasoning without loss of solution quality. In: Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI 2003), pp. 121–126. Morgan Kaufmann, San Mateo (2003)
17. Carrick, C., Yang, Q., Abi-Zeid, I., Lamontagne, L.: Activating CBR Systems through Autonomous Information Gathering. In: Althoff, K.-D., Bergmann, R., Branting, L.K. (eds.) ICCBR 1999. LNCS (LNAI), vol. 1650, pp. 74–88. Springer, Heidelberg (1999)
18. Kohlmaier, A., Schmitt, S., Bergmann, R.: A Similarity-Based Approach to Attribute Selection in User-Adaptive Sales Dialogs. In: Aha, D.W., Watson, I. (eds.) ICCBR 2001. LNCS (LNAI), vol. 2080, pp. 306–320. Springer, Heidelberg (2001)
19. Thompson, C., Göker, M., Langley, P.: A personalized system for conversational recommendations. Artificial Intelligence Research (JAIR) 21, 393–428 (2004)
20. Ricci, F., Venturini, A., Cavada, D., Mirzadeh, N., Blaas, D., Nones, M.: Product Recommendation with Interactive Query Management and Twofold Similarity. In: Ashley, K.D., Bridge, D.G. (eds.) ICCBR 2003. LNCS, vol. 2689, pp. 479–493. Springer, Heidelberg (2003)
21. Mirzadeh, N., Ricci, F., Bansal, M.: Feature selection methods for conversational recommender systems. In: IEEE International Conference on e-Technology, e-Commerce, and e-Services, pp. 772–777 (2005)
22. Frank, A., Asuncion, A.: UCI machine learning repository (2010)

Feature Weighting and Confidence Based Prediction for Case Based Reasoning Systems

Debarun Kar, Sutanu Chakraborti, and Balaraman Ravindran

Department of Computer Science and Engineering,
Indian Institute of Technology Madras, Chennai- 600036, India
{debarunk,sutanuc,ravi}@cse.iitm.ac.in

Abstract. The quality of the cases maintained in a case base has a direct influence on the quality of the proposed solutions. The presence of cases that do not conform to the similarity hypothesis decreases the alignment of the case base and often degrades the performance of a CBR system. It is therefore important to find out the suitability of each case for the application of CBR and associate a solution with a certain degree of confidence. Feature weighting is another important aspect that determines the success of a system, as the presence of irrelevant and redundant attributes also results in incorrect solutions. We explore these problems in conjunction with a real-world CBR application called InfoChrom. It is used to predict the values of several soil nutrients based on features extracted from a chromatogram image of a soil sample. We propose novel feature weighting techniques based on alignment, as well as a new alignment and confidence measure as potential solutions. The hypotheses are evaluated on UCI datasets and the case base of Infochrom and show promising results.

1 Introduction

The principal assumption that is made when developing case-based reasoning (CBR) systems is that "*similar problems have similar solutions*". However, this traditional view of CBR may not hold good across the entire case base. This may be primarily due to one or more of the following limitations:

– The inability to encode a similarity measure that successfully captures the utility of each case for a particular problem scenario.
– An inappropriate feature weighting scheme that fails to identify the relevance of a particular feature in solving a given problem.
– Absence of relevant features.
– The presence of noise in the data.
– Lack of proper case base maintenance procedure.

The first three limitations are mostly caused by the lack of sufficient knowledge about the problem domain, as it is not always available in an explicit form. On the other hand, noise is often introduced into the case base due to incorrect recording of past experiences, or the inclusion of cases which have no relevance to the problem solving domain under consideration. This results in experience that is neither representative of the domain nor comprehensive in the sense that it does not aid us in solving a diverse range of problems. Therefore, it affects the primary source of knowledge, based on which concrete decisions are made. This often leads to poor quality solutions, thus leading to the degradation in performance of the entire CBR application. However, assuming that such limitations exist, it is essential to quantify the ability of a CBR application in solving a particular problem.

Now, since a typical CBR system is developed based on the similarity hypothesis, it is important to measure the extent to which the similarity assumption holds good for a particular CBR application. This property of the case base is called alignment. Alignment can be measured either *locally* or *globally*. Local alignment is calculated for each case by considering its neighborhood, thus giving us an idea about the surface induced by the similarity measure in that specific region of the case base, while global alignment calculates the alignment of the entire case base, thus providing us with an overview of the effect of the similarity measure when the entire case base acts as the neighborhood. Therefore, it not only aids the system designers to find better similarity measures, but also identifies the suitability of CBR for the application domain. This, we believe, can be a significant contributing factor in not only quantifying the level of confidence for proposed solutions, but also determining the appropriate set of feature weights for the CBR system to use in different problem scenarios.

Feature weighting, as mentioned before, is an important aspect of any prediction system, as it tries to identify the root causes for certain observed behaviors by estimating the relative impact of each feature on a particular target variable. An automated feature weighting method based on the analysis of the data is especially useful in domains that involve a large number of features, as it is practically infeasible for a domain expert to manually encode the weight of each feature. It is also applicable in situations where the implicit knowledge about the feature weights is unavailable even to the experts. We hypothesize that an optimal set of feature weights can be determined by employing techniques based on alignment.

The aim of this paper is two-fold. Firstly, we want to emphasize the need to quantify the reliability of a solution proposed by a system for each test case. Secondly, we want to assert the importance of determining a suitable set of feature weights to enhance our knowledge about the domain and predict successfully. Both of the above problems are addressed in this paper with respect to a practical application[1] called InfoChrom [1]. We review this particular CBR system in section 2 and also present the motivation for our work. In section 3, we propose three novel feature weighting algorithms. The first one is suited for both classification and regression tasks. The second one is meant only for problems of the classification domain and is based on a complexity measure called GAME$_{class}$

[1] This work is supported by a grant from the Department of Science and Technology, Government of India, and is done in collaboration with the Murugappa Chettiar Research Centre, Chennai.

[2]. We adapt it for regression tasks by proposing a new alignment measure called GAME$_{reg}$. This is followed by a confidence measure based on local alignment of the cases in a case base. In section 4, experimental results on UCI data sets and the case base used by InfoChrom are presented and analyzed. A brief review of previous work related to feature weighting, alignment and confidence measures is given in section 5. Finally, we end our paper with conclusions and future prospects for our work in section 6.

2 InfoChrom

In [1], Khemani *et al.* described an application of CBR for the estimation of soil properties, called InfoChrom. Human determination of soil properties is only qualitative, whereas testing the amounts of several macro and micro nutrients through chemical analysis is a highly expensive, time consuming and laborious process. Hence, there is a great potential in developing an application with a profound socio-economic impact to predict the quantitative information of various soil properties from previously stored records.

The case base for InfoChrom consisted of 10000 samples, each represented in terms of 153 features extracted from a soil chromatogram image (Fig. 1) and the corresponding values of 15 soil properties. This forms a standard feature vector representation, with the problem part of a case being the values of 153 numeric features extracted by image processing techniques and the solution part consists of 15 real valued soil nutrients obtained by chemical analysis. Thereafter, by defining local and global similarity measures with an appropriate amalgamation function and using the hypothesis "*similar chromatograms have similar soil properties*", they predicted the values of soil nutrients for new soil samples. Khemani *et al.* [1] gives a detailed description about chromatograms, the attribute schema used and the target variables.

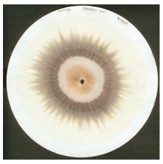


Fig. 1. A soil chromatogram sample

2.1 Drawbacks and Motivation

Over time, new cases and attributes were added to the case base and it currently consists of 15166 cases, each represented in a 176 dimensional feature space, along with a set of 15 target variables. Recently, it was noticed that there is degradation in the performance of the system for a set of new test images. So, while trying to analyze the cause for poor performance, we found that feature relevance was determined by applying four different sets of feature weights and visually comparing the most similar images retrieved. However, we believe that the determination of relative importance of the features should either be on the basis of already available expert knowledge or knowledge extracted from the data itself, which in turn can be validated by domain experts. Now, in this case, there are a large number of attributes and it is very difficult to manually set the weights for each independent variable. Moreover, the effect of the individual features on each one of the 15 target variables may vary and even be unknown to soil scientists. Therefore, correlation and mutual information (MI) analysis were performed on this data with two objectives. First, to identify the set of redundant attributes by identifying features with high correlation between them. Secondly, to find out features which are most relevant for a particular nutrient by looking at the correlation and MI values between the target variables and the features. The corresponding correlation and MI values were found to be very low (typically between -0.1 to 0.1), indicating that there may not be adequate information in the extracted features or in the current weight setting to accurately model the behavior of the soil nutrients.

Another common cause for the increase in error in prediction tasks is the presence of noise in the data set. So, we looked at the ranges of the nutrients and found some discrepancies with respect to the normal ranges for a few target variables. For example, the case base consisted of cases with values of the target variable pH as 25.5 and 19. As we know that the range for pH is 0 to 14, clearly these are erroneous examples. Such cases which are indicative of measurement or recording error tend to produce incorrect results. The presence of very similar problems, but with completely different solutions can also result in high prediction error, as it would reduce the alignment of the case base. Several dissimilar images with the same values for all or some of the nutrients are also present. In the case of InfoChrom, new cases and features were added without determining their relevance to the case base. In the process, several such irrelevant and redundant cases got added and the performance of the system dropped over time. Principal components regression (PCR) and partial least squares regression (PLSR) were performed to extract a relevant combination of features for the prediction task. It resulted in poor performance on the new case base, but the prediction was more accurate on modified datasets formed for each nutrient, by removing several sets of possibly noisy values, identified by kernel density estimation (KDE) plots. The latter performance was still not satisfactory and required further introspection and analysis.

However, due to the lack of domain expertise, it is often difficult for system designers to solve the above problems by answering questions such as "*what is the correct range of values for each of the soil nutrients*", or "*which one of the identical cases has the correct solution and should be included in the case base*". Also, in some cases even the domain expert may not have the adequate background knowledge to identify anomalies. In such situations where knowledge is not available in an explicit form, automated techniques to infer them can be adopted. For example, a suitable set of feature weights can be determined along with the identification of regions of the case space which are unamenable for the application of CBR, due to low alignment. Hence, when a given query falls in that space, the CBR system should be able to say "*I am only p% confident of my prediction for this particular query*", where p is small. This is important to gain trust from the end users of the system, because otherwise they believe that the system is completely confident about every prediction it makes and their faith starts to wane as they discover disparities in the produced results. In our case, the farmers started to lose confidence in the system because of some incorrect test results and it became a serious concern for the success of the application. Now based on the obtained solutions and their corresponding confidence values, a domain expert can validate the results for the cases he is aware of and try to re-examine otherwise unknown information using his knowledge of the subject. A feedback can then be taken which will enrich both the system and its designer's knowledge about the domain and may result in a better application in the future.

3 Our Approaches

In section 3.1, we introduce novel feature weighting techniques based on alignment. An algorithm for determining the level of confidence for a particular prediction is introduced in section 3.2.

3.1 Determining Feature Weights with Global Alignment

We use the alignment of a case base to calculate a set of feature weights with the basic intuition that the features which contribute more towards the global alignment should have higher weights. This is because alignment is a measure of the suitability of CBR to a particular problem domain. Hence the features which increase this suitability are those that are more useful for the application of CBR. Detailed explanations about the algorithms and the alignment measures used to achieve this are presented later in this section.

Now, let us consider a set of N cases C_1, C_2,..., C_N, with the problem space P consisting of the values of q features F_1, F_2, ..., F_q and the solution space S is formed by the target variable T. Also, let the weight of feature F_i be w_i $(1 \leq i \leq q)$. Then the similarity in the problem space between two cases C_a and C_b can be calculated as the weighted sum of their feature space similarities :

$$SimP(C_a, C_b) = \sum_{i=1}^{q} w_i * SimF(C_{ai}, C_{bi}) \tag{1}$$

where, $SimF(C_{ai},C_{bi})$ denotes the similarity between the corresponding values of feature F_i for the cases $C_a$and C_b, and is computed as follows :

$$SimF(C_{ai}, C_{bi}) = \frac{Max(DistF_i) - DistF(C_{ai}, C_{bi})}{Max(DistF_i) - Min(DistF_i)} \tag{2}$$

Algorithm 1. Algorithm to calculate feature weights based on alignMassie

```
1. For each feature F$_i$, do :
   (a) Calculate local alignment of each case C$_j$ as :
   (b) Calculate global alignment of case base (CB) as :
   (c) Assign weight of feature F$_i$ as : w$_i$ = alignMassie(CB).
```

$$alignMassie(C_j) = \frac{\sum\limits_{k \epsilon NNP(C_j)} SimF(C_{ki}, C_{ji}) * SimS(C_k, C_j)}{\sum\limits_{k \epsilon NNP(C_j)} SimF(C_{ki}, C_{ji})} \quad (3)$$

$$alignMassie(CB) = \frac{\sum\limits_{j \epsilon CB} alignMassie(C_j)}{|CB|} \quad (4)$$

Here, *DistF(C$_{ai}$,C$_{bi}$)* denotes the city-block distance between the values C$_{ai}$ and C$_{bi}$. *Max(DistF$_i$)* and *Min(DistF$_i$)* are the maximum and minimum distances between all pairs of values for feature F$_i$ respectively. The target space similarity between any two cases is calculated in the same way.

Feature Weights by alignMassie (Weights$_{alignMassie}$) : Our first algorithm based on the alignment measure proposed by Massie [3] is straightforward and is applicable to problems of both classification and regression domains. Here we take each feature in turn and calculate the local alignment of each case with alignMassie and aggregate these local effects into a global alignment measure. The global alignments thus calculated for each feature are then assigned as the respective feature weights. We chose alignMassie to calculate the alignment because it is both intuitive and respects the problem-solution asymmetry with good overall predictive ability. In Algorithm 1, *SimF* and *SimS* denotes the problem side similarity based on feature F$_i$ and the solution side similarity respectively and *NNP(C$_j$)* is the set of nearest neighbors for case C$_j$ on the problem side.

Feature Weights by GAME$_{class}$ (Weights$_{GAME_{class}}$): GAME$_{class}$ [2] is an extended version of the Global Alignment MEasure (GAME) for the classification domain. The principal idea behind it is to sort the cases in order of their problem side similarities and see whether the nearest neighbors belong to the same class. This gives us an idea as to whether adapting the solutions of problems similar to the target case will help in predicting its class label. We use a similar approach to find feature weights. Instead of sorting cases, we sort values of each feature F$_i$ according to their *SimF* similarities and the corresponding class labels are noted. According to our intuition, the feature weights should be inversely proportional to the number of flips (the number of interchanges between two class labels). For example, let us assume that for a cancer prediction task we have 9 cases represented in a 5-dimensional feature space with binary

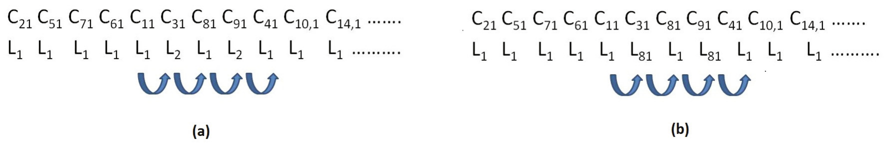
Algorithm 2. Algorithm to determine feature weights based on GAME$_{class}$

```
1. For each feature F$_i$, do :
    (a) Sort the values of F$_i$ in ascending order.
    (b) Replace case identifiers of the sorted sequence with corresponding class labels.
    (c) Calculate the number of flips (Flips) from one class label to another.
    (d) Compute GAME$_{class}$ as
            $GAME_{class} = log(\frac{MaxFlips - MinFlips}{Flips - MinFlips})$    (5)
        MaxFlips=(n-1) and MinFlips=(k-1) in a case base of 'n' cases and 'k' features.
    (e) Assign weight of feature F$_i$ as : w$_i$ = GAME$_{class}$.
```

classes Y(Yes) and N(No), indicating the presence or absence of cancer. Considering the values of feature F_1 for all the 9 cases, let the sorted version obtained be $C_{11}C_{51}C_{31}C_{21}C_{41}C_{61}C_{91}C_{71}C_{81}$, along with the corresponding class label sequence as YYYYNYNNN. With 3 flips, F_1 should thus receive higher weight than feature F_2 with the class string YNYNYNYNY, which gets the lowest weight. A feature with the class sequence as YYYYYNNNN or NNNNYYYYY should be given the highest weight. Algorithm 2 calculates GAME$_{class}$ when cases are represented with feature F_i only and assigns it to the weight w_i of that feature.

Modified Version of GAME$_{class}$ for Regression Tasks (Weights$_{GAME_{reg}}$): One drawback of GAME$_{class}$ and hence, Algorithm 2 is that it is applicable to classification problems only. But, for a real-valued prediction task we do not have discrete set of classes. To address this issue, we propose a novel alignment measure and a feature weighting algorithm based on that. We divide the range of the target variable into small subsets and assign a class label to each of them. Ideally, the subsets should be indicative of the clustering tendency of that random variable. A simple solution is to apply an unsupervised discretization technique like binning, where a bin corresponds to one subset and hence, a particular class. We can then use the obtained class information to find the alignment of the case base and hence, a set of feature weights for the regression task.

However, it is important to note that, unlike classification, a flip between two classes obtained in the case of regression is actually a flip between two ranges of the target variable. Therefore, the significance of a flip between two adjacent classes can be drastically different from a flip between the two extreme ones. To illustrate this, let us take the example of predicting housing prices (₹ per square feet) in Mumbai based on the area of the house, status of the population and some other features. Assume that the prices lie in the range between ₹1000 and ₹100,000, and a simple binning algorithm uses a bin size of 1000 to form 99 equally sized bins or classes. The classes are labelled from L_1 to L_{99} for the 99 consecutive bins. Now let us consider a portion of the sorted sequence for a feature F_1($C_{21}C_{51}...C_{14,1}$) and its corresponding class string as shown in Fig 2.



C21 C51 C71 C61 C11 C31 C81 C91 C41 C10,1 C14,1 ....
L1 L1 L1 L1 L1 L2 L1 L2 L1 L1 L1 ....
(a)
C21 C51 C71 C61 C11 C31 C81 C91 C41 C10,1 C14,1 ....
L1 L1 L1 L1 L1 L81 L1 L81 L1 L1 L1 ....
(b)

Fig. 2. Illustration for the motivation behind flip weights

Considering F_1 to be the area of the house and the samples shown in the figure to be houses of similar areas, we expect all of them to have the same price range and hence, the same class label. However, it can be observed in Fig 2(a) that there are a few flips between the class L_1, representing the range ₹1000 - ₹2000 and L_2, denoting the range ₹2000 - ₹3000 for houses of similar areas. The higher price range for cases C_3 and C_9 may be because of the higher status of the population in that area as compared to the other cases. Now, based on feature F_1 alone if we had performed a 3-nearest neighbor based majority voting prediction for a test case T with the value of F_1 as C_{81}, the class label assigned to T would be L_2 and not L_1. A value within the range of ₹2000 - ₹3000 would then be predicted for T. However, compare this with the situation depicted in Fig 2(b), where there is a flip between class L_1 and L_{81}, the latter being the class corresponding to the range ₹81,000 - ₹82,000. So for the same test case T, the prediction will now be in the range of ₹81,000 - ₹82,000. Clearly, the prediction in the second case results in higher error than the first. Therefore, a flip between distant cases should be penalized more than a flip between nearby cases. This is captured by associating weights with each flip based on the proximity of the classes involved in the flip.

One can use the relative difference between the class indices involved in the flip as the weight of that flip. However, if we use a binning technique that divides the interval into unequal bins, then the above method may not be able to capture the real significance of the flips. Here, we used the similarity between the mid-points of the ranges of the classes involved to calculate the weights. Algorithm 3 gives a detailed description. Finally, the set of feature weights is calculated using a modified alignment measure for regression tasks, called GAME_{reg}, as mentioned in Algorithm 4. Please note that a small constant can added to the denominator of $w_{flip<i,j>}$ and GAME_{reg} to avoid division by zero when SimF returns zero and $\text{Flips}_{weighted}$ is equal to $\text{MinFlips}_{weighted}$ respectively.

Algorithm 3. Algorithm to determine the weight of a flip

```
1. For each pair of classes $< L_i, L_j >$, do :
   (a) Calculate the similarity between the midpoints of $L_i$ and $L_j$, denoted as $Mid(L_i)$ and $Mid(L_j)$ respectively, by Equation 2 as $SimF(Mid(L_i), Mid(L_j))$.
   (b) Compute the weight of the flip between $L_i$ and $L_j$ as :
       $$w_{flip<i,j>} = \frac{1}{SimF(Mid(L_i), Mid(L_j))}$$
```

Algorithm 4. Algorithm for determining feature weights using GAME$_{reg}$

```
1. Discretize the target variable into distinct classes using any unsupervised discretization technique like binning.
2. Calculate the weights of flips $w_{flip<i,j>}$ between all pairs of classes L$_i$ and L$_j$ using Algorithm 3.
3. For each feature F$_i$, do :
   (a) Sort the values of F$_i$ in ascending order.
   (b) Replace the case identifiers of the sorted sequence with the corresponding class labels.
   (c) Calculate GAME$_{reg}$ as :
       $$GAME_{reg} = log(\frac{MaxFlips_{weighted} - MinFlips_{weighted}}{Flips_{weighted} - MinFlips_{weighted}}) \quad (6)$$
       where,
       MinFlips$_{weighted}$ = Sum of the flip weights for all consecutive pairs of classes $< L_i, L_j >$,
       MaxFlips$_{weighted}$ = Sum of the top (k-1) highest flip weights + [(n-(k-1)) * highest_flip_weight], considering that the flip weights are sorted in descending order and $n$ being the number of cases and $k$ the number of classes.
   (d) Assign the weight of feature F$_i$ as : w$_i$ = $GAME_{reg}$.
```

3.2 Confidence of a Prediction

Given that we have found a suitable set of feature weights, the CBR system may still be unable to predict with 100% accuracy for all the test cases. Associating a confidence value with every prediction is thus necessary for assessing the quality of the prediction. We use alignMassie to compute the local alignment of each case in the case base. Then the confidence for the solution of a previously unseen case C_T is calculated as the weighted mean of the local alignments of its nearest neighbors, with the problem side similarities between C_T and its neighbors acting as weights. This is because, higher the alignments of the most similar cases (C_i) of C_T, higher should be the confidence while adapting their solutions. The confidence measure is thus given by :

$$Confidence(S_T) = \frac{\sum\limits_{i \epsilon NNP(C_T)} SimP(C_T, C_i) * alignMassie(C_i)}{\sum\limits_{i \epsilon NNP(C_T)} SimP(C_T, C_i)} \quad (7)$$

where S_T is the proposed solution for case C_T and *SimP(C_T,C_i)* is calculated using equation 1.

4 Evaluation

In our experiments we use k-NN based retrieval strategy to propose a solution for a test case. We calculate its similarity with all the other cases in the case base using equation 1. Then we obtain the solution for the test sample by taking a weighted average of the solutions of its k nearest neighbors, with the problem side similarities acting as weights. Please note that $\mathrm{w}_i = 1$ ($1 \leq i \leq q$) when no feature weighting is employed while calculating similarity between the cases.

4.1 Datasets

To demonstrate the effectiveness of the proposed feature weighting algorithms, we compared Algorithm 2 with the mutual information (MI) based feature weighting technique on four classification data sets from the UCI machine learning repository. In our experiments, when computing the feature weights based on MI, the continuous valued attributes were discretized using Fayyad & Irani's algorithm [4]. Algorithm 4 was tested on two regression data sets from the same repository and also on the soil data used by InfoChrom. Performances reported on the UCI datasets are actually average performances of the algorithms on 10 random train-test splits, with 70% of the original data used as the case base and the rest for testing. To calculate alignMassie and to predict based on the k-NN algorithm, we experimented with k ranging from 1 to 10. The best performance was given by the set of feature weights and the value of k which caused the highest increase in global alignment. All the results reported are based on that particular value of k ($1 \leq k \leq 10$) and is different for different datasets.

4.2 Performance Measures

We measured performance in terms of classification accuracy and relative error for classification and regression tasks respectively. When no confidence value is associated with a solution, the performance of an algorithm for a test set T is calculated using equations 8 and 9. We propose two new measures (eqns 10 and 11) to evaluate the confidence augmented solutions for both types of prediction tasks by taking a weighted mean of the accuracies or errors for all the test samples, where the confidence values act as weights. Therefore, if a system with a higher level of confidence for a case predicts its solution with high error, it will be penalized more than if it predicts a case with same error but with a lower level of confidence.

$$Accuracy = \frac{|N_C|}{|T|} \tag{8}$$

$$RelativeError = \frac{1}{|T|}\sum_{i=1}^{|T|} \left|\frac{Original_i - Predicted_i}{Original_i}\right| \tag{9}$$

$$Accuracy_{Confidence} = \frac{\sum\limits_{i \epsilon N_c} Confidence(S_i)}{\sum\limits_{j \epsilon T} Confidence(S_j)} \tag{10}$$

$$RelativeError_{Confidence} = \frac{\sum_{i=1}^{|T|} \left|\frac{Original_i - Predicted_i}{Original_i} * Confidence(S_i)\right|}{\sum_{i=1}^{|T|} Confidence(S_i)} \tag{11}$$

Here, N_C denotes the set of correctly classified samples, $Original_i$ and $Predicted_i$ are the original and predicted solutions for case C_i. Our confidence evaluation procedure is different from previous work [5, 6], where confidence was evaluated by comparing the median errors or accuracies for different degrees of confidence.

4.3 Experimental Results and Observations

The performance of algorithms 1 and 2, both with and without confidence are reported in Table 1. They are compared with the case when no feature weighting scheme is used and when MI is used to decide the feature weights. Results in **boldface** and *italics* indicate significant differences (with two-tailed t-tests at confidence=0.05) with respect to the application of feature weights and confidence measures respectively. Results accompanied by * imply significant differences of that feature weighting algorithm when compared with Weights_{MI}.

It can be observed that $\text{Weights}_{alignMassie}$ causes noticeable improvement only in the case of Glass Identification dataset, while there is significant improvement in accuracies for the retrieval algorithm with $\text{Weights}_{GAME_{class}}$ when compared to that without any feature weights. $\text{Weights}_{GAME_{class}}$ performs almost similarly or even better than Weights_{MI}, except for the Waveform-40 dataset. We particularly chose the Waveform-40 data for evaluation of our feature weighting algorithms because, it is known to contain noisy attributes, especially the latter 19 completely noisy features with mean 0 and variance 1, that were included in addition to the 21 attributes of the Waveform-21 dataset. Therefore, a good feature weighting algorithm will be able to capture the real significance of each attribute and give very low or zero weights to these 19 features. We found that the MI based feature weighting technique gives zero weight to 21 out of the 40 attributes, including the last 19 attributes, while $\text{Weights}_{GAME_{class}}$ gives very low weights to the irrelevant features, but does not capture the importance of the feature weights as good as the MI based measure. This probably causes the difference in accuracies between the two methods for Waveform-40. The correlation between the feature weights obtained by $\text{Weights}_{alignMassie}$ and $\text{Weights}_{GAME_{class}}$ with Weights_{MI} are shown in Table 2. It highlights an agreement between $\text{Weights}_{GAME_{class}}$ and Weights_{MI} in the sense that they capture the same relative significance of the features, which can also be verified by similar

Table 1. Comparison of accuracies (in %) of simple k-NN with various feature weighting methods. Results are reported for both confidence-free (Without Conf) and confidence-augmented (With Conf) predictions.

Dataset	Without Weights		With $\text{Weights}_{alignMassie}$		With $\text{Weights}_{GAME_{class}}$		With Weights_{MI}	
	Without Conf	With Conf	Without Conf	With Conf	Without Conf	With Conf	Without Conf	With Conf
Iris	93.555	93.659	93.333	**93.843**	**94.444**	**94.501**	**94.666**	**94.667**
Glass Identification	70.540	*75.747*	**73.243**	***78.217***	**74.666**	***81.854***	**74.594**	***81.341***
Waveform-21	67.393	*69.041*	**67.486**	***69.064***	**69.926***	***71.268****	**68.986**	***70.384***
Waveform-40	61.246	*63.521*	61.253	***63.641***	**65.466**	***67.466***	**70.313**	***72.817***

Table 2. Correlation between the feature weights obtained by Weights$_{alignMassie}$ and Weights$_{GAME_{class}}$ with Weights$_{MI}$

Correlation (ρ)	Datasets			
	Iris	Glass identification	Waveform-21	Waveform-40
ρ(Weights$_{alignMassie}$, Weights$_{MI}$)	-0.282	-0.449	-0.105	0.317
ρ(Weights$_{GAME_{class}}$, Weights$_{MI}$)	0.998	0.812	0.985	0.983

performance levels as shown in Table 1. The increase in performance when the accuracies are augmented with confidence imply that the CBR system is able to better identify the regions in the case-space where it is expecting a good or bad prediction, as the case may be.

Performance of Algorithm 4 on 10 random train-test splits of two UCI regression datasets is shown in Table 3. Significant improvement can be observed in Weights$_{GAME_{reg}}$ over other algorithms on both datasets. It is also evaluated on 1025 queries using the 15,166 cases in the case base of InfoChrom. The modified version of the system formed with Weights$_{GAME_{reg}}$ is hereafter referred to as InfoChrom+. InfoChrom+ augmented with confidence is named InfoChrom++. The average relative errors for the 15 target variables for InfoChrom+ and InfoChrom++ are compared against the existing system and several machine learning approaches like Regression Tree, Artificial Neural Network (ANN), Principal Components Regression (PCR) and Partial Least Squares Regression (PLSR). The results are shown in terms of bar charts in figure 3. Relative errors for PCR and PLSR are shown for the modified datasets (section 2.1).

The noticeable drop in the relative errors of InfoChrom+, as compared to InfoChrom, is primarily due to the fact that Weights$_{GAME_{reg}}$ is able to successfully identify both irrelevant and redundant features. This has been verified for a few target variables using available domain knowledge. For example, soil scientists claim that a large number of spikes in a brownish chromatogram image indicate the presence of humus. Weights$_{GAME_{reg}}$ was found to give higher weights to spike related features, such as its number, height, average width and color. Also, for predicting the target variable EC solution, it gives highest weights to inner region attributes, which is also found to be consistent with the scientists' domain knowledge. Moreover, attributes which are exactly similar in their behavior are given the same weights, thus also helping us to figure out possibly redundant

Table 3. Comparison of relative errors (in %) of simple k-NN with various feature weighting methods on the regression datasets. Results are reported for both confidence-free (Without Conf) and confidence-augmented (With Conf) predictions.

Dataset	Without Weights		With Weights$_{alignMassie}$		With Weights$_{GAME_{reg}}$		With Weights$_{MI}$	
	Without Conf	With Conf	Without Conf	With Conf	Without Conf	With Conf	Without Conf	With Conf
Wine Quality Red	9.32	9.25	9.3	**9.22**	**8.95***	***8.82****	**9.06**	***8.91***
Wine Quality White	8.96	*8.79*	**8.93**	***8.78***	**8.68***	***8.56****	9.16	*8.93*

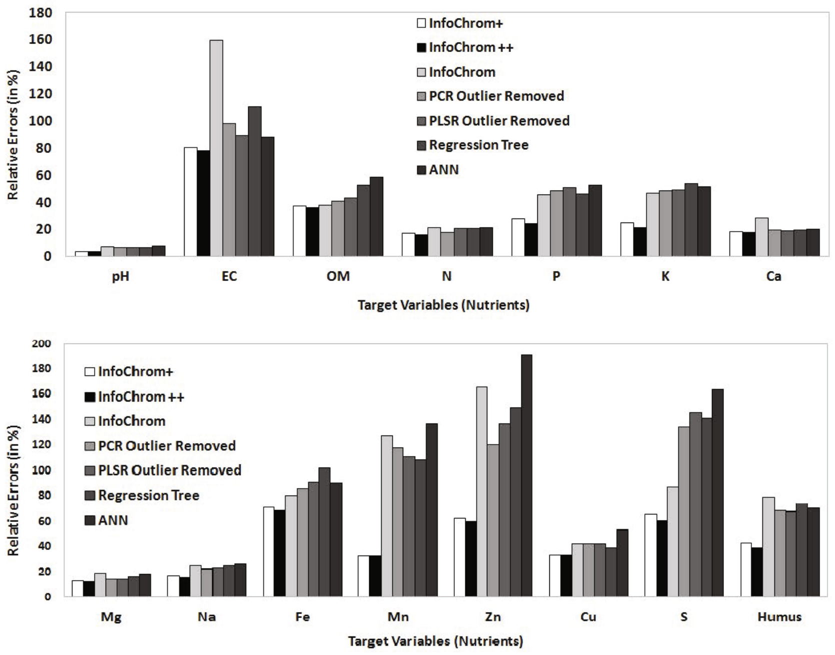


Fig. 3. Comparison of relative errors for 15 target variables between various algorithms

attributes. For example, the data is represented in terms of attributes like the outer region area in pixels, the outer region area in cm, the outer region width in pixels and the outer region width in cm. Three of these four features are clearly redundant and the feature weighting scheme rightly gives the exact same weights to all four of them. The decrease in relative errors for InfoChrom++, as compared to InfoChrom+, demonstrates the effectiveness of our confidence based evaluation measure, as much as it highlights the need for a confidence based prediction mechanism to increase the faith of the user in the system.

5 Related Work and Discussions

Research on feature weighting techniques in both CBR and machine learning has concentrated on trying to eliminate the *"curse of dimensionality"* problem for k-nearest neighbor (k-NN) based approaches, that occurs mainly because of its biased distance function [7]. Several variants of k-NN were proposed [8–10], distinguished in terms of a 5-dimensional framework and compared [11]. Various alignment measures have been proposed for both classification and regression based prediction tasks, as well as for the evaluation of TCBR systems. Massie *et al.* [12] described measures of case base complexity, especially for problems with class labels as their solutions. A local alignment measure was proposed by Lamontagne [13], which was based on the amount of overlap between the neighborhoods of a case in the problem and solution spaces. In [3], alignMassie, a measure which respects the problem-solution asymmetry by giving more weights to nearby cases in the problem side was described. Chakraborti *et al.* [2] proposed GAME$_{class}$ for classification tasks, based on the compression level of an image formed by stacking similar cases and features together. Three measures of global alignment, alignGame, alignMST and alignCorr were suggested and compared by Raghunandan *et al.* [14]. Our measure, GAME$_{reg}$, is different from the existing alignment measures as it is even applicable to classification tasks with non-orthogonal class labels. An example is a five-point Likert scale {1,2,3,4,5} used to rate products in recommender systems. When mapped to class labels, class 2 is more similar to class 3 as compared to class 5. The hierarchical classification of wikipedia articles into categories is another example.

Confidence based prediction has been addressed by Cheetham *et al.* [5, 6]. They suggested a set of properties based on the similarity measure and the cases retrieved by it, that can act as potential indicators of confidence. The best *"confidence indicators"* were then identified and confidence formulae based on historical error rates and fuzzy membership functions of the indicators were proposed. In this case, there are a host of indicators to be considered and their effects need to be appropriately combined to get a final confidence value. This also requires a lot of parameter optimization every time a change is made in the case base to maintain the confidence computation. Our measure is based on only one significant indicator of case base suitability called alignment. Important information about the similarity measures on both the problem and solution sides, as well as their possible effects on the performance of the retrieval algorithm, are captured successfully by alignment. Also, we only need to recompute the alignments of cases which lie within k nearest neighbors of a new case when it is added to the case base, thus requiring less computation.

6 Conclusions and Future Work

In this paper, we have explored the utility of alignment to solve the problem of feature weighting. Two novel feature weighting methods have been introduced using existing measures of global alignment. We have also proposed a new, robust measure of alignment for regression tasks and a feature weighting method based on that has been suggested. The notion of confidence based on local alignment has been introduced. We hope that this will be beneficial for practical applications to develop an essence of trust among the users of the system. Our algorithms have shown significantly better performance on high dimensional real-world data, as well as on UCI datasets. In the future, we would like to explore non-linear machine learning methods like isomap [15] and compare them with our algorithms on high dimensional textual data as applicable in TCBR.

Acknowledgements. The authors would like to thank members of the AIDB Lab at IIT Madras, specially Anand Kumar and Kunal Mulay for their help and support.

References

1. Khemani, D., Joseph, M.M., Variganti, S.: Case Based Interpretation of Soil Chromatograms. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 587–599. Springer, Heidelberg (2008)
2. Chakraborti, S., Cerviño Beresi, U., Wiratunga, N., Massie, S., Lothian, R., Khemani, D.: Visualizing and Evaluating Complexity of Textual Case Bases. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 104–119. Springer, Heidelberg (2008)
3. Massie, S., Wiratunga, N., Craw, S., Donati, A., Vicari, E.: From Anomaly Reports to Cases. In: Weber, R.O., Richter, M.M. (eds.) ICCBR 2007. LNCS (LNAI), vol. 4626, pp. 359–373. Springer, Heidelberg (2007)
4. Fayyad, U.M., Irani, K.B.: Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence, Chambery, France, pp. 1022–1029. Morgan Kaufmann Publishers Inc. (1993)
5. Cheetham, W.: Case-Based Reasoning with Confidence. In: Blanzieri, E., Portinale, L. (eds.) EWCBR 2000. LNCS (LNAI), vol. 1898, pp. 15–25. Springer, Heidelberg (2000)
6. Cheetham, W., Price, J.: Measures of Solution Accuracy in Case-Based Reasoning Systems. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 106–118. Springer, Heidelberg (2004)
7. Wettschereck, D., Aha, D.W., Mohri, T.: A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. Artif. Intell. Rev. 11(1-5), 273–314 (1997)
8. Kelly, J.D., Davis, L.: A hybrid genetic algorithm for classification. In: Proceedings of the 12th International Joint Conference on Artificial Intelligence, IJCAI 1991, vol. 2, pp. 645–650. Morgan Kaufmann Publishers Inc., San Francisco (1991)
9. Aha, D.W.: Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. Int. J. Man-Mach. Stud. 36(2), 267–287 (1992)
10. Wettschereck, D.: A study of distance-based machine learning algorithms. Ph.D. dissertation, Department of Computer Science, Oregon State University (1994)
11. Wettschereck, D., Aha, D.W.: Weighting Features. In: Aamodt, A., Veloso, M.M. (eds.) ICCBR 1995. LNCS, vol. 1010, pp. 347–358. Springer, Heidelberg (1995)
12. Massie, S., Craw, S., Wiratunga, N.: Complexity Profiling for Informed Case-Base Editing. In: Roth-Berghofer, T.R., Göker, M.H., Güvenir, H.A. (eds.) ECCBR 2006. LNCS (LNAI), vol. 4106, pp. 325–339. Springer, Heidelberg (2006)
13. Lamontagne, L.: Textual cbr authoring using case cohesion. In: Proceedings of 3rd Textual Case-Based Reasoning Workshop at the 8th European Conf. on CBR (2006)
14. Raghunandan, M.A., Chakraborti, S., Khemani, D.: Robust Measures of Complexity in TCBR. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 270–284. Springer, Heidelberg (2009)
15. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science (New York, N.Y.) 290(5500), 2319–2323 (2000)

A Case-Based Approach to Mutual Adaptation of Taxonomic Ontologies

Sergio Manzano[1], Santiago Ontañón[2], and Enric Plaza[1]

[1] IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Catalonia (Spain)
{sergio,enric}@iiia.csic.es
[2] Computer Science Department,
Drexel University
Philadelphia, 19104 PA, USA
santi@cs.drexel.edu

Abstract. We present a general framework for addressing the problem of semantic intelligibility among artificial agents based on concepts integral to the case-based reasoning research program. For this purpose, we define case-based semiotics (CBS) (based on the well known notion of the semiotic triangle) as the model that defines semantic intelligibility. We show how traditional CBR notions like transformational adaptation can be used in the problem of two agents achieving mutual intelligibility over a collection of concepts (defined in CBS).

1 Introduction

We propose an approach based on case-based semiotics (CBS) to determine problems in consistency or ambiguity based on the well known notion of the semiotic triangle. This approach aims at supporting the participating agents in evolving their individual ontologies on-demand, in a way that is enough to coordinate their activity in a particular task or subdomain. This participatory ontology is understood as an adaptation of the individual ontologies that converges into a shared mutually consistent and unambiguous ontology guided by our case-based approach to semiotics. Our approach is based on two basic assumptions:

(i) Case-based Assumption: participating agents share their environment and are capable of understanding their case description language(s). They either (1) share the case description language or (2) they share some basic ontology and language that allows them to explain their case description language(s)[1].

(ii) Taxonomy Assumption: Concepts in an ontology are organized in a hierarchy. More complex structures of ontologies are left for future work. In particular, DL-based ontologies require further development of inductive generalization methods before supporting this kind of approach.

This work is a generalization of the *concept convergence* approach [6], in which two agents deliberate about the meaning of a concept using their case-bases in

[1] How to achieve (2) is beyond the scope of this paper, but see [9].

order to a achieve a shared, agreed-upon meaning of a concept. This generalization is due to the fact that concepts do not exist in isolation, but are related to other neighboring concepts in what we currently call an ontology. Concept convergence introduced the use of the semiotic triangle (see Fig. 1) to define concept meaning in a case-based agent. Our view with respect to case-based semiotics (CBS) is that *specific cases* are needed to perform certain forms of reasoningThis paper, in particular, focuses on the problem of mutual intelligibility for artificial agents endowed with a domain ontology. We think that the process by which two agents can adapt their ontologies to active mutual intelligibility requires *reasoning about cases*. Specifically, we think that a purely logic-based approach is not sufficient, and that a view of concept meaning based on classical logical semantics is not sufficient. This issue is a long-standing philosophical debate between the logic-based semantic view of meaning and the semiotic view of meaning (see e.g. [2]). We propose that concept meaning is better modeled by the semiotics approach that has a two-layer description of concepts: the intentional description or definition of a concept (in some formalism) and an extensional description of a concept (as is classically used in CBR).

Although this paper does not deal with *case-based problem solving* (in which new problems are solved using precedents or solved cases), our approach explicitly deals with case-based reasoning in the general sense: performing intelligent tasks by *reasoning about cases*. Moreover, we will show how mutual intelligibility about concepts can be seen as a process of mutual adaptation of taxonomies, using a process that is equivalent to transformational adaptation.

2 Background and Related Work

Most approaches use the *ontology alignment* metaphor to deal with the relationship between two different ontologies; it's a metaphor in that it originates by analogy with molecular sequence alignment [3]. Intuitively, ontology alignment (or matching) is a process that aims at finding "classes of data" that are *semantically equivalent*. Ontology alignment has been studied on database schemas, XML schemas, taxonomies, formal languages, entity-relationship models, and dictionaries. Formally, while matching is the process of finding relationships or correspondences between entities of different ontologies, alignment is a set of correspondences between two (or more) ontologies (by analogy with molecular sequence alignment) [3]. Thus, the alignment is the output of the matching process, which is very similar (in a conceptual sense) to partial matching in CBR retrieval and to structure-mapping in analogical mechanisms. Notice also that ontology alignment is different from ontology merging: ontology merging takes as input two (or more) source ontologies and returns a merged ontology based on the given source ontologies.

There are two families of approaches to ontology alignment, commonly called syntactic and semantic approaches. Syntactic approaches establish matchings among predicates, terms or other structural properties of a formalism, essentially focusing on a notion of similarity. Semantic approaches establish logical equivalence correspondences among ontology terms, essentially focusing on a notion of semantic equivalence —in the logical sense of "semantic". We propose a third approach, a semiotic viewpoint that takes into account both the extensional and intensional definitions of a concept. Related to our approach are methods that work on "populated ontologies," i.e. ontologies that also contain instances of their concepts. Some approaches use instances to compute similarities among them in order to help them determine which concepts match. Although this is related to CBR, this is not the path taken here.

Another related approach is [8, 7], where a combination of Formal Concept Analysis (FCA) with Information Flow models for modeling and sharing common semantics is proposed. Their use of FCA is interestingly related with the approach taken here by the case-based semiotics for representing concepts. FCA has a two-layer representation of concepts, as we have in CBS with the intensional level and the extensional, that in FCA are called the intent and extent respectively of the concept. FCA, however, works only on attribute-value representations of instances and the intensional representations are subsets of attribute-value pairs, while our approach is more general, only requiring a representation formalism that has the subsumption operation. Similarly, FCA-merge [10] uses FCA over a common set of shared instances to merge two ontologies expressed as FCA lattices. Finally, "mutual online ontology alignment" [11] uses clustering and interchange of cases, but only uses the extensional description of concepts, while [1] proposes a similarity that takes into account the three dimensions of the semiotic triangle (see Fig. 1).

3 Adapting Taxonomies

A well known tenet in CBR is that after partial matching (i.e. retrieval), we need to adapt what's matched (because it is only *partially* matched) in order to reuse it for some purpose. Thus, while ontology alignment/matching is related to CBR retrieval and to structure-mapping in analogical mechanisms, the partiality of this process requires a second process: adaptation for reuse. This is the focus of this paper: retrieve for reuse, and in particular mutual adaptation of ontologies for reaching a shared, participatory ontology (or more precisely a fragment of an ontology).

In particular, we envision a context-dependent mutual adaptation of two ontologies held by two agents. These two agents aim at performing a particular task or goal, which defines a context in which (part of) their taxonomies need to be mutually intelligible. This does not mean an agent has to modify forever its ontology, only create a modified version for working within a particular context. Our approach can be summarized in the following schema:

$$O_1 \hookrightarrow T_1 \xrightarrow{adapt} T_1' \rightleftarrows T_2' \xleftarrow{adapt} T_2 \hookleftarrow O_2$$

where two agents, with ontologies O_1 and O_2, in order to perform some task, select ($\hookrightarrow$) a segment of their ontologies (T_1 and T_2) as relevant to the task, and

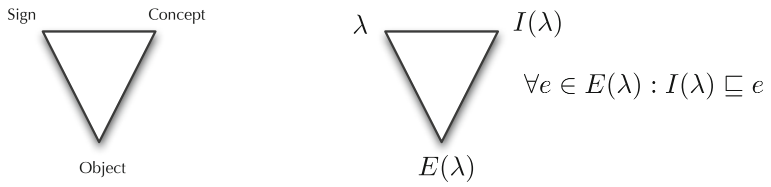


Fig. 1. The classic semiotic triangle on the left and a CBS concept C as a triplet $\langle \lambda, I, E \rangle$ on the right

then need to create two adapted versions of these segments ($T_1 \xrightarrow{adapt} T_1'$), such that are mutually intelligible ($T_1' \rightleftarrows T_2'$). In this view, the agents do not renounce or change their core ontologies, but they are capable of adapting (segments of) them to a particular context. In this paper we will focus on the adaptation process, assuming the agents are capable of previously agreeing on the context (i.e. the goal to achieve and the part of the ontology that is relevant).

We will propose a transformational adaptation approach to achieve mutually intelligible ontologies, but this approach has some limitations. Specifically, we will encompass only hierarchical ontologies (henceforth taxonomies). Moreover, while denotational semantics are commonly used in logic-based ontologies, we will propose a case-based approach to defining meaning and mutual intelligibility of concepts. This approach, based on the semiotics approach to meaning, takes into account not only the "abstract" definition of a concept but also the "experiences" with concrete episodes where this concept is used. The next subsection presents this case-based semiotics (CBS) approach to meaning and mutual intelligibility of concepts.

3.1 CBS Taxonomies

Our representation of hierarchical ontologies (taxonomies for short) is based on the semiotic triangle for concepts. We will define a CBS concept for a language $\mathcal{L}$ that possesses a subsumption relation ($\sqsubseteq$) among $\mathcal{L}$'s formulas. The language describing cases, without loss of generality, will be sublanguage $\mathcal{L}_c \subseteq \mathcal{L}$.

Definition 1. *(CBS Concept) A CBS concept C is a triplet $\langle \lambda, I, E \rangle$, given a signature $\langle \mathcal{L}, \sqsubseteq \rangle$ and a set of labels Λ, where:*

1. *$\lambda \in \Lambda$; where λ is a label (the* name *for the concept) from the set of labels Λ,*
2. *$I \in \mathcal{L}$ (I is a formula in a the language $\mathcal{L}$; where I is called the* intensional *definition of λ, also noted as $I(C)$,*
3. *$E = \{e_1, \ldots, e_n\}$ is a non-empty set of cases such that $\forall e_i \in E : e_i \in \mathcal{L}_c$; where the set E is called the* extensional *definition of λ, also noted as $E(C)$, and*
4. *$\forall e_i \in E : I \sqsubseteq e_i$*

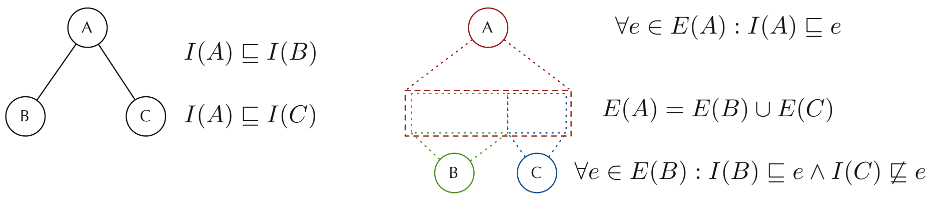

A
B
C
$I(A) \sqsubseteq I(B)$
$I(A) \sqsubseteq I(C)$
A
B
C
$\forall e \in E(A) : I(A) \sqsubseteq e$
$E(A) = E(B) \cup E(C)$
$\forall e \in E(B) : I(B) \sqsubseteq e \wedge I(C) \not\sqsubseteq e$

Fig. 2. For a taxonomy with , A and two children B and C, the intensional relations are shown at the left while the extensional relations are shown at the right

That is to say, a concept in CBS has a name, an intensional definition (that is a formula in some language), and a set of cases belonging to that concept (the extensional definition of that concept). For simplicity, we will sometimes denote a concept triplet by a symbol $C = \langle \lambda, I, E \rangle$ and we will use $I(C)$ and $E(C)$ to denote its intensional (I) and extensional definitions (E).

However, for an ontology we will need a discriminant definition; for this purpose we will use the notion of contrast set. We will say that a concept C is defined over a set of cases E whenever $E(C) \subseteq E$.

Definition 2. *(Contrast Set) Given a set of cases $E = \{e_1, \ldots, e_n\}$ we say a set of concepts $(C_1, \ldots, C_m)$ defined over E is a contrast set whenever $\forall i = 1, \ldots, m$:*

$$\left(\forall e_j \in E(C_i) : I(C_i) \sqsubseteq e_j\right) \wedge \left(\forall e_k \in \bigcup_{j=1,\ldots,m, j \neq i} E(C_j) : I(C_i) \not\sqsubseteq e_k\right)$$

That is to say, a case in E belongs (is subsumed by) at most one concept in the contrast set $(C_1, \ldots, C_m)$. However, not all cases need be members of a concept, which requires the contrast set to be a partition.

Definition 3. *(Conceptual Partition) A contrast set $(C_1, \ldots, C_m)$ defined over a set of cases E is a conceptual partition $\Pi((C_1, \ldots, C_m), E)$ iff $\forall e_i \in E, \exists C_j : I(C_j) \sqsubseteq e_i$.*

That is to say, a conceptual partition of a set of cases is an exhaustive classification of the set of cases where all cases belong to only one of the concepts.

We turn now to define a CBS hierarchical ontology (or CBS taxonomy for short). The taxonomy of concepts can be seen as a tree where nodes are CBS concepts. More formally, a taxonomy is an *arborescence*, i.e. a directed graph in which, for a vertex x called the *root* and any other vertex y, there is exactly one directed path from x to y. We will denote an arborescence by $\langle \mathbf{C}, \mathbf{A} \rangle$ where $\mathbf{C}$ is a set of nodes (or vertices) and $\mathbf{A}$ is set of arcs; for a $C \in \mathbf{C}$, we will denote the children of C as $A(C)$.

Definition 4. *(CBS Taxonomy) Given a collection of concepts $\mathbf{C} = \{C_1, \ldots, C_m\}$ defined over a set of cases $E = \{e_1, \ldots, e_n\}$, and an arborescence $\langle \mathbf{C}, \mathbf{A} \rangle$ with root C_1, the triple $\langle \mathbf{C}, \mathbf{A}, E \rangle$ is a CBS Taxonomy whenever:*

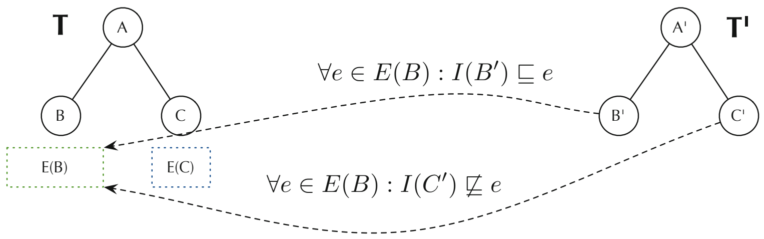

T
A
B
C
E(B)
E(C)
T'
A'
B'
C'
$\forall e \in E(B) : I(B') \sqsubseteq e$
$\forall e \in E(B) : I(C') \not\sqsubseteq e$

Fig. 3. Example of a concept B' in taxonomy T' converging with respect to the concept B in taxonomy T

1. $E(C_1) = E$ *and* $\forall e_j \in E : I(C_1) \sqsubseteq e_j$ *(root is sound and complete w.r.t.* E*)*
2. $\forall C_i \in \mathbf{C}' : \Pi(A(C_i), E(C_i))$ *is a conceptual partition,*
3. $\forall C_i \in \mathbf{C}' \wedge \forall C_j \in A(C_i) : I(C_i) \sqsubset I(C_j)$ *(intensional subsumption)*

where $\mathbf{C}' \subset \mathbf{C}$ *is the set of non-terminal concepts in the taxonomy.*

Fig. 2 shows some of these properties in a small example of taxonomy with root A and two children B and C. The subsumption relation is established among intensional descriptions of concepts, while extensional descriptions are related by set inclusion. Moreover, the two concepts at the same level, B and C, form a partition upon the cases in the extension of its father A.

Two concepts *labels* are *mutually intelligible* (a.k.a. aligned) when their CBS concepts converge. Conceptual convergence is defined as follows.

Definition 5. *(CBS Concept Convergence) Two CBS concepts* C_i *and* C_j *belonging to conceptual partitions* $\Pi(\mathbf{C}_i, E_i)$ *and* $\Pi(\mathbf{C}_j, E_j)$*in taxonomies* T_i *and* T_j *respectively, and with* $C_i^{\blacktriangle}$ *and* $C_j^{\blacktriangle}$ *the parents of* $\mathbf{C}_i$ *and* $\mathbf{C}_j$ *converge with respect to taxonomy* T_i *whenever:*

1. $\forall e \in E(C_i) : I(C_j) \sqsubseteq e$
2. $\forall e \in E(C_i), K \in \mathbf{C}_j - \{C_j\} : I(K) \not\sqsubseteq e$
3. $\forall e \in E(C_i) : I(C_j^{\blacktriangle}) \sqsubseteq e$

When the dual properties of 1 to 3 are satisfied, C_i *and* C_j *converge with respect to* T_j*. When* C_i *and* C_j *converge w.r.t. both* T_i *and* T_j*, we say* C_i *and* C_j *are conceptually convergent, noted as* $(C_i \cong C_j)$*. Moreover, we say their labels are mutually intelligible* $(\lambda_i \leftrightarrow \lambda_j)$ *for* T_i *and* T_j*.*

Property 1 states that C_j is consistent with C_i's extensional description, Property 2 that partition $\Pi(\mathbf{C}_j, E_j)$ is consistent with C_i's extensional description, and Property 3 that C_j's parent is consistent with C_i's extensional description.

Thus, convergence of two concepts occurs when both concepts converge with respect to the *other* taxonomy. Figure 3 shows an example of a concept B' in taxonomy T' converging with respect to the taxonomy T. Intuitively, the example in Fig. 3 means that $B' \sqsubseteq B$, since B' covers the cases in $E(B)$ and none of the cases in the remaining concepts of the partition under A. Thus, if two agents

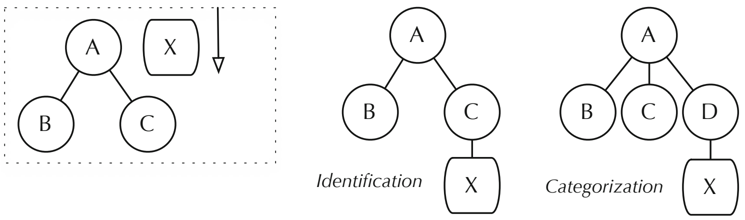


Fig. 4. Two adaptation operations over Hierarchical Ontologies: *Identification* (a case is identified as belonging to concept C) and *Categorization* (a case is identified as belonging to a new, previously non-existent, concept D)

Ag and Ag' are using taxonomies T and T' respectively, we say concept B is *intelligible for* agent Ag' — in the sense that there will be no misunderstanding or disagreement for agent Ag' with respect to concept B of agent Ag. Definition 5 states that two concepts converge w.r.t. both T and T' we have both $B' \sqsubseteq B$ and $B \sqsubseteq B'$, and thus they are equivalent ($B \cong B'$) w.r.t. to CBS. Consequently, when two agents communicate with each other using their labels ($\lambda \leftrightarrow \lambda'$) for the "same concept," their usage will be mutually intelligible.

Notice however, that the equivalence ($C_i \cong C_j$) in Definition 5 does not mean they are logically equivalent; what is assured is that they are equivalent w.r.t. the known set of known cases relevant to C_i and C_j, namely $E(C_i^{\blacktriangle}) \cup E(C_i^{\blacktriangle})$ (the set of observed cases in the contrasts sets to which C_i and C_j belong). Indeed, previously unseen cases can be identified or not as belonging to (C_i or C_j), leading to a disagreement that would require *adapting* again their taxonomies.

Thus, ontology matching and convergence is an evolving process according to case-based semiotics. Any agreement on the meaning of a sign or label is first participatory (applying to the involved agents) and contextual (depending on the finite knowledge of the world of the agents expressed as the set of cases grounding the concept's meaning). Finally, notice that our form of concept *alignment* is that of concepts being mutually intelligible w.r.t. to CBS. Thus, the alignment of two ontologies is to be defined as convergence of their concepts.

Definition 6. *(CBS Taxonomy Convergence) Two taxonomies* $\langle \mathbf{C}, \mathbf{A}, E \rangle$ *and* $\langle \mathbf{C}', \mathbf{A}', E' \rangle$ *with roots* A *and* A' *are CBS-convergent whenever* $\forall C \in \mathbf{C}, \exists C' \in \mathbf{C}'$ *such that* $C \cong C'$ *and* $\forall C' \in \mathbf{C}', \exists C \in \mathbf{C}$ *such that* $C \cong C'$.

3.2 Adaptation Operators

We will define several operations of transformational adaptation over the space of hierarchical ontologies. These operations are Identification, Categorization, Split, and Merge, and are shown in Figures 4, 5 and 6. These operators are similar to (and inspired from) the ones on the CobWeb unsupervised learning system [4], the main difference being derived from our distinction between cases (at the extensional level) and concepts (at the intensional level), which is nonexistent in CobWeb. Figure 4 shows on the left a new case X that is already classified

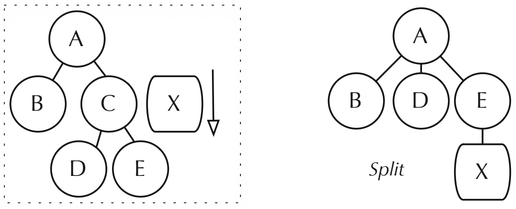


Fig. 5. The adaptation operation *Split*: concept C is "split" into its subconcepts that are promoted to the higher level, while the case X is later identified to one of the promoted concepts

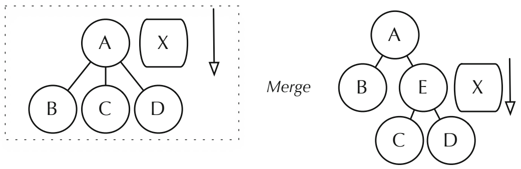


Fig. 6. The adaptation operation *Merge*: concepts C and D are "merged" into a higher level super-concept E and they are demoted to the lower level, while the case X is yet to be processed below the new concept E

as being a member of concept A; in other words $I(A) \sqsubseteq X$ (the intensional definition of A subsumes X). Applying the operator Identification we obtain the tree shown in the middle of Fig. 4. This operator characterizes the situation where a new case is identified as member of a concept (e.g. the concept C) with no further change required except maybe to generalize to insure $I(C) \sqsubseteq X$. However, consider the case where generalizing $I(C)$ to include X would mean that $I(C)$ also subsumes cases under B; this means that X cannot be identified as member of C. If this is also the case for B then X cannot be identified as member of B or C and (as shown at the right of Fig. 4) we need a new concept, let's call it D, that encompasses X. This situation is characterized by the operator Categorize, that creates a "new category" for a case X. Thus, the result of operator Categorize is moving from a partition (B, C) of the extension of A to a partition (B, C, D).

4 Mutual Adaptation of Taxonomies

Two agents communicate and deliberate about the meaning of their taxonomies, or more specifically, about a fragment of their taxonomies starting from a common root. If the root under discussion is the taxonomies root then the agents will deliberate about the meaning of all concepts in their taxonomies. For this purpose, agents need to recognize situations where there is no agreement and then apply some adaptation operators. This approach is similar to goal-driven learning (GDL) [5], Goal-driven learning decomposes the learning problem in three steps: blame assignment, learning goal generation, and repair (or learning) strategy. GDL considers a single agent reasoning introspectively about detecting its own failures (blame assignment), deciding what needs to be learnt to correct it (learning goal generation), and determining a way to achieve this goal (repair strategy).

4.1 Non-structural Adaptations

In non-structural adaptations, disagreements involve mismatches between intensional and extensional definitions that do not require transforming the *is-a* relationship between concepts (as can be seen in [6]).

Generalization. This situation is characterized as follows: agent Ag_1 has a case X subsumed by concept B, while agent Ag_2 has a concept B' that subsumes most cases in B but not X. Moreover, the other concepts in the partition K' where B' is located in T' do not subsume X either. Thus, the partition K' does not account for X and since Ag_1 knows it should be covered by B', Ag_2 should change the definition of B'. Therefore Ag_1 sends the argument "your concept B' should also cover X" to Ag_2. Then, Ag_2 generalizes $I(B')$ to cover X while not covering any case subsumed by the other concepts in partition K'.

Specialization. This situation is characterized as follows: agent Ag_1 has a case X subsumed by concept B, while agent Ag_2 has a concept C' different from B' that does subsume X. Since Ag_1 current hypothesis is that B and B' should converge while B and C' should not, Ag_2 should change the definition of C'. Therefore Ag_1 sends the argument "your concept C' should not cover X, which should be covered by B'" to Ag_2. Consequently, Ag_2 has to specialize concept's intension $I(C')$ so that C' it does no longer cover X. Additionally, B' may or may not cover X. If not, Ag_2 generalizes $I(B')$ to cover X while not covering any case subsumed by the other concepts in partition K'.

4.2 Structural Adaptations

Structural adaptations are triggered by mismatches in the way the cases are sorted by partitions, and require the transformation of partitions; that is to say transforming the tree of is-a relationships among concepts, including the creation of new concepts. Let us start with the second situation in Fig. 4: categorization. Let us assume, e.g., an agent Ag_2 sends case X to agent Ag_1 and eventually Ag_1's ontology is adapted by including a new concept D to encompass case X.

The scenario requires some initial conditions, as follows. Assume the agents have already achieved concept convergence over the root A; therefore both agree that X (sent from Ag_2 to Ag_1) can be identified as belonging to A. However, they do not agree under which concept, in the partition set under A, should case X be allocated. In order to move from the left part of Fig. 4 to the right part, the agents have to agree on the following: a) X is not under B (or B' for the second agent); and X is not under C (or C') either. Thus, since X should be under A but is not part of B or C, a new concept is needed for Ag_1; since Ag_2 already

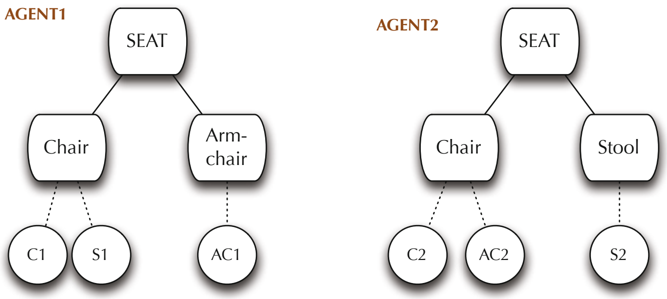


Fig. 7. The initial state of two agents taxonomies in the *Seat* domain

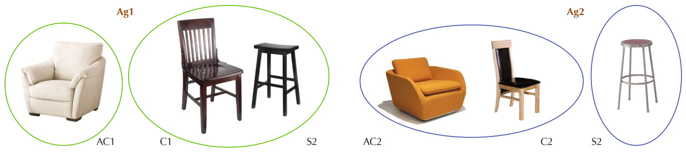


Fig. 8. The two taxonomies in the *Seat* domain

has identified X under a concept D' (with label λ'_D), agent Ag_1 will create this new concept D (with label λ_D), and X will be situated under D. Since they are mutually intelligible ($\lambda_D \leftrightarrow \lambda'_D$) the adaptation process ends there.

To explain the Split adaptation operator we will introduce an example shown in Fig. 7. The Seat domain is very simple but will illustrate the kind of mismatches that can be found and resolved by mutual adaptation. Agent Ag_1 in Fig. 7 knows two kind of seats: chairs and armchairs, while agent Ag_2 knows two kind of seats: chairs and stools. Ag_1 divides seats depending on whether they have arms or not, while Ag_2 divides seats depending on whether they have backs or not, as shown in Fig. 8[2]. Notice that both agents have cases that are stools ($S1$ and $S2$), chairs ($C1$ and $C2$) and armchairs ($AC1$ and $AC2$); they just choose to conceptualize them differently.

We can easily imagine a convergence of both ontologies into one shared by both agents and that has the three involved concepts: stools, chairs and armchairs. Indeed, we will follow the deliberation and adaptations that achieve that, but notice that the particular solution achieved is not unique. As we will show, the agents reach an ontology with *Seat* as root and three children (*Stool*, *Chair* and *Armchair*); nevertheless, other ontology structures are possible and correct results, for instance an ontology where *Seat* is the root with children *Stool* and *Chair*, in which *Chair* has two children concepts: *Armchair* and *SimpleChair* (i.e. a seat with back and no arms).

[2] The second ontology is the one found in the Wikipedia (armchair is a subtype of chair) while some of the authors claim they feel more intuitive the first one is more intuitive, and to classify an armchair not as a chair and see it as a kind couch.

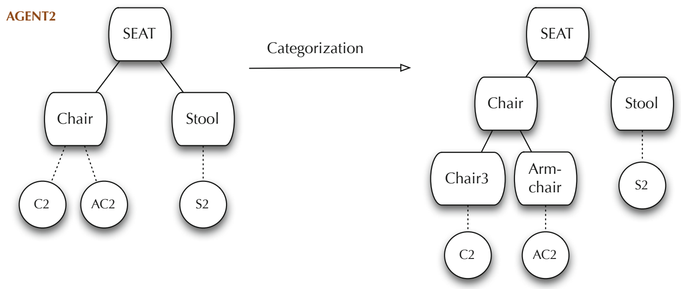


Fig. 9. Adaptation of the taxonomy of agent $Ag2$ by adding concept *Armchair*

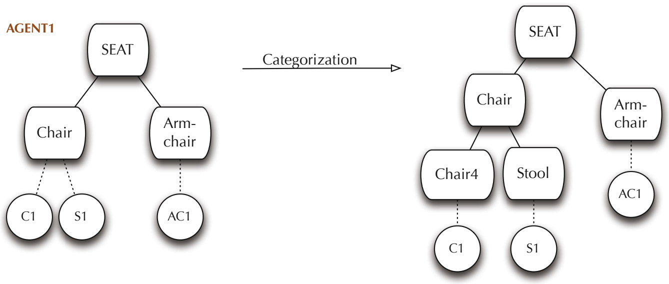


Fig. 10. Adaptation of the taxonomy of agent $Ag1$ by adding concept *Stool*

Agent Ag_1 in Fig. 7 has received the intensional definition of concepts $Chair_2$ and $Stool_2$ from Ag_2, and agent Ag_2 the intensional description of Ag_1's concepts. Considering first Ag_1, the agent has found the following disagreements:

1) $Stool_2$ covers case $S1$ that is covered by the intensional definition of concept $Chair_1$; thus a chair like $S1$ is a stool of Ag_2, a concept not existing in Ab_1
2) $Chair_2$ covers case $AC1$ that , according to Ag_1, is not a chair but is under concept $Armchair_1$, a concept that is not present in Ag_2's taxonomy.

In order to proceed, Ag_1 asks Ag_2 to create and include the concept *Armchair* in its taxonomy. Ag_2 accepts, which implies the following:

1) a new concept using the intensional definition of $Armchair_1$ has to be created, and call it $Armchair_2$ (thus $I(Armchair_2) := I(Armchair_1)$)

2) Ag_2 determines that $Armchair_2$ covers case $AC2$ but not case $C2$, thus the adaptation operation Categorization can be applied to concept $Chair_2$ creating $Armchair_2$ as a subconcept of $Chair_2$,

3) however, now the children of $Chair$ (case $C2$ and $Armchair_2$) do not form a partition (since case $C2$ is not a concept). Thus a new concept $Chair_3$ is created to cover case $C2$; as shown in Fig. 9 the children of $Chair$ now form a partition.

Notice that in the example we show only one case per concept just for brevity's sake. In general, adding $Armchair_2$ under $Chair_2$ would mean that all cases subsumed by (the intensional definition of) $Chair_2$ that are also subsumed by

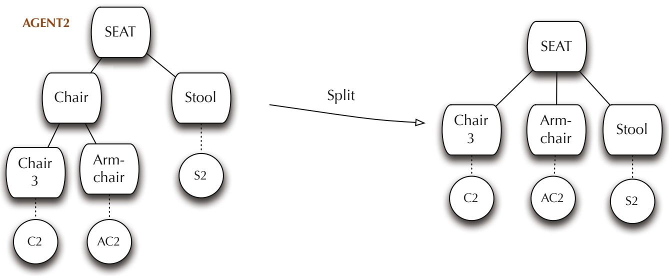


Fig. 11. Adaptation of the taxonomy of agent *Ag*2 by splitting the old concept *Chair* and promoting concept *Chair*$_3$ and *Armchair* as subconcepts of *Seat*

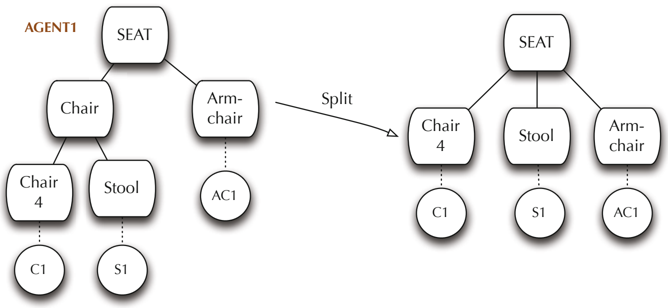


Fig. 12. Adaptation of the taxonomy of agent *Ag*2 by splitting the old concept *Chair* and promoting concept *Chair*$_3$ and *Armchair* as subconcepts of *Seat*

(the intensional definition of) $Armchair_2$ become the extensional definition of $Armchair_2$, i.e. $E(Armchair_2) = \{c \in E(Chair_2) | I(Armchair_2) \sqsubseteq c\}$, while the rest become the extensional definition for a new concept to complete the partition: $E(Chair_3) = E(Chair_1) - E(Armchair_2)$. Finally, the intensional definition $I(Chair_3)$ of the new concept is inferred by induction over the cases of the extensional definition $E(Chair_3)$.

A similar process is carried out when agent Ag_2 asks Ag_1 to include the *Stool* concept, as shown in Fig. 10. Now both agents have incorporated a new concept coming from the other agent refining their respective ontologies. However, as can be observed comparing Fig. 9 and Fig. 10 their ontologies do not match: although there lower level concepts converge (since *Chair*, *Stool* and *Armchair* partition the extensional definition of the overall concept *Seat* the same way), the intermediate concepts (the "old" concepts of *Chair* in both agents) do not converge. This disagreement can be resolved applying adaptation operator split (Fig. 5) to the "old" concepts of *Chair* in both agents. Figures 11 and 12 show that the same result is obtained by both agents using the split operation.

Finally, the Merge adaptation operation works in a similar way to Split. Recalling Fig. 6, we see Merge would be applied when one agent has an intermediate concept that the other has not. We will not develop the example in full, but it is easy to see how merge can be used in the example of Figure 13. Given the state of agents Ag_1 and Ag_2 in Fig. 13, when Ag_2 applies Merge to concepts *Chair*

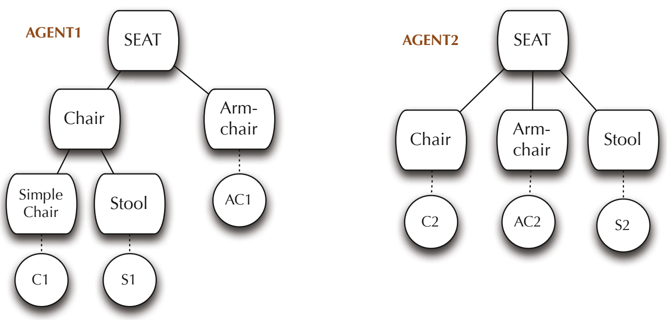


Fig. 13. An state where the Merge operator would make two taxonomies convergent

and *Armchair* creating a new superconcept *NewChair* the two taxonomies converge. Specifically, they have the following alignments: ($Chair \leftrightarrow NewChair$), ($SimpleChair \leftrightarrow Chair$), ($Stool \leftrightarrow Stool$), ($Armchair \leftrightarrow Armchair$). Clearly, this is not the only configuration that leads to a convergence. A second, equivalent solution is that agent Ag_1 applies Split to *Chair* (promoting *SimpleChair* and *Stool* to the level of *Armchair*) thus reaching a taxonomy convergent with that of Ag_2. Both solutions are equally adequate from the point of view of CBS.

5 Mutual Adaptation as Search

The CBS approach allows us to characterize (1) disagreements in the intended meaning of concepts in two taxonomies and (2) the transformations upon ontologies performed by adaptation operations. Thus, mutual adaptation of ontologies is viewed as a search process over the space of possible taxonomies under case-based semiotics. We say that two concepts from T and T' are in coincidence when, although they do not converge, they both subsume a subset of the cases subsumed by the other.

Definition 7. *(CBS Coincident Concepts) Two CBS concepts C_i and C_j in taxonomies T and T' respectively, and with parents $C_i^{\blacktriangle}$ and $C_j^{\blacktriangle}$ such that $C_i^{\blacktriangle} \cong C_j^{\blacktriangle}$ are in coincidence ($C_i \rightleftharpoons C_j$) whenever $\exists K_i \subseteq E(C_i), K_j \subseteq E(C_j)$ (with $K_i \neq \emptyset$ and $K_j \neq \emptyset$) such that $I(C_i) \sqsubseteq K_j$ and $I(C_j) \sqsubseteq K_i$.*

Two concepts that are in coincidence are basically candidates for converging if the current disagreement or mismatches are solved by adaptation operators. The search process maintains a list of coincident concept pairs and constitute the candidates to which adaptation operators can be applied.

Figure 14 and Figure 15 show some examples of the CBS typology of disagreements for taxonomies to which some adaptation operators may be applied. For instance, Fig. 14 shows on the left the situation where a case X' of taxonomy T' is covered by concept A in T ($I(A) \sqsubseteq X'$) bot none of the concepts in the conceptual partition (B, C) cover X' (i.e. $I(B) \not\sqsubseteq X'$ and $I(C) \not\sqsubseteq X'$). This may be due to two different situations depending on X' in taxonomy T', shown

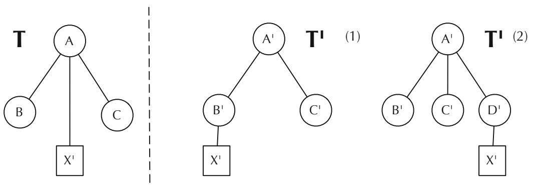


Fig. 14. A type of concept disagreement in which a case X' of T' is not covered by a conceptual partition in T

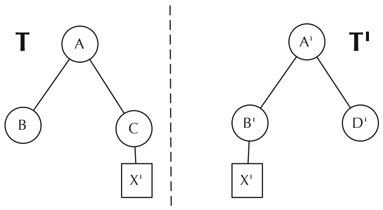


Fig. 15. A type of concept disagreement in which a case X' of T' is covered by a different concept in T

to the right of Fig. 14: either (1) X' is covered by a concept, say B' and thus $B \ncong B'$, or (2) X' is covered by concept that does not exist in T, say D' in T'. To solve this disagreement and achieve convergence, in situation (1) the intensional definition is changed using the Generalization adaptation operation on B, while in situation (2) the Categorization adaptation operation is used to include a new concept D in the conceptual partition. Another instance of disagreement is shown in Fig. 15, where a case X' that belongs to concept B' in T' is however covered in taxonomy T by a concept that is not the coincident concept B.

6 Discussion

We have presented a general framework for addressing the problem of semantic intelligibility among artificial agents based on concepts integral to the case-based reasoning research program. Mutual intelligibility of concepts should be grounded, in our approach, to collections of cases (i.e. descriptions of objects or situations). Using a semiotic viewpoint instead of a classical logic semantics allows us to work with cases in a principled way, that we have formalized as CBS (case-based semiotics), in which a concept has a label and two (mutually dependent) levels of description: the intensional level and the extensional level.

Mutual intelligibility of concepts is moreover modeled as a process of mutual adaptation, in which artificial agents modify their knowledge structures to reach a convergent model (in the CBS framework) of the concepts they need to share. This mutual adaptation process is viewed as a search process performed by adaptation operators, as is classically obtained by transformational adaptation in CBR. However, the situation is here more complex than in classical transformational adaptation, since there are two agents involved. A particular interaction protocol to implementing search in the space of possible taxonomies remains future work, although the adaptation operators that define the search space have already been defined here.

Finally, the CBS framework allows the acquisition of new concepts in a natural way. A new concept implies either the reorganization of the partition of the cases known to an agent or the acquisition of a new, unknown case. In the CBS approach, learning from cases and adapting the knowledge structure commonly called "ontology" are seamlessly integrated in the same process. As part of the future work we intend to show that two agents using our adaptation operators can always converge on a shared taxonomy, even when they have concepts and cases unknown to one another.

Acknowledgments. This research was partially supported by projects Next-CBR (TIN2009-13692-C03-01) and Agreement Technologies (CONSOLIDER CSD2007-0022).

References

[1] Aimé, X., Furst, F., Kuntz, P., Trichet, F.: SemioSem: A Semiotic-Based Similarity Measure. In: Meersman, R., Herrero, P., Dillon, T. (eds.) OTM 2009 Workshops. LNCS, vol. 5872, pp. 584–593. Springer, Heidelberg (2009)
[2] Eco, U.: I limiti dell'interpretazione. Bompiani (1990)
[3] Euzenat, J., Shvaiko, P.: Ontology Matching. Springer (2007)
[4] Fisher, D.H.: Knowledge acquisition via incremental conceptual clustering. Machine Learning 2(2), 139–172 (1987)
[5] Leake, D.B., Ram, A. (eds.): Goal-Driven Learning. MIT Press (1995)
[6] Ontañón, S., Plaza, E.: Concept convergence in empirical domains. In: Pfahringer, B., Holmes, G., Hoffmann, A. (eds.) DS 2010. LNCS (LNAI), vol. 6332, pp. 281–295. Springer, Heidelberg (2010)
[7] Schorlemmer, M., Kalfoglou, Y.: Progressive ontology alignment for meaning coordination: An information-theoretic foundation. In: Proc. AAMAS 2005, pp. 737–744. ACM Press (2005)
[8] Schorlemmer, M., Kalfoglou, Y., Atencia, M.: A formal foundation for ontology-alignment interaction models. International Journal on Semantic Web and Information Systems 3(2), 50–68 (2007)
[9] Steels, L.: Why We Need Evolutionary Semantics. In: Bach, J., Edelkamp, S. (eds.) KI 2011. LNCS, vol. 7006, pp. 14–25. Springer, Heidelberg (2011)
[10] Stumme, G., Maedche, A.: FCA-MERGE: Bottom-up merging of ontologies. In: Proc. IJCAI 2001, pp. 225–230. Morgan Kaufmann Publishers Inc. (2001)
[11] Wang, J., Gasser, L.: Mutual online ontology alignment. In: Proc. AAMAS 2002 Workshop on Ontologies in Agent Systems, OAS 2002. CEUR Workshop Proceedings, vol. 66 (1990)

A Lazy Learning Approach to Explaining Case-Based Reasoning Solutions

David McSherry

School of Computing and Information Engineering
University of Ulster, Coleraine BT52 1SA, Northern Ireland
{dmg.mcsherry}@ulster.ac.uk

Abstract. We present an approach to explanation in case-based reasoning (CBR) based on demand-driven (or lazy) discovery of explanation rules for CBR solutions. The explanation rules discovered in our approach resemble the classification rules traditionally targeted by rule learning algorithms, and the learning process is adapted from one such algorithm (PRISM). The explanation rule learned for a CBR solution is required to cover both the target problem and the most similar case, and is used together with the most similar case to explain the solution, thus integrating two approaches to explanation traditionally associated with different reasoning modalities. We also show how the approach can be generalized to enable the discovery of explanation rules for CBR solutions based on *k*-NN. Evaluation of the approach on a variety of classification tasks demonstrates its ability to provide easily understandable explanations by exploiting the generalizing power of rule learning, while maintaining the benefits of CBR as the problem-solving method.

Keywords: case-based reasoning, lazy learning, explanation, confidence.

1 Introduction

Explanation has been a topic of ongoing research interest in case-based reasoning (CBR) during the past decade (e.g., [1–12]), with several contributions related to explanation of CBR solutions in the context of classification problem solving. For example, Cunningham *et al.* [1] compare the traditional approach to explaining CBR solutions (i.e., showing the user the most similar case) with the traditional approach to explanation in rule-based reasoning (i.e., showing the user the rule on which the solution is based). They report that users found CBR explanations more convincing than rule-based explanations in a binary classification task for which solutions and explanations based on both reasoning modalities were generated. As noted by the authors, however, user evaluations are difficult to perform across a range of datasets, and the relatively simple representation of cases (5 attributes) in their experiment may have favored the CBR approach.

Moreover, McSherry [8] argues that the most similar case may be of limited value for explaining a CBR solution given that some of the features that it shares with the target problem, thus contributing to its similarity, may in fact provide evidence against the solution. The explanatory value of the most similar case is also questioned by Doyle *et al.* [2], who show that a case closer to the class boundary can sometimes provide a more convincing explanation for a CBR solution. In this paper, we present a novel approach to explaining CBR solutions based on *explanation* rules discovered by lazy learning from the available cases. The discovered explanation rule for a CBR solution resembles a typical classification rule [13] and is used together with the most similar case to explain (or justify) the solution. In this way, our approach integrates two approaches to explanation traditionally associated with different reasoning modalities.

In Section 2, we describe the basic concepts in our approach to explanation in CBR, using an example case base to illustrate the discussion. In Sections 3 and 4, we present our approach to demand-driven discovery of explanation rules for CBR solutions based on nearest neighbor retrieval (1-NN) and show how it can be generalized to enable the discovery of explanation rules for CBR solutions based on *k*-NN. In Section 5, we present empirical results that demonstrate the ability of our approach to provide easily understandable explanations for CBR solutions in a variety of classification tasks. We also investigate the hypothesis that the coverage of an explanation rule (i.e., the number of cases that satisfy the rule's conditions) can be useful as an indicator of the *confidence* associated with the CBR solution it is used to explain. Our conclusions are presented in Section 6.

2 Basic Concepts

In this section, we describe the role of explanation rules in our approach to explanation in CBR, and the example case base that we use to illustrate the approach. The similarity measures used in the examples and experiments discussed in the paper are also briefly described.

Example Case Base. Table 1 shows an artificial case base in the travel methods domain that we use to illustrate our approach to explanation in CBR. The cases describe journeys to various destinations in an urban area and the travel method (walk, taxi, or drive) used for each journey. Distance to the destination is measured in minutes of walking time. The class attribute is the travel method and the majority class (41%) is *walk*. Choosing a suitable travel method for a journey within a town or city should be a familiar task for most readers, thus making it easy to assess the quality of the explanations provided in our approach.

Explanation Rule. An explanation rule is very similar to a classification rule [13], but is not used for classification in our approach. Instead, it is used to explain (or justify) the solution provided by a CBR system. For example, if *taxi* is the solution provided by a CBR system for a given problem in the travel methods domain, the explanation rule used to explain the solution might be:

Rule 1. **if** car owner = no **and** distance $\geq$ 20 **then** taxi

Like a classification rule, an explanation rule has one or more conditions on the LHS (or may have no conditions) and a solution (or predicted class) on the RHS.

Table 1. Artificial case base in the travel methods domain used to illustrate the proposed approach to explanation in CBR

Case No.	Destination	Walking Distance (mins.)	Car Owner	Travel Method
1	supermarket	15	yes	drive
2	pub	45	no	taxi
3	supermarket	20	no	taxi
4	pub	10	no	walk
5	theatre	35	yes	drive
6	theatre	15	yes	walk
7	pub	40	yes	taxi
8	supermarket	50	yes	drive
9	pub	20	yes	walk
10	theatre	40	no	taxi
11	theatre	25	no	walk
12	supermarket	55	no	taxi
13	pub	45	yes	taxi
14	pub	15	no	walk
15	theatre	20	yes	walk
16	theatre	45	yes	drive
17	pub	5	yes	walk

Rule Accuracy and Coverage. An explanation rule *R* covers a given case *C* if all the conditions in *R* are satisfied by *C*. The *coverage* of an explanation rule is the number (or percentage) of cases that it covers. The *accuracy* of an explanation rule is the percentage of cases that it covers for which it gives the correct solution. For example, Rule 1 covers 5 cases in the example case base (i.e., Cases 2, 3, 10, 11, and 12) and correctly classifies 4 of them (i.e., Cases 2, 3, 10, and 12). Its accuracy on the case base is therefore:

$$\text{accuracy(Rule 1)} = \frac{4}{5} \times 100 = 80\% \tag{1}$$

A rule with no conditions such as **if** _ **then** *S* covers all cases in the case base and correctly classifies all cases with solution *S*. As described in Section 3, rule accuracy and coverage play important roles in our demand-driven (or lazy) approach to the discovery of explanation rules. When a discovered explanation rule is used to explain a CBR solution, its accuracy and coverage are also presented to the user e.g.,

if car owner = no **and** distance ≥ 20 **then** taxi (0.80, 5)

In Section 5, we investigate the hypothesis that the coverage of an explanation rule can be useful as an indicator of the *confidence* associated with the CBR solution it is used to explain.

Similarity Measures. The similarity measures used in the examples discussed in the paper, and experiments reported in Section 5, are briefly described below. However, our approach to explanation in CBR can be used with any method of similarly assessment. For any problem description *P* and case *C* we define:

$$Sim(P,C) = \frac{\sum_{a \in A} w_a \times sim_a(P,C)}{\sum_{a \in A} w_a} \tag{2}$$

where *A* is the set of case attributes, $sim_a(P,C)$ is a local measure of the similarity between the values of *a* in *P* and *C*, and w_a is the importance weight assigned to $a \in A$. In the examples and experiments discussed in the paper, the case attributes are equally weighted, though any weighting of the case attributes can be used in our approach to explaining CBR solutions. For any attribute whose value is missing in the problem description (*P*) or case (*C*) we assign a local similarity score of zero. For a nominal attribute *a* whose values in *P* and *C* are known we assign a local similarity score of zero for unequal values and one for equal values. For a continuous attribute *a* whose values in *P* and *C* are known we define:

$$sim_a(P,C) = 1 - \frac{|v_1 - v_2|}{\max(a) - \min(a)} \tag{3}$$

where v_1, v_2 are the values of *a* in *P* and *C*, and *min*(*a*), *max*(*a*) are the minimum and maximum values of *a* in the case base.

3 Explaining Nearest Neighbor Solutions

In this section, we describe our approach to learning explanation rules for CBR solutions based on nearest neighbor retrieval (1-NN) and use the travel methods case base (Table 1) to illustrate the approach.

3.1 Learning Explanation Rules

Our approach to learning explanation rules for CBR solutions is adapted from PRISM [14–17], an algorithm for learning "modular" rules directly from a given set of training examples as an alternative to generating rules from a decision tree. An important feature of PRISM is that it tends to produce rules that have fewer conditions, and are thus easier to understand, than rules generated from a decision tree [14,15]. In common with most rule learning algorithms, PRISM uses a *sequential covering* strategy [13] to learn a set of rules from the dataset. It learns rules for each class in turn, starting with the complete training set for each class. As each rule is learned, the examples covered by the rule are removed from the dataset and the process is repeated until all examples of the target class are covered by the rules that have been learned. The algorithm builds each rule by starting with an initial rule with one condition and specializing it incrementally using hill climbing until it reaches 100% accuracy on the training examples or cannot be further specialized.

In contrast to PRISM's *eager* learning approach, our approach to explanation rule learning is *lazy* (or demand driven) in that a single rule is learned from the currently available cases only when needed to explain a CBR solution. Thus there is no need for repeated application of the learning process as in PRISM's sequential covering strategy. The learning process is also highly constrained in our approach in that the discovered explanation rule is required to cover both the target problem and the most similar case. It must also correctly classify the most similar case.

The original version of PRISM [14] could only be applied to datasets with discrete attributes and with no missing values, although techniques for addressing these limitations have been implemented in later versions [15–17]. Our algorithm for explanation rule learning creates rule conditions based on continuous attributes in a similar manner to later versions of PRISM, and allows missing values in case or problem descriptions. The use of rule coverage to break ties between equally accurate rules, as in our approach, is also a feature of more recent versions of PRISM.

Our algorithm (Learn-ER) for learning an explanation rule to explain a CBR solution (*S*) based on nearest neighbor retrieval (1-NN) is shown in Fig. 1. Starting with an initial rule with no conditions and *S* as its conclusion, it repeatedly adds a single condition to the rule, provided the specialized rule is more accurate and covers both the problem description (*P*) and most similar case (*C*). At each stage, Learn-ER

Algorithm: Learn-ER

Inputs: problem description (*P*), most similar case (*C*), solution (*S*), attributes (*As*), case base (CB)

Output: An explanation rule (*R*) for *S*

```
begin
    $R \leftarrow$ if _ then $S$
    while $accuracy(R) < 100$ and $As \neq \varnothing$ do
    begin
        $Rs \leftarrow$ set of all rules that cover $P$ and $C$ and are created by adding a
        single condition $a = v$, $a \geq v$, or $a \leq v$ to $R$, over all $a \in As$ and values
        $v$ of $a$ in CB
        if $Rs = \varnothing$
        then return $R$
        select the most accurate rule $R^* \in Rs$ using rule coverage to break ties
        between equally accurate rules
        if $accuracy(R^*) \leq accuracy(R)$
        then return $R$
        $R \leftarrow R^*$
        $As \leftarrow As - \{a\}$
    end
    return $R$
end
```

Fig. 1. Algorithm for learning an explanation rule

selects the most accurate specialization of the current rule, using rule coverage to break ties between equally accurate rules. If any of the criteria for further specialization of the current rule cannot be satisfied (e.g., when its accuracy reaches 100%), then the learning process stops and the current rule is returned as the explanation rule.

It is worth noting some important features of the algorithm that help to ensure the efficiency of the rule learning process:

- A case attribute can appear at most once in a discovered explanation rule.
- A nominal attribute *a* can appear only in rule conditions of the form $a = v$, where v is the value of *a* in the target problem and also the value of *a* in the most similar case. This follows from the fact that the rule is required to cover both the target problem and the most similar case.
- A continuous attribute *a* can appear only in rule conditions of the form $a \geq v$ or $a \leq v$ that are satisfied by both the target problem and the most similar case. It must also be true that *v* is one of the values of *a* in the case base.

It is possible, though unlikely, that Learn-ER will fail to discover an explanation rule with one or more conditions, in which case it will return a rule with no conditions. For example, no explanation rule with one or more conditions can cover the target problem and most similar case if all the case attributes are nominal and the most similar case has different values from the target problem for all case attributes. However, as shown by our experimental results in Section 5, Learn-ER rarely fails to return a rule with one or more conditions that can be used to provide a meaningful explanation for a given CBR solution.

3.2 Explaining Travel Method Choices

As an example to illustrate our lazy learning approach to explaining CBR solutions, consider the following problem description in the travel methods domain:

P: destination = pub, distance = 35, car owner = yes

The most similar case in the travel methods case base (Table 1) is:

Case 7: destination = pub, distance = 40, car owner = yes, travel method = taxi

Its similarity to the problem description, from Eqns. 2 and 3, is:

$$Sim(\text{P, Case 7}) = \frac{1}{3} \times \left(1 + (1 - \frac{40 - 35}{55 - 5}) + 1\right) = 0.97$$

To discover an explanation rule for *taxi*, the nearest neighbor solution for P, Learn-ER first considers all immediate specializations of the initial rule **if** _ **then** taxi (i.e., rules with one condition) that cover both the target problem and the most similar case (Case 7). There are two such rules involving a nominal attribute:

1. **if** destination = pub **then** taxi (0.43, 7)
2. **if** car owner = yes **then** taxi (0.20, 10)

There are also four such rules that specify an upper limit for distance:

3. **if** distance ≤ 55 **then** taxi (0.35, 17)
4. **if** distance ≤ 50 **then** taxi (0.31, 16)
5. **if** distance ≤ 45 **then** taxi (0.33, 15)
6. **if** distance ≤ 40 **then** taxi (0.25, 12)

Finally, there are six such rules that specify a lower limit for distance:

7. **if** distance ≥ 5 **then** taxi (0.35, 17)
8. **if** distance ≥ 10 **then** taxi (0.38, 16)
9. **if** distance ≥ 15 **then** taxi (0.40, 15)
10. **if** distance ≥ 20 **then** taxi (0.50, 12)
11. **if** distance ≥ 25 **then** taxi (0.56, 9)
12. **if** distance ≥ 35 **then** taxi (0.63, 8)

For example, Rule 10 covers 12 cases and correctly classifies 6 of them (i.e., Cases 2, 3, 7, 10, 12, and 13) so its accuracy is 50%. However, the most accurate rule (63%) is Rule 12, which covers 8 cases. So the rule that Learn-ER selects for further specialization is Rule 12. As an attribute can appear in at most one rule condition, there are only two immediate specializations of Rule 12 that cover both the target problem and the most similar case and correctly classify the most similar case:

13. **if** distance ≥ 35 **and** destination = pub **then** taxi (1.00, 3)
14. **if** distance ≥ 35 **and** car owner = yes **then** taxi (0.40, 5)

Rule 13 is 100% accurate as it correctly classifies all cases that it covers (i.e., Cases 2, 7, and 13), while Rule 14 correctly classifies only 40% of cases that it covers. So Rule 13 is selected as the best specialization of Rule 12. As Rule 13 is 100% accurate, it is returned as the explanation rule for the CBR solution (*taxi*) without further specialization. So the explanation rule for *taxi* as the solution of the target problem (P: destination = pub, distance = 35, car owner = yes) is:

if distance ≥ 35 **and** destination = pub **then** taxi (1.00, 3)

The user is also shown the most similar case as part of the explanation for the CBR solution i.e.,

Case 7: destination = pub, distance = 40, car owner = yes, travel method = taxi

3.3 Example Explanation Rules

Table 2 shows problem descriptions, nearest neighbor solutions, and discovered explanation rules for several problems in the travel methods domain, including the example problem discussed in Section 3.2. The conditions in each rule are arranged to match the ordering of the case attributes in Table 1. Rules 3 and 4 in Table 2 are fairly realistic explanations for the travel methods they predict for going to the supermarket, although a person might use their background knowledge about shopping to explain that some form of transport (taxi or car) is needed to bring home the shopping. Similarly, Rules 1 and 2 are perhaps the most useful explanations for taking a taxi or walking to a pub that can be expected in the absence of background knowledge. For example, Rule 1 can be paraphrased as follows, which seems like good advice in the travel methods domain:

If you are going to the pub and it is too far to walk then take a taxi

Rules 5–7 can also be seen to provide realistic explanations for the travel methods they predict for going to the theatre.

Table 2. Example problem descriptions, CBR solutions, and discovered explanation rules in the travel methods domain

Rule No.	**Problem Description**			**CBR Solution**	**Explanation Rule**
	Destination	Walking Distance (mins.)	Car Owner		
1	pub	35	yes	taxi	**if** destination = pub **and** distance ≥ 35 **then** taxi
2	pub	20	no	walk	**if** destination = pub **and** distance ≤ 25 **then** walk
3	supermarket	10	yes	drive	**if** destination = supermarket **and** car owner = yes **then** drive
4	supermarket	15	no	taxi	**if** destination = supermarket **and** car owner = no **then** taxi
5	theatre	30	yes	drive	**if** destination = theatre **and** distance ≥ 25 **and** car owner = yes **then** drive
6	theatre	45	no	taxi	**if** distance ≥ 40 **and** car owner = no **then** taxi
7	theatre	25	yes	walk	**if** destination = theatre **and** distance ≤ 25 **then** walk

4 Explanation in *k*-NN

In this section, we generalize our approach to explaining nearest neighbor (1-NN) solutions to provide rule-based explanations for *k*-NN solutions. An important difference in *k*-NN is that the solution for the most similar case may be out-voted by another solution in the retrieval set (i.e., the *k* most similar cases). In this situation, showing the user the most similar case and/or an explanation rule that predicts its solution does not seem a useful way to explain the *k*-NN solution. As described in Section 4.1, the role of the most similar case in our lazy learning approach to explanation in CBR is replaced by a case we refer to as the nearest *supporting* neighbor. In Section 4.2, we use an example based on the well-known Iris dataset from the UCI Machine Learning Repository [18] to illustrate our approach to explaining *k*-NN solutions.

4.1 Nearest Supporting Neighbor

Typically in *k*-NN, the class assigned to a target problem is the *majority* class in the retrieval set (i.e., the class that occurs most frequently in the *k* most similar cases). Fig. 2 shows an example in which the class assigned to the target problem by 5-NN is *positive* even though the solution for the most similar case is *negative*. Also shown in the figure is the nearest supporting neighbor for the target problem, defined as the most similar case *C* such that *solution*(*C*) = *S*, where *S* is the solution assigned by *k*-NN to the target problem. Note that the nearest supporting neighbor in 1-NN is the same as the nearest neighbor.

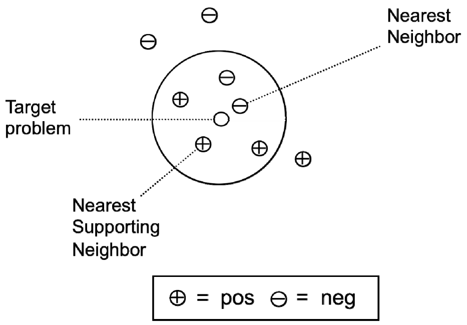


Fig. 2. The nearest "supporting" neighbor replaces the most similar case in our approach to explaining *k*-NN solutions

To explain a *k*-NN solution, we use a modified version of Learn-ER (Fig. 1) to discover an explanation rule that covers both the target problem and the nearest supporting neighbor, and correctly classifies the nearest supporting neighbor. So the only change needed in the Learn-ER algorithm is to replace the most similar case by the nearest supporting neighbor. As in our approach to explaining 1-NN solutions, the discovered explanation rule is presented to the user together with the nearest supporting neighbor. The user is also shown the accuracy and coverage of the explanation rule.

4.2 Explaining *k*-NN Solutions for the Iris Dataset

The Iris dataset [18] describes a collection of 150 Iris plants in terms of their sepal length, sepal width, petal length, and petal width, and the class attribute is the type of Iris plant (Setosa, Versicolour, Virginica). Table 3 shows the three most similar cases in the dataset retrieved by 3-NN for the target problem:

sepal length = 6.4, sepal width = 2.7, petal length = 5.0, petal width = 1.5

The most similar case is Case 134 (Virginica) but the 3-NN solution based on the majority class in the retrieval set is Verisolor. The nearest supporting neighbor is Case 73 (Versicolor) and the explanation rule discovered by Learn-ER is:

if sepal width ≤ 2.7 **and** petal width ≤ 1.6 **and** sepal length ≥ 6.2
then versicolor (1.0, 3)

The discovered explanation rule covers the target problem, the nearest supporting neighbor (Case 73), and two other cases (not shown) in the dataset. It correctly classifies the 3 cases that it covers, which do not include the most similar case (Case 134) or the other Versicolor case in the retrieval set (Case 55). However, the user is shown only the explanation rule and the nearest supporting neighbor (Case 73) in our approach to explaining *k*-NN solutions. It is also worth noting that while there are 3 conditions in the explanation rule for the CBR solution in the above example, explanation rules for *k*-NN solutions have less than two conditions on average in our experiments on the Iris dataset (Section 5).

Table 3. Three most similar cases in the Iris dataset [18] for the target problem: sepal length = 6.4, sepal width = 2.7, petal length = 5.0, petal width = 1.5

Case No.	Sepal Length	Sepal Width	Petal Length	Petal Width	Type	Similarity
134	6.3	2.8	5.1	1.5	virginica	0.978
73	6.3	2.5	4.9	1.5	versicolor	0.968
55	6.5	2.8	4.6	1.5	versicolor	0.966

5 Empirical Study

In the experiments reported in this section, we evaluate our approach to explanation in CBR in terms of its ability to provide easily understandable explanations. We assess the simplicity of explanations in our approach in terms of the average length of the discovered explanation rules (i.e., number of conditions) for a variety of classification tasks. We also investigate the hypothesis that the coverage of the explanation rule discovered for a CBR solution can be useful as an indicator of the *confidence* associated with the solution.

5.1 Experimental Method

We apply our approach to explanation in CBR to a selection of datasets from the UCI Machine Learning Repository [18]. The datasets used in our experiments are described in Table 4. The number of attributes shown for each dataset does not include the class attribute. Our experiments are based on a *leave-one-out* approach in which each case is temporarily removed from the case base and its description is presented as a problem to be solved by *k*-NN for *k* = 1, 3, and 5. An explanation rule for the CBR solution is then created using the *k*-NN version of our Learn-ER algorithm (Section 4). For each left-out case, we record whether or not the CBR solution is correct. We also record the length, coverage, and accuracy of the explanation rule for each CBR solution.

5.2 Explanation Rule Length

Table 5 shows the average lengths of the explanation rules discovered for *k*-NN solutions in our experiments on the UCI datasets. A striking feature of the results is the simplicity of the discovered explanation rules, as measured by their average length, on all the datasets including those with the largest numbers of attributes. For example, less than 2 of the 34 case attributes in the Dermatology dataset are used, on average, in the discovered explanation rules for *k*-NN solutions. It is also worth noting that Learn-ER failed to discover an explanation rule with one or more conditions in less than 1% of trials over all datasets and values of *k* in the experiment.

Table 4. UCI datasets used in the experiments

	Attributes	**Continuous Attributes**	**Cases**	**Classes**	**Missing Values**
Contact Lenses	4	–	24	3	no
Iris	4	4	150	3	no
Voting Records	16	–	435	2	yes
Hepatitis	19	6	155	2	yes
Dermatology	34	1	366	6	yes

Table 5. Average lengths of explanation rules for *k*-NN solutions

Dataset	**Attributes**	**1-NN**	**3-NN**	**5-NN**	***k*-NN Average**
Contact Lenses	4	1.79	1.42	1.54	1.58
Iris	4	1.71	1.71	1.70	1.71
Voting Records	16	2.59	2.63	2.61	2.61
Hepatitis	19	2.16	2.08	2.06	2.10
Dermatology	34	2.01	1.97	1.98	1.99

Moreover, there are only small differences in the average lengths of the discovered explanation rules for $k = 1, 3, 5$ in most of the datasets. This suggests that the value of *k* for which *k*-NN gives the best accuracy results for a given dataset can often be selected without having to consider a significant trade-off between classification accuracy and explanation quality. For some datasets, the value of *k* has an important effect on the accuracy of *k*-NN, for example with accuracy on the Hepatitis dataset increasing from 80.0% for $k = 1$ to 86.5% for $k = 5$ in our experiments.

As further evidence of the simplicity of explanations provided by our lazy learning approach to explanation in CBR, we examine the distribution of explanation rule lengths for Dermatology, the dataset with most attributes (34). Fig. 3 shows the observed lengths of explanation rules and their frequencies for 5-NN solutions on the Dermatology dataset. There is only a single condition in more than 50% of the discovered explanation rules, and rule lengths from 1 to 4 represent more than 90% of those observed in the experiment.

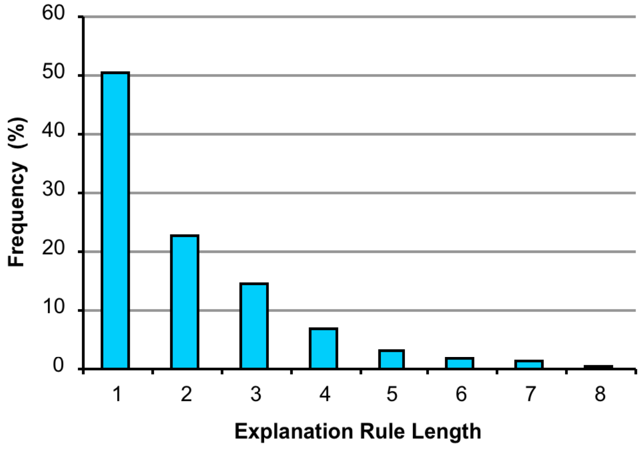


Fig. 3. Lengths of explanation rules for 5-NN solutions based on the Dermatology dataset [18]

In summary, the results from our empirical study of explanation rule length provide clear evidence of the ability of our lazy learning approach to provide simple explanations for CBR solutions.

5.3 Explanation Rule Coverage

To assess the usefulness of explanation rule coverage as an indicator of the *confidence* associated with a CBR solution, we compare the coverage of explanation rules for correct and incorrect CBR solutions in our approach. For example, Table 6 shows the minimum, average, and maximum coverage of explanation rules for correct (143) and incorrect (7) solutions provided by 3-NN on the Iris dataset [18]. An interesting feature of the results is the increase in average rule coverage by a factor of 2.04 from explanation rules for incorrect solutions (17.1) to explanation rules for correct solutions (34.9). In other words, explanation rules for correct solutions cover about twice as many cases, on average, as those for incorrect solutions.

Table 6. Minimum, average, and maximum coverage of explanation rules for correct (143) and incorrect (7) 3-NN solutions based on the Iris dataset [18]

	Minimum Coverage	**Average Coverage**	**Maximum Coverage**
Correct Solutions	1	34.9	49
Incorrect Solutions	1	17.1	45

Average coverage of explanation rules was also greater for correct solutions in our experiments with *k*-NN for *k* = 1, 3, 5 on the other UCI datasets. The results are summarized in Table 7.

Table 7. Factors by which average coverage of explanation rules increases from incorrect solutions to correct solutions

Dataset	1-NN	3-NN	5-NN
Contact Lenses	1.13	1.11	1.20
Iris	2.17	2.04	1.72
Voting Records	2.34	2.23	1.78
Hepatitis	1.51	1.45	1.25
Dermatology	3.51	2.86	4.18

These findings support our hypothesis that explanation rule coverage can be useful as an indicator of the confidence associated with a CBR solution. However, explanation rule coverage is far from being a definitive measure for assessing the quality of CBR solutions. For example, it can be seen from Table 6 that an explanation rule for a correct solution may have very low coverage, while an explanation rule for an incorrect solution may have higher coverage than the average for correct solutions. It is also worth noting that explanation rules for correct solutions had lower coverage, on average, than those for incorrect solutions in a similar (1-NN) experiment on the travel methods case base (Table 1). However, the latter result should perhaps be treated with caution given the small size and artificial nature of the travel methods case base.

6 Conclusions

We presented a lazy learning approach to explanation in CBR that aims to provide more informative explanations than is possible by simply showing the user the most similar case. Our demand-driven approach to the discovery of explanation rules for CBR solutions is likely to be most beneficial in situations where the most similar case (or nearest supporting neighbor in *k*-NN) has many features in common with the target problem, making it difficult for the user to tell which features are most relevant in the context of the proposed solution. We also presented experimental results for a variety of classification tasks, demonstrating the ability of the proposed approach to provide easily understandable explanations for CBR solutions. For example, less than 2 of the 34 case attributes in the Dermatology dataset [18] were used, on average, in discovered explanation rules for *k*-NN solutions. Another important finding of the research presented is that the coverage of an explanation rule can sometimes be useful as an indicator of the confidence associated with the CBR solution it is used to explain.

References

1. Cunningham, P., Doyle, D., Loughrey, J.: An Evaluation of the Usefulness of Case-Based Explanation. In: Ashley, K.D., Bridge, D.G. (eds.) ICCBR 2003. LNCS(LNAI), vol. 2689, pp. 122–130. Springer, Heidelberg (2003)
2. Doyle, D., Cunningham, P., Bridge, D.G., Rahman, Y.: Explanation Oriented Retrieval. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 157–168. Springer, Heidelberg (2004)

3. Evans-Romaine, K., Marling, C.: Prescribing Exercise Regimens for Cardiac and Pulmonary Disease Patients with CBR. In: McGinty, L. (ed.) ICCBR 2003 Workshop Proceedings, pp. 45–52. NTNU, Dept. of Computer and Information Science, Trondheim (2003)
4. Leake, D., McSherry, D.: Introduction to the Special Issue on Explanation in Case-Based Reasoning. Artif. Intell. Rev. 24, 103–108 (2005)
5. Massie, S., Craw, S., Wiratunga, N.: A Visualisation Tool to Explain Case-Base Reasoning Solutions for Tablet Formulation. In: Macintosh, A., Ellis, R., Allen, T. (eds.) AI 2004, pp. 222–234. Springer, London (2005)
6. Maximini, R., Freßmann, A., Schaaf, M.: Explanation Service for Complex CBR Applications. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 302–316. Springer, Heidelberg (2004)
7. McSherry, D.: Conversational Case-Based Reasoning in Medical Decision Making. Artif. Intell. Med. 52, 59–66 (2011)
8. McSherry, D.: Explaining the Pros and Cons of Conclusions in CBR. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 317–330. Springer, Heidelberg (2004)
9. Plaza, E., Armengol, E., Ontañón, S.: The Explanatory Power of Symbolic Similarity in Case-Based Reasoning. Artif. Intell. Rev. 24, 145–161 (2005)
10. Rissland, E.L.: The Fun Begins with Retrieval: Explanation and CBR. In: Roth-Berghofer, T.R., Göker, M.H., Güvenir, H.A. (eds.) ECCBR 2006. LNCS (LNAI), vol. 4106, pp. 1–8. Springer, Heidelberg (2006)
11. Roth-Berghofer, T.R.: Explanations and Case-Based Reasoning: Foundational Issues. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 389–403. Springer, Heidelberg (2004)
12. Sørmo, F., Cassens, J., Aamodt, A.: Explanation in Case-Based Reasoning – Perspectives and Goals. Artif. Intell. Rev. 24, 109–143 (2005)
13. Tan, P.-N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley, Boston (2006)
14. Cendrowska, J.: PRISM: an Algorithm for Inducing Modular Rules. Int. J. Man. Mach. Stud. 27, 349–370 (1987)
15. Bramer, M.A.: Principles of Data Mining. Springer, London (2007)
16. Bramer, M.A.: Inducer: A Public Domain Workbench for Data Mining. Int. J. Syst. Sci. 36, 909–919 (2005)
17. Stahl, F., Bramer, M.A.: Induction of Modular Classification Rules: Using Jmax-Pruning. In: Bramer, M.A., Petridis, M., Hopgood, A. (eds.) AI 2010, pp. 79–92. Springer, London (2010)
18. Frank, A., Asuncion, A.: UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences (2010)

Confidence in Workflow Adaptation

Mirjam Minor, Mohd. Siblee Islam, and Pol Schumacher

University of Trier - Department of Business Information Systems II
D-54286 Trier, Germany
{minor,islam,pol.schumacher}@uni-trier.de

Abstract. This paper is on assessing the quality of adaptation results by a novel confidence measure. The confidence is computed by finding evidence for partial solutions from introspection of a huge case base. We assume that an adaptation result can be decomposed into portions, that the provenance information for the portions is available. The adaptation result is reduced to such portions of the solution that have been affected by the change. Furthermore, we assume that a similarity measure for retrieving the portions from a case base can be specified and that a huge case base is available providing a solution space. The occurrence of each portion of the reduced solution in the case base is investigated during an additional retrieval phase after having adapted the case. Based on this idea of retrieving portions, we introduce a general confidence measure for adaptation results. It is implemented in the area of workflow adaptation. A graph-based representation of cases is used. The adapted workflow is reduced to a set of sub-graphs affected by the change. Similarity measures are specified for a graph matching method that implements the introspection of the case base. Experimental results on workflow adaptations from the cooking domain show the feasibility of the approach. The values of the confidence measure have been evaluated for three case bases with a size of 200, 2,000, and 20,000 cases each by comparing them with an expert assessment.

1 Introduction

Adaptation has achieved significant attention in Case-Based Reasoning (CBR) over the past few years [1–7]. A retrieved solution has to be adapted in order to be reused for a current problem [8]. Several adaptation techniques and frameworks have been introduced, which apply rules or operators [1–3], merge cases [4, 5], or reuse dedicated adaptation cases [6, 7].

In most adaptation approaches, it is difficult to assess the quality of the adaptation result with respect to the given problem a priori. A notable exception is the work on adaptation for configuration tasks [1] where a cost function has been employed as a quality measure for a solution. In the absence of a formal quality measure like a cost function, the quality of the adaptation result can be approximated by quality indicators like the utility, correctness, or completeness of the solution. The utility of the solution might serve as a quality criterion which can be approximated by the value of the similarity function the retrieved solution has achieved prior to the adaptation [9]. Furthermore, correctness criteria like the syntactical correctness of the adapted solution [7] or the consistency of the solution with a knowledge model [5] can be considered. The completeness of the adaptation describes to what extent the problem is covered by the adaptation result [4].

In this paper, we introduce a confidence measure for an adaptation result, which serves as an additional indicator for the quality of the adapted solution. We focus on a special type of problem cases, namely on workflows that are to be adapted. However, our notion of confidence in adaptation is not restricted to cases that contain procedural knowledge or workflows. It can be applied as well for structural or textual cases. Our ideas have been inspired by confidence measures from data mining where the confidence c of a discovered association rule $X \Rightarrow Y$ for two data items X and Y is predicted by occurrence, namely by the percentage of the data itemsets containing X (i.e. d_X) that also contain Y ($d_{X,Y}$) [10], expressed by $c = \frac{d_{X,Y}}{d_X}$. In CBR, confidence has also been discussed. The confidence in a solution retrieved from a case base has been investigated [11] as well as the confidence in a classification created by a CBR system [12]. Both CBR approaches predict confidence based on similarity measures. In our approach, the confidence in an adapted solution is predicted based on introspection of the case base by determining whether parts of the solution occur somewhere else. The adapted solution is (I) reduced to those portions that have been affected by the adaptation, (II) a retrieval is performed for each portion, and (III) a confidence value is derived from the retrieval results. The approach is limited to positive confidence values from a case base of "good experiences". Furthermore, it is limited to a local perspective since it does not consider dependencies between the portions. Thus, our confidence measure can be regarded one indicator among others for the quality of an adaptation result. Our hypothesis is that such an indicator enables the CBR system to suggest only solutions with adapted portions that have some evidence by occurence in the case base and thus increase the confidence in the entire solutions.

The remainder of the paper is organized as follows. In Section 2, we briefly sketch the adaptation of workflows and explain how the adapted portions can be tracked during this process. In Section 3, we introduce a confidence measure for adaptation results and apply it for adapted workflows. A formative evaluation is described in Section 4. We conclude the paper with a summary and a discussion of future work in Section 5.

2 Workflow Adaptation

In order to assess an adapted solution by a confidence measure, the adaptation process has to be tracked and those portions of the solution have to be identified that have been modified during adaptation. Our approach focusses on workflow adaptation. Workflows are "the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules"

[13]. Adaptive workflow systems (also called agile workflow systems) facilitate structural changes of workflows at run time [14–18]. Workflows can be created (for instance based on a template from a repository) and tailored for a particular demand or business case. Workflows can still be adapted after they have been started, for example if some unforeseen events occur. The changes apply to workflow elements, i.e., to atomic parts of the workflow. In our approach, we track workflow elements that have been modified during automated adaptation. The particular adaptation method that has been used to modify the workflow is not of interest for the confidence measure. For the experimental evaluation (compare Section 4), we have chosen a case-based adaptation approach [7]. Any alternative adaptation approach, e.g. a rule-based or plan-based approach, would have been applicable as well.

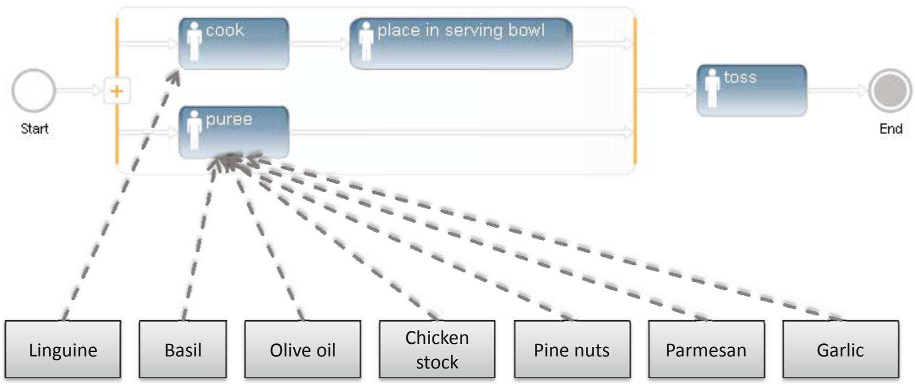


Fig. 1. Sample workflow in CFCN for cooking pasta with basil pesto

Figure 1 depicts a sample workflow from the cooking domain. We use the Cake Flow Cloud Notation (CFCN, compare [19]) to illustrate the work. CFCN has been developed in recent research projects at the University of Trier [19, 20] as a part of the Collaborative Agile Knowledge Engine (Cake) [21, 22]. The cake system provides modeling and enactment support for workflows including adaptation support. CFCN consists of several types of workflow elements like tasks, data objects, data links (from a data object to a task or vice versa), and control flow elements like AND-splits, AND-joins etc. The sample in Figure 1 describes a recipe for basil pesto sauce over pasta. The ingredients are represented by data objects ("Linguine", "Basil", etc.) while the cooking activities form the tasks ("cook", "puree", etc.). The linguine are cooked in parallel to pureeing the other ingredients as indicated by the AND-split symbolized by a '+'. In Figure 2, an adapted workflow is shown, in which the "Pine nuts" have been substituted by "Walnuts". This requires an additional task "crack" that has been inserted before "puree".

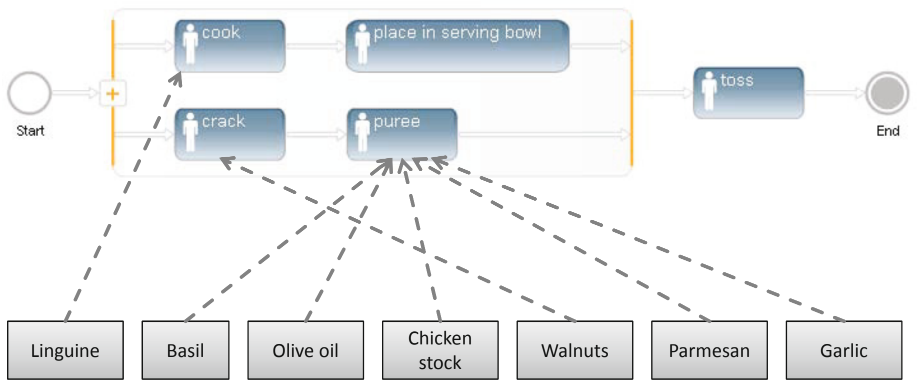


Fig. 2. Adapted sample workflow from Figure 1

In order to track the adaptation process of a workflow, the provenance of each workflow element is used, i.e., whether it stems from the original workflow or whether it has been inserted or modified during the adaptation process, like the task "crack", for instance. As a result, the list of those tasks is stored that have been affected by the adaptation. This includes newly inserted tasks as well as tasks with new or deleted data links. In the above sample, the list of modified tasks consists of the task "crack" which has been newly inserted together and the task "puree" since its former data object "Pine nuts" has been deleted during adaptation.

3 Confidence in Adaptation Quality

The confidence in an adapted solution is predicted by introspection for the changed parts. The approach makes the following three assumptions: The adapted solutions can be decomposed into parts, which are described with the same formalism as the solutions. The provenance information for each of those parts is available. A similarity measure can be defined for the parts with respect to the case base. If these assumptions are fulfilled, a confidence measure can be defined as follows.

Let $\mathcal{S}$ be the (possibly infinite[1]) universe of adapted solutions and $\mathcal{CB}$ the universe of case bases. The confidence c can be predicted by a confidence measure $c : \mathcal{S} \times \mathcal{CB} \rightarrow \mathbb{R}$.

A reduction function $reduce : \mathcal{S} \rightarrow 2^{\mathcal{S}}$ transforms an adapted solution $s \in \mathcal{S}$ into a reduced, decomposed solution $\hat{S} \in 2^{\mathcal{S}}$. The solution $\hat{S} = reduce(s) = \{s_1, s_2, ...s_n\}$ consists only of the portions of s that have been affected by the adaptation according to the provenance information. The atomic units for the decomposition have to be specified properly, such that the s_i's can be compared with the cases from the case base later on. With this, the confidence for a solution

[1] For instance, in case of adaptation rules that can be arbitrarily often repeated.

$s \in S$ with respect to a case base $CB \in \mathcal{CB}$ can be computed by means of an aggregated similarity function sim_{Φ}:

$$c(s, CB) = sim_{\Phi}(\{s_1, s_2, ..., s_n\}, CB)$$

where Φ is an aggregation function $\Phi : \mathbb{R}^n \to \mathbb{R}$ for the similarity values of the particular portions s_i with $i = 1, 2, ..., n$ and the cases c_j from the case base $CB = \{c_1, c_2, ..., c_m\}$. For instance, a maximum function for the best matching case to every portion can be chosen and combined with a minimum function for the portion that has achieved the lowest similarity value:

$$\begin{aligned} sim_{\Phi}(\hat{S}, CB) = \min_{s_i} \max_{c_j} sim(s_i, c_j) \\ s.t.\ s_i \in \hat{S}, c_j \in CB \end{aligned}$$

The maximum function in Φ is quite natural, as it is implemented by any retrieval, which computes the best matching case. For each portion, the "best" occurrence is chosen. Chosing the minimum value over all portions expresses the pessimistic view that the portion that achieved the smallest maximum similarity value determines the overall confidence. To be more optimistic, the minimum function could be replaced, for instance, by a weighted sum. In the following, the reduce and retrieve steps will be described for adapted workflows.

3.1 Reduce

An adapted workflow is converted into a reduced adapted workflow that comprises only the affected tasks with corresponding data items related to the tasks. The *reduce* function derives the reduced workflow from the list of tasks that have been tracked as inserted or modified during the workflow adaptation (compare Section 2). The tasks are stored with all their related data objects. Figure 3 represents the filtered workflow derived in a first part of the *reduce* function from the main workflow represented in Figure 2. The resulting workflow is a sequence of the two tasks "crack" with the data object "Walnuts" and "puree" with the five remaining data objects "Basil", "Olive oil", "Chicken stock", "Parmesan" and "Garlic".

As the second part of the *reduce* function, the reduced workflow is decomposed into a set of queries $\hat{S} = \{s_1, s_2, ..., s_n\}$ in preparation for the retrieval. For this, we have chosen a graph representation recently introduced by Bergmann and Gil [23]. A workflow can be transformed into a directed graph $W = (N, E, S, T)$ where N is a set of nodes and $E \subset N \times N$ is a set of edges having a type $T\colon N \cup E \to \Omega$ and a semantic description $S\colon N \cup E \to \Sigma$ where the type and semantic description are associated to each node and edge that are taken from Ω and Σ respectively. Ω consists of the types *workflow node, data node, task node, control flow node, control flow edge, part of edge* and *data flow edge*. W has exactly one *workflow node*. The *task nodes* and *data nodes* represent tasks and data objects respectively. A *control flow node* stands for contol flow elements. The *data flow edge* is used to describe the data links. The *control flow edge* is

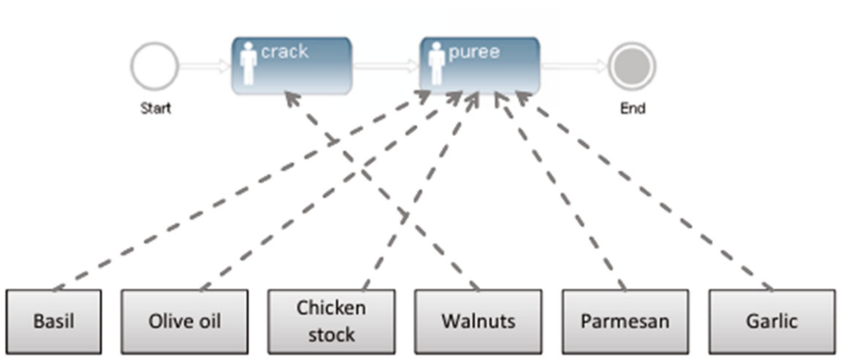


Fig. 3. Filtered workflow

used to represent the control flow of the workflow, e.g., from task to task or from task to control flow element. The *part of edge* shows the relation between *workflow node* and *data node*, *task node*, or *control flow node*. Σ is a semantic meta data language. In our approach, it consists of a universe of names for tasks and data objects. W is then split into sub-graphs for each task that consist of the workflow node and one task node along with the data nodes that are related to it. Thus, $\hat{S}$ consists of a set of sub-graphs s_i. Figure 4a depicts a sample graph W representing the filtered workflow derived from the main workflow in Figure 3. The sub-graphs derived from the above sample graph W are depicted in Figure 4b and 4c.

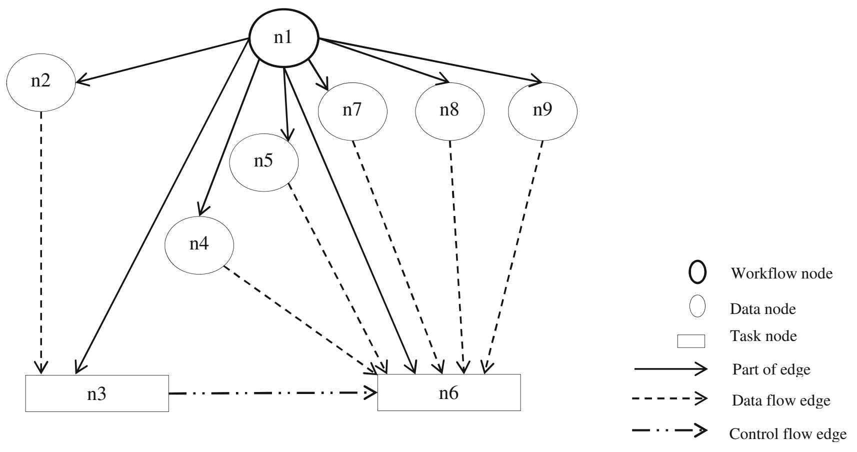
3.2 Retrieve

Each sub-graph s_i of the reduced workflow representation described above is asked as a query to the case base. For each s_i, the retrieval is performed by means of a graph matching method introduced by Bergmann and Gil [23]. Broadly speaking, nodes and edges of the query graph s_i are mapped to the best matching nodes and edges of a graph c_j from the case base. The mapping with the highest similarity is chosen as a retrieval result.

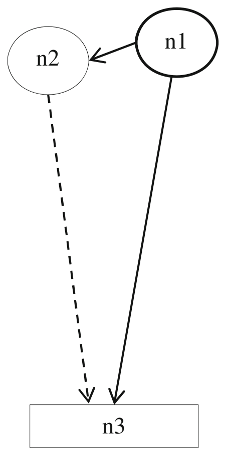
Local similarity functions for graph nodes (sim_N) and edges (sim_E) have to be specified. The similarity of a query node $n_q \epsilon N$ with a case node $n_c \epsilon N$ is described by the similarity function $sim_N(n_q, n_c)$ according to Bergmann and Gil by:

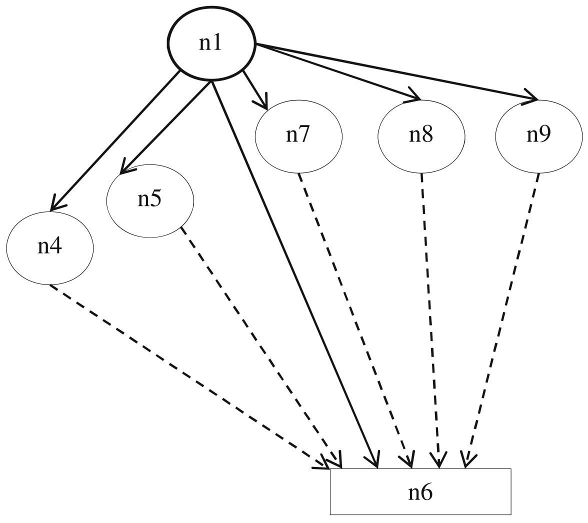
$$sim_N(n_q, n_c) = \begin{cases} sim_\Sigma(S_q(n_q), S_c(n_c)) & \text{if } T_q(n_q) = T_c(n_c) \\ 0 & \text{otherwise} \end{cases}$$

sim_Σ denotes a local similarity function for semantic descriptions. We have chosen a similarity function sim_Σ that is derived from a Levenshtein distance measure. The Levenshtein distance is purely syntactic and measures the minimum number of edit operations to transform one string into another at the character level.



(a) Graph W for the sample workflow from Figure 3





(b) Sub-graph s_1 for task "crack"

(c) Sub-graph s_2 for task "puree"

Fig. 4. Decomposition of a sample workflow graph into two sub-graphs

The similarity of a query edge $e_c \epsilon E$ with a case edge $e_q \epsilon E$ is described by the similarity function $sim_E(e_q, e_c)$ according to Bergmann and Gil by:

$$sim_E(e_q, e_c) = \begin{cases} F_E \begin{pmatrix} sim_\Sigma(S_q(e_q), S_c(e_c)), \\ sim_N((e_q.l), (e_c.l)), \\ sim_N((e_q.r), (e_c.r) \end{pmatrix} & \text{if } T_q(e_q) = T_c(e_c) \\ 0 & \text{otherwise} \end{cases}$$

Where F_E is specified as $F_E(S_e, S_l, S_r) = S_e * 0.5 * (S_l + S_r)$ and

$$sim_\Sigma = \begin{cases} 1 & \text{if } T_q(e_q) = T_c(e_c) \\ 0 & \text{otherwise} \end{cases}$$

The similarity of two graphs s_i and c_j is computed by means of legal mappings, i.e., a node can be mapped by a partial injective mapping function $m : N_q \cup E_q \rightarrow N_c \cup E_c$ if the following five constraints are satisfied:

$$\begin{array}{lll} T_q(n_q) = T_c(m(n_q)) & T_q(e_q) = T_c(m(e_q)) & \\ m_q(e_q.l) = m(e_q.l) & m_q(e_q.r) = m(e_q.r) & \forall_{x,y} m(x) = m(y) \rightarrow x = y \end{array}$$

Two edges can be mapped if the respective nodes that are connected by the edges can also be mapped.

The mapping can be partial. The similarity is computed for all possible mappings as described by the following equation where $Dom(m)$ is domain of m.

$$sim_m(s_i, c_j) = F_w \begin{pmatrix} (sim_N(n, m(n)) | n \epsilon N_q \cap Dom(m)), \\ (sim_E(e, m(e)) | e \epsilon E_q \cap Dom(m)), \\ |N_q, E_q| \end{pmatrix}$$

Where $F_w((sn_1, ..., sn_i), ((se_1, ..., se_j), n_N, n_E) = \frac{sn_1 + ... + sn_i + se_1 + ... + se_j}{n_N + n_E}$.

The mapping with the maximum similarity value sim_m is chosen as the overall similarity sim between a solution part $s_i \in \mathcal{S}$ and a case $c_j \in CB$. Without loss of generality, we have chosen local similarity functions with values between 0 and 1, i.e., $sim : \mathcal{S} \times CB \rightarrow [0, 1]$.

4 Evaluation

We conducted some experiments in the cooking domain in order to evaluate the approach. Cooking instructions have been formalized as workflows as described in Section 2. Three experimental case bases CB_{200}, CB_{2000}, and CB_{20000} with 200, 2,000, and 20,000 cooking workflows each have been created by an workflow extraction approach [24] from recipes of an online cooking community[2]. For 26 sample workflows, change requests have been formulated by hand. 19 of them could be adapted successfully by an automated, case-based adaptation method [7] (compare Table 1). The confidence values for the adapted workflows have

[2] www.allrecipes.com

Table 1. Confidence values and quality assessment of successfully adapted workflows

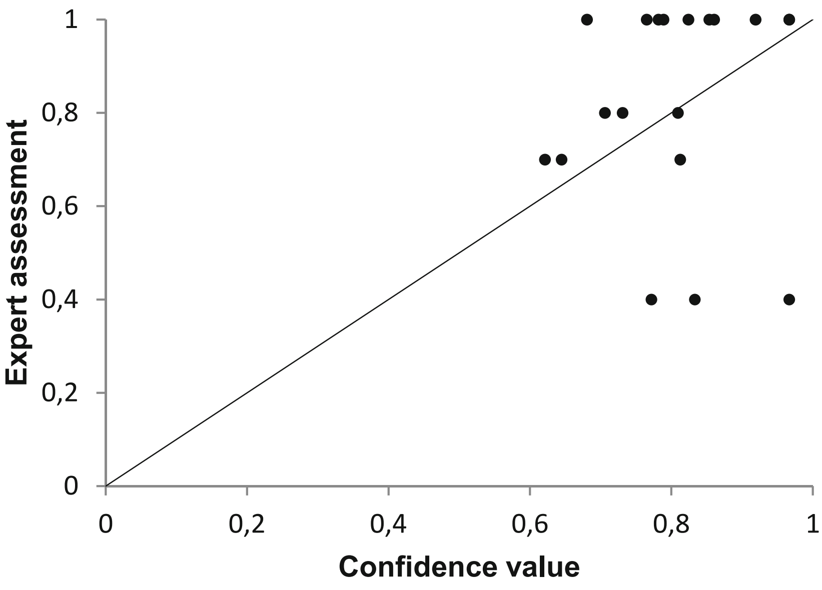
Case no.	Change request	CB_{200}	CB_{2000}	CB_{20000}	e_s
1	Omit onions	0.789	0.846	0.876	1.0
2	Replace commercial soup by sauce hollandaise	0.645	0.694	0.772	0.7
3	Replace garlic by ramsons	0.809	0.809	0.816	0.8
4	Replace butter by olive oil	0.813	0.820	0.887	0.7
5	Replace commercial soup by a mixture of whipping cream and eggs	0.621	0.668	0.717	0.7
6	Omit olives	0.919	0.919	0.967	1.0
7	Replace olive oil by butter	0.967	0.967	0.967	1.0
8	Replace spaghetti by macaroni	0.967	0.967	0.967	1.0
9	Top additionally with cheese	0.967	0.967	0.967	0.4
10	Omit mushrooms	0.731	0.727	0.771	0.8
11	Replace ricotta and cream by cottage cheese	0.833	0.879	0.907	0.4
12	Replace spinach by chard	0.782	0.814	0.814	1.0
13	Replace pine nuts by almonds	0.772	0.872	0.852	0.4
14	Omit pimiento	0.706	0.720	0.794	0.8
15	Omit nutmeg	0.854	0.871	0.898	1.0
16	Replace sundried tomatoes by fresh tomatoes	0.681	0.712	0.808	1.0
17	Refine olive oil with herbs	0.824	0.839	0.887	1.0
18	Omit capers	0.765	0.808	0.876	1.0
19	Omit parsley	0.860	0.882	0.896	1.0

been computed by the introspective confidence measure as described above. An expert was asked to assess the quality of the adaptation results and assign scores from 0 (for "bad") to 10 (for "very well") to each of the adapted workflows. This empirical value reflects the subjective opinion of the expert whether the recipe described by the adapted workflow would produce a tasty dish. We compared the confidence values achieved for CB_{200}, CB_{2000} and CB_{20000} with the quality scores assigned by the expert.

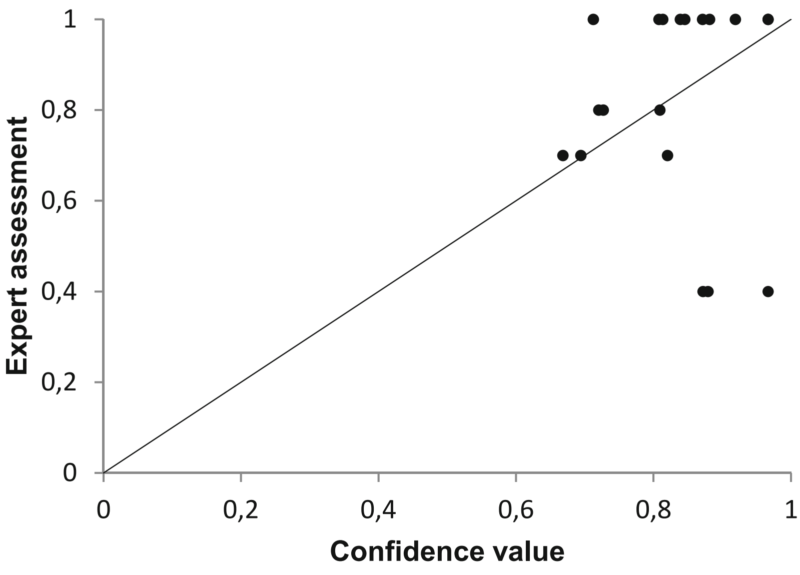
The following hypotheses have been investigated:

H1 The values of the confidence measure increase with the size of the case base.
H2 The values computed by the confidence measure are lower than the values provided by the expert $c(s, CB) \leq e_s$, i.e., the confidence measure provides an underestimation for the empirical quality.
H3 The values computed by the confidence measure approach the expert values with an increasing case base size: $|c(s, CB_i) - e_s| \geq |c(s, CB_j) - e_s|$ if $|CB_i| \leq |CB_j|$.

Hypothesis $H1$ was confirmed by the experiments (see the confidence values in Table 1). Only for change requests 10 and 11, the values of c decrease slightly from case base CB_{200} to CB_{2000}. Figures 5a - c depict the expert values in comparison to the automated values.

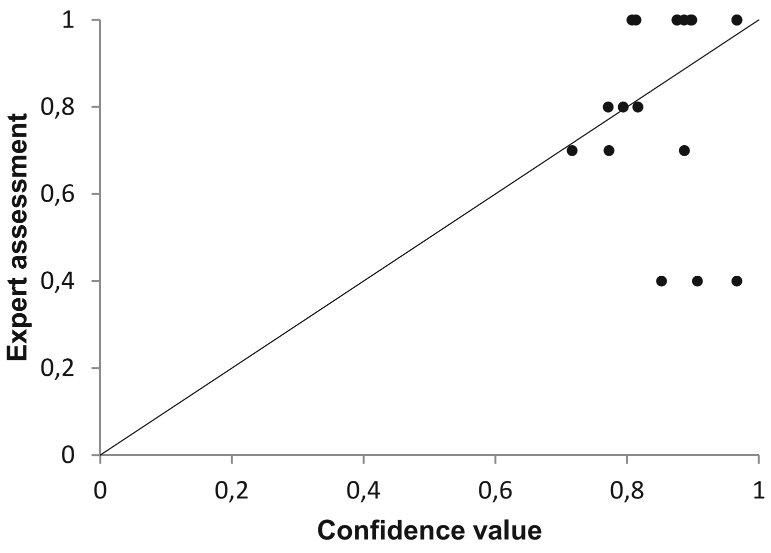


(a) for CB_{200}



(b) for CB_{2000}

Fig. 5. Comparison of the expert assessments with the confidence values



(c) for CB_{20000}

Fig. 5. (*Continued*)

Hypothesis $H2$ could not be confirmed as 17 of the 57 values of c overestimate the quality value assigned by the expert. The values for change requests 9, 11 and 13 clearly contradict $H2$. The reason for the low expert value given for change request 9 is that the adaptation method suggested to cook the additional cheese instead of topping the dish with cheese. In the recipe for change request 11, the cottage cheese has been inserted at the wrong place by the automated adaptation. In the recipe for change request 13, the expert did not like that the almonds have not been cut before using them in the recipe. However, the few other values are only slight overestimations. In Figures 5a - c, the data points below the diagonals illustrate the overestimating values.

Hypothesis $H3$ was confirmed by 30 of 38 values. We repeated the experiment with three groups of 10 case bases of 200, 2,000 and 20,000 cases that have been extracted from the same recipe Web page arbitrarily. The average difference between the expert value and the values of the ten $c(s, CB_{(200,i)})$ is 0.157, 0.137 for $c(s, CB_{(2000,i)})$ and 0.129 for $c(s, CB_{(20000,i)})$. We admit that the number of measured values is too small for a statistically solid statement.

Hypothesis $H1$ and $H3$ have been confirmed by our experiments while $H2$ was contradicted. One reason for the results on $H2$ could be, that the similarity function applied for the sub-graphs was too optimistic since it tolerates incomplete mappings. Furthermore, it seems promising to relax the property of being an underestimation for future evaluations on hypothesis $H2$.

5 Conclusion

In this paper, we introduced a confidence measure for adaptation results based on introspection of the case base. The adaptation results are decomposed into portions, the provenance for each portion is determined, and those portions that stem from the adaptation process are retrieved from the case base. The occurrence of a portion provides some empirical evidence for the feasibility of the adapted solution. Without loss of generality, we restricted the approach to workflow adaptation. We defined a confidence measure based on a graph representation. We have chosen sub-graphs as atomic portions of the adapted solution, which consist of a workflow task with its according data objects. The retrieval has been performed by a sub-graph matching. We conducted some experiments on cooking workflows describing a cooking instruction from a recipe step-by-step. Automatically adapted workflows have been assessed by both, a confidence measure and a human expert. The experiments provided promising results, since the confidence values improved with the size of the case base with respect to baseline values from the expert. The results confirmes the feasibility of the confidence measure.

The approach provides many opportunities for future work. The experiments should be repeated in further domains of workflow adaptation. The measure could be specified also for the adaptation results of structural or textual cases. The confidence measure could be complemented by a measure for potential dependencies between adapted portions or by user scores of the cases containing the retrieved portions. Negative confidence values could be included, e.g. from inverse workflow patterns [25]. Different variants for the similarity measures underlying the confidence measure could be investigated. Our next steps will be to consider more semantic information in the local similarity measures, for instance, from task ontologies, and to conduct experiments with further workflow domains.

Acknowledgements. This work was funded by the German Research Foundation, project number BE 1373/3-1. Furthermore, we would like to thank the student researchers Stephan Zonker and Alexander Stromer for their contributions in implementing and evaluating the approach. In addition, we appreciate the feedback from Ralph Bergmann, which was very helpful to improve the paper.

References

[1] Bergmann, R., Wilke, W.: Towards a new formal model of transformational adaptation in case-based reasoning. In: Proceedings of th German Workshop on CBR, University of Rostock, pp. 43–52 (1998)

[2] De Mantaras, R., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M., Cox, M., Forbus, K., et al.: Retrieval, reuse, revision and retention in case-based reasoning. Knowledge Engineering Review 20(3), 215–240 (2005)

[3] Muñoz-Avila, H., Cox, M.T.: Case-based plan adaptation: An analysis and review. IEEE Intelligent Systems 23(4), 75–81 (2008)

[4] d'Aquin, M., Lieber, J., Napoli, A.: Decentralized Case-Based Reasoning for the Semantic Web. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 142–155. Springer, Heidelberg (2005)

[5] Cojan, J., Lieber, J.: Belief Merging-Based Case Combination. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 105–119. Springer, Heidelberg (2009)

[6] Leake, D., Kinley, A., Wilson, D.: Learning to Improve Case Adaptation by Introspective Reasoning and CBR. In: Aamodt, A., Veloso, M.M. (eds.) ICCBR 1995. LNCS, vol. 1010, pp. 229–240. Springer, Heidelberg (1995)

[7] Minor, M., Bergmann, R., Görg, S., Walter, K.: Reasoning on business processes to support change reuse. In: 13th IEEE Conference on Commerce and Enterprise Computing, pp. 18–25. IEEE Computer Society, Luxemburg (2011)

[8] Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications 7(1), 39–59 (1994)

[9] Lee-Urban, S., Muñoz-Avila, H.: Adaptation versus Retrieval Trade-Off Revisited: An Analysis of Boundary Conditions. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 180–194. Springer, Heidelberg (2009)

[10] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A., et al.: Fast discovery of association rules. In: Advances in Knowledge Discovery and Data Mining, vol. 12, pp. 307–328 (1996)

[11] Cheetham, W., Price, J.: Measures of Solution Accuracy in Case-Based Reasoning Systems. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 106–118. Springer, Heidelberg (2004)

[12] Delany, S.J., Cunningham, P., Doyle, D., Zamolotskikh, A.: Generating Estimates of Classification Confidence for a Case-Based Spam Filter. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS (LNAI), vol. 3620, pp. 177–190. Springer, Heidelberg (2005)

[13] Workflow Management Coalition: Workflow management coalition glossary & terminology (1999), `http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf` (last access on May 23, 2007)

[14] Reichert, M., Dadam, P.: ADEPT flex–supporting dynamic changes of workflows without losing control. Journal of Intelligent Information Systems 10(2), 93–129 (1998)

[15] Weber, B., Wild, W.: Towards the Agile Management of Business Processes. In: Althoff, K.-D., Dengel, A.R., Bergmann, R., Nick, M., Roth-Berghofer, T.R. (eds.) WM 2005. LNCS (LNAI), vol. 3782, pp. 409–419. Springer, Heidelberg (2005)

[16] Minor, M., Tartakovski, A., Schmalen, D., Bergmann, R.: Agile workflow technology and Case-Based change reuse for Long-Term processes. International Journal of Intelligent Information Technologies 4(1), 80–98 (2008)

[17] Leake, D.B., Kendall-Morwick, J.: Towards Case-Based Support for e-Science Workflow Generation by Mining Provenance. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 269–283. Springer, Heidelberg (2008)

[18] Montani, S., Leonardi, G.: A Case-Based Approach to Business Process Monitoring. In: Bramer, M. (ed.) IFIP AI 2010. IFIP AICT, vol. 331, pp. 101–110. Springer, Heidelberg (2010)

[19] Minor, M., Bergmann, R., Görg, S.: Adaptive workflow management in the cloud -Towards a novel platform as a service. In: Proceedings of the ICCBR 2011 Workshops, pp. 131–138 (2011)

[20] Görg, S., Bergmann, R., Minor, M., Gessinger, S., Islam, S.: Collecting, reusing and executing private workflows on social network platforms. In: WWW 2012 Workshop Proceedings (2012)

[21] Bergmann, R., Freßmann, A., Maximini, K., Maximini, R., Sauer, T.: Case-Based Support for Collaborative Business. In: Roth-Berghofer, T.R., Göker, M.H., Güvenir, H.A. (eds.) ECCBR 2006. LNCS (LNAI), vol. 4106, pp. 519–533. Springer, Heidelberg (2006)

[22] Minor, M., Schmalen, D., Kempin, S.: Demonstration of the agile workflow management system cake II based on Long-Term office workflows. In: CEUR Proceedings of the BPM 2009 Demonstration Track, Business Process Management Conference 2009 (BPM 2009), vol. 489 (September 2009)

[23] Bergmann, R., Gil, Y.: Retrieval of Semantic Workflows with Knowledge Intensive Similarity Measures. In: Ram, A., Wiratunga, N. (eds.) ICCBR 2011. LNCS, vol. 6880, pp. 17–31. Springer, Heidelberg (2011)

[24] Schumacher, P., Minor, M., Walter, K., Bergmann, R.: Extraction of procedural knowledge from the web. In: WWW 2012 Workshop Proceedings. ACM (2012)

[25] Sauer, T., Minor, M., Bergmann, R.: Inverse workflows for supporting agile business process management. In: Maier, R. (ed.) WM 2011: 6th Conference on Professional Knowledge Management. LNI, vol. P-182, pp. 194–203 (2011)

Retrieval and Clustering for Business Process Monitoring: Results and Improvements

Stefania Montani and Giorgio Leonardi

DISIT, Sezione di Informatica, Università del Piemonte Orientale, Alessandria, Italy

Abstract. Business process monitoring is a set of activities for organizing process instance logs and for highlighting non-compliances and adaptations with respect to the default process schema. Such activities typically serve as the starting point for a-posteriori log analyses.

In recent years, we have implemented a tool for supporting business process monitoring, which allows to retrieve traces of process execution similar to the current one. Moreover, it supports an automatic organization of the trace database content through the application of clustering techniques. Retrieval and clustering rely on a distance definition able to take into account temporal information in traces.

In this paper, we report on such a tool, and present the newest experimental results.

Moreover, we introduce our recent research directions, that aim at improving the tool performances, usability and visibility with respect to the scientific community.

Specifically, we propose a methodology for avoiding exhaustive search in the trace database, by identifying promising regions of the search space, in order to reduce computation time.

Moreover, we describe how our work is being incorporated as a plug-in in ProM, an open source framework for process mining and process analysis.

1 Introduction

Business Process (BP) monitoring is a set of activities for organizing process instance logs and for highlighting non-compliances and adaptations with respect to the default process schema. Such activities typically serve as the starting point for a-posteriori log analyses.

In recent years, we have implemented a BP monitoring framework [16, 18, 17], which we have so far applied to the field of stroke management[1]. Our tool supports *end users* (e.g. user physicians) in process instance modification, by retrieving traces of execution similar to the current one: suggestions on how to adapt the default process schema in the current situation may be obtained by analyzing the most similar retrieved examples of change, recorded as traces that share the starting sequence of actions with the current query. Additionally, our framework can automatically cluster the execution traces stored in the database on the basis

[1] However, our work is domain independent.

of their similarity, and allows *process engineers* (e.g. expert physicians, administrators) to inspect the obtained clusters, in order to visualize the most frequent changes. Indeed, since changes can be an indicator of non-compliance, clustering can be seen as the first step in a process quality evaluation activity, which can be realized by means of formal (e.g. logic-based) verification approaches. Moreover, since changes can also be due to a weak or incomplete initial process schema definition, engineers can exploit retrieval and clustering results to draw some suggestions on how to redefine process schemata, in order to incorporate the most frequent and significant changes once and for all, at least at the local level (e.g. referring to the hospital they work for).

In our framework, trace retrieval relies on the *retrieval* step of Case-Based Reasoning (CBR) [1]. As for clustering, we resort to a hierarchical clustering technique, known as Unweighted Pair Group Method with Arithmetic Mean (UPGMA) [24]. Both retrieval and clustering exploit a distance definition able to take into account temporal information in traces. Technical details of the approach are summarized in section 2.

Our tool is being tested in the stroke management domain. In section 3 we report on some new experiments, specifically referring to the clustering functionality.

The results we have obtained up to now are very encouraging, but were drawn on quite a small trace database.

On the other hand, we are aware that search and calculation of similarity can become computationally expensive when working on very large databases. This problem has already been highlighted in process instance databases retrieval [4].

To this end, we are currently designing and implementing a methodology able to enhance the performances of our tool, avoiding exhaustive search of similar traces. Specifically, we are resorting to a *pivoting-based technique* (see e.g. [23, 20]), which allows one to focus on particularly promising regions of the search space, and to neglect the others. Details of this extension are described in section 4.

Moreover, in order to enhance usability, and to integrate our framework with powerful BP management tools already available in the literature, we are incorporating it as a set of plug-ins in the ProM framework [25]. ProM is an open source framework for process mining and process analysis. In ProM, once the process schema is learned, the incorporation of our facilities will help in the analysis of deviations from the process schema itself, which can be the input for a (formal) compliance verification, as observed above. Moreover, ProM offers process representation and visualization standards that can be helpful to improve usability and user-friendliness of our work. Finally, the implementation within ProM will make our work more visible, and available to the BP management scientific community. Technical details of the integration in ProM are presented in section 5.

Finally section 6 addresses some discussions and our concluding remarks.

2 A Framework for Supporting BP Monitoring

In our framework, trace retrieval relies on the *retrieval* step of Case-Based Reasoning (CBR) [1]. Indeed, CBR has been often resorted to in workflow systems with flexible adaptation capabilities, such as *agile workflow* systems [27]. CBR is in fact particularly well suited for managing exceptional situations, even when they cannot be foreseen or preplanned. Examples of CBR-based workflow tools can be found in e.g. [13–15]. In particular, in our work case retrieval is performed by a K-Nearest Neighbor technique, consisting in identifying the closest k cases (i.e. traces) with respect to an input one.

As for clustering, we resort to a hierarchical clustering technique, known as Unweighted Pair Group Method with Arithmetic Mean (UPGMA) [24]. UPGMA is typically applied in bioinformatics, where sequences of symbols (similar to our traces of actions) have to be compared. UPGMA also allows to build the phylogenetic tree of the obtained clusters, which can be resorted to for user-friendly visualization purposes, very useful in some domains (e.g. medical ones).

Case retrieval and clustering strongly rely on the notion of case, and on case distance definition. Technically, we define a *case* as a trace of execution of a given process schema. In particular, every trace is a sequence of actions, each one stored with its execution starting and ending times. Actions timestamps also allow to derive information about action durations and (partial) overlaps/delays between actions.

Case *distance* is then calculated on the basis of:

– atemporal information (i.e. action types);
– temporal information (i.e. action durations, qualitative and quantitative constraints between pairs of consecutive actions).

Operatively, we first take into account action types, by calculating a modified edit distance which we have called **trace edit distance**. In trace edit distance, the cost of a *substitution* is not always set to 1, as in the classical edit distance. We define it as a value $\in [0, 1]$ which depends on what action appears in a trace as a substitution of the corresponding action in the other trace. In particular, we organize actions in a *taxonomy*, on the basis of domain knowledge, and substitution penalty is set to the normalized number of arcs on the path between the two actions in the taxonomy [16]. *Insertions* do not always cost 1 as well. In fact, the insertion of one (or of a few) action(s) may sometimes introduce a (minor) change in a specific trace with respect to a reference trace, without changing the overall semantics/goals of the sequence of actions being considered. Our definition allows to capture this situation, by distinguishing between *indirections* (i.e. insertions of one or more actions within the trace, otherwise very similar to the reference one), and insertions in the head/tail portion of the trace, which becomes a superstring of the reference one. Recognizing such indirections can be very relevant for many BP monitoring applications (especially medical ones). Our definition introduces a knowledge-based parametrized weight $\in [0, 1]$, which depends on the action type. The final penalty of an indirection is set to 1 multiplied by the action weight. *Deletions* simply work dually with respect to insertions. Trace edit distance between two traces is finally defined as the total cost of a sequence of edit operations which transform one trace into the other. Formal definitions can be found in [16]. The minimization of trace edit distance provides the optimal alignment of the two traces.

Given the alignment, we can take into account temporal information. In particular, we compare the durations of aligned actions by means of a metric known as **interval distance** [18]. Interval distance calculates the normalized difference between the length of two intervals (representing action durations in this case).

Moreover, we are able to take into account the contribution between two pairs of corresponding actions on the traces being compared (e.g. actions A and B in trace P; the aligned actions A' and B' in trace Q). We quantify the distance between their qualitative relations (e.g. A and B overlap in trace P; A' meets B' in trace Q, see [3]), by resorting to a metric known as **neighbors-graph distance** [17]. If the neighbors-graph distance is 0, because the two pairs of actions share the same qualitative relation (e.g. A and B overlap in trace P; A' and B' also overlap in trace Q), we compare the quantitative constraints by properly applying interval distance (e.g. by calculating interval distance between the two overlap lengths).

These three contributions (i.e. minimal trace edit distance, interval distance between durations, neighbors-graph distance or interval distance between pairs of actions) are finally combined in an additive way.

A more formal description of our framework can be found in [16, 18, 17].

3 Experimental Results

We presented some retrieval and clustering experimental results in our previous works [16, 18, 17]. All experiments were conducted in the field of stroke management.

Since then, we collected more real world stroke management traces. In this paper, we report on additional clustering experiments, conducted on these new data.

In particular, in our experiments we aimed at verifying whether the distance function summarized in section 2 was able to overcome the performances of a previous version we introduced in [16], which did not take into account temporal information. The hypothesis we wished to test was the following: including temporal information allows to better characterize clusters, leading to a higher cluster *homogeneity*.

Moreover, we report on an additional experimental study, in which we tried to enhance the clustering results by introducing a pre-processing step, meant to separate traces belonging to patients with different feature values.

The database on which we made our experiments was composed of 377 traces collected at one of the largest stroke management units in the Lombardia region, Italy.

3.1 Comparing Two Distance Measures

As a first experiment, we compared the clustering results obtained by adopting two different distance measures: the one summarized in this paper, and a former version [16], which did not take into account temporal information.

Figure 1 shows part of the cluster hierarchies we obtained. We can observe that the content of the resulting clusters at the various levels of the hierarchies is very different in the two situations.

In particular, the hierarchy built using the distance measure which does not consider temporal information (see figure 1, upper part) is very unbalanced: every node is split into two children, one of which corresponds to a very big cluster (containing most of the traces of its parent node), while the other contains just a few traces. However, an inspection of cluster contents, made with the help of the domain experts working with us, revealed that small clusters were not originated from the ability to quickly isolate anomalous situations, and that their content is typically not *homogeneous* (see below).

On the other hand, the hierarchy built resorting to the distance measure described in this paper (see figure 1, lower part) appears to be much more balanced, and every node is normally split into two clusters of more comparable dimensions.

A different view of the clustering output is shown in figure 2, where the upper part shows the hierarchy of clusters, while the lower part details the content of one cluster (highlighted in the upper part of the figure itself). Identical actions in different traces are in the same color. In this way it is quite straightforward to compare traces by visual inspection. Temporal constraints are rendered as well. For instance, trace 2 shows an *overlaps* example, while trace 8 shows a *during* example. Durations and delays are straightforwardly interpretable as well.

We also studied the cluster contents, in order to verify cluster *homogeneity.*

Homogeneity is a widely used measure of the quality of the output of a clustering method (see e.g. [28, 22, 9]). A classical definition of cluster homogeneity is the following [28]:

$$H(C) = \frac{\sum_{x,y \in C}(1 - dist(x,y))}{\binom{|C|}{2}}$$

where $|C|$ is the number of elements in cluster C, and $1 - dist(x,y)$ is the similarity between any two elements x and y in C.

A (weighted) average of the homogeneity H of the individual clusters can then be calculated on (some of) the clusters obtained through the method at hand, in order to assess its quality. Average cluster homogeneity allows to compare the output of different clustering techniques on the same dataset (or the output obtained by differently setting up the same clustering technique, as we did by running UPGMA with two different distance measures).

An appropriate definition of $dist(x,y)$ is problem dependent [28]. In our domain, we exploited the normalized edit distance between pairs of traces. The choice of the edit distance allowed us to compare our more complex distance measures with a very classical metric, used as a common reference.

We computed the average of cluster homogeneity values level by level in the two hierarchies.

Clusters obtained by using the new metric had an average homogeneity of 0.41 at level 2 of the hierarchy, while with the metric in [16] they had an average homogeneity of 0.25. At level 3, the new metric led to an average homogeneity of 0.45, while the metric in [16] allowed to reach only a value of 0.28. Similar results were obtained if working at other intermediate levels of the hierarchy, with some clusters built using the old metric even reaching an average edit distance value of 0.88.

Such experiments show that the use of temporal information in the distance definition allows to obtain more homogeneous and compact cluster (i.e. able to aggregate closer examples) in the intermediate levels of the hierarchy, which is a desirable results and a meaningful outcome, especially in a domain in which the role of time is obviously central.

These results also confirm the outcomes we already obtained in our first experimental studies [18, 17], which were conducted on a smaller database.

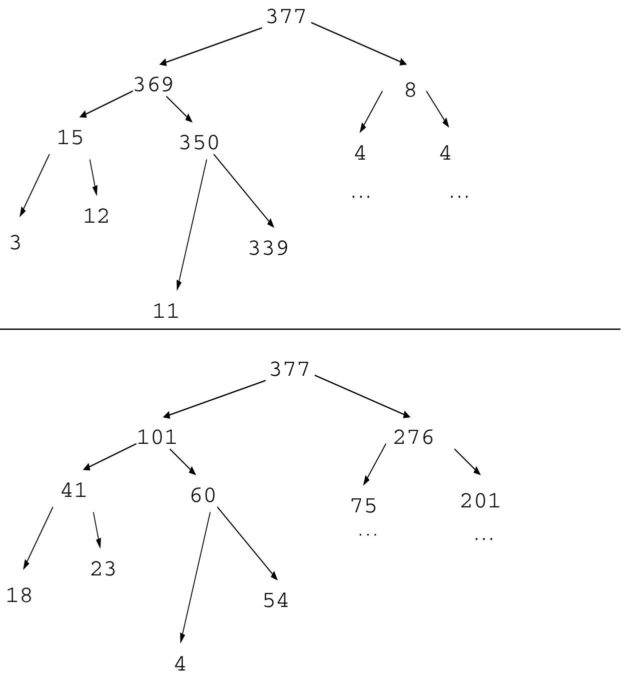


Fig. 1. Part of the cluster hierarchies obtained by applying the distance definition in [16] (top) and the one presented in this paper (bottom). Every node represents a cluster and reports the number of traces in the cluster itself.

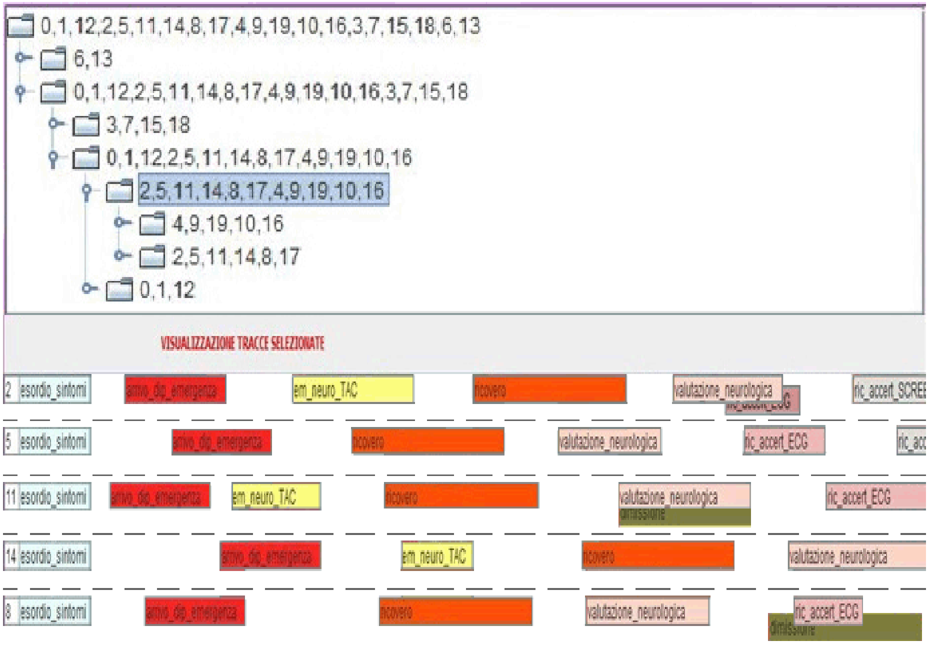


Fig. 2. A snapshot of clustering output as rendered by the system graphical interface

3.2 Verifying Care Delivery Similarity across Age Groups

Our second experiment was more oriented at the specific application needs, aiming at verifying the medical correctness of the stroke management procedures, in relation to patient age groups.

In particular, a recent cohort study [21] verified that, in the context of a province-wide coordinated stroke care system in Canada, stroke care delivery was similar across all age groups (even though, quite naturally, increasing age was associated with stroke severity and fatality). We aimed at verifying the absence of significant differences in stroke care delivery in different age groups in our database as well.

To this end, we implemented a pre-processing step, meant at separating traces belonging to patients with different anagraphical characteristics in different groups. In particular, we divided our database into 3 groups, filtering traces on the basis of the age of the patient they were applied to (i.e. ≤ 65, > 65 and ≤ 80, > 80). Traces in group ≤ 65 covered 25% of the total; traces in group > 80 covered 34%.

Observe that age is not explicitly recorded in traces, which just store the executed actions, and their temporal constraints. Indeed age is not used in our distance calculation - only in pre-processing.

We then performed intra-group clustering exploiting the distance measure summarized in this paper, and calculated cluster homogeneity.

Homogeneity values were comparable across the three classes, and comparable to the ones obtained on the whole database. At level 3 of the hierarchy, for instance, the average homogeneity was 0.43 in group ≤ 65, and 0.44 in group > 80 - not very different from the 0.45 value obtained on the entire database.

Such results highlight that intra-group treatments are not more homogeneous than inter-group ones; we believe that this can be seen as an indicator of the fact that care delivery is similar across all ages. In order to prove this statement, we plan to conduct a detailed evaluation of cluster contents with domain experts in the next months.

Along the same direction, in the future we will also pre-process our database by filtering traces on the basis of the National Institutes of Health Stroke Scale (NIHSS) values. The initial NIHSS score provides important prognostic information: approximately 60% to 70% of patients with an acute stroke and a baseline NIHSS score < 10 will have a favorable outcome after 1 year, as compared with only 4% to 16% of those with a score > 20 [2]. We plan to investigate if cluster homogeneity increases when applying a pre-processing step able to separate traces in groups corresponding to different NIHSS values: this could indicate that different procedures are applied to different groups of patients (which seems to be reasonable).

We will also evaluate whether to include these additional patient's features, which are not explicitly reported in traces, in the distance definition, in order to automatically include their treatment in the clustering process.

4 Improving Performances through a Pivoting-Based Technique

Retrieval time was very reasonable in the experiments we have conducted so far (see [16]). However, we were working on a small database. As the number of items in the database becomes higher and higher, the tool performances could progressively and significantly worsen. Nevertheless, computation time can be reduced, if a non-exhaustive search and distance calculation strategy is implemented.

The solution we are proposing is pivoting-based retrieval (PBR; see also [20, 23]).

The main idea in PBR consists in:

- computing the distance between a representative case and all the other cases (off-line);
- computing the distance between the representative case and the input case;
- estimating the distance between the input case and all the remaining cases by using triangle inequality, thus finding a lower and an upper bound for the distance value.

The intervals whose lower bound is higher than the minimum of all the upper bounds can be pruned (see figure 3). The following iterative procedure is then applied:

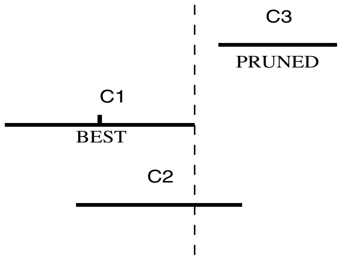


Fig. 3. Bound pruning in PBR

1. Initialization: $BESTp = \infty$ e $SOL = \{\ \}$
2. Choose the Pivot case as the minimum of the midpoints of the intervals; compute the distance between the input case and the Pivot ($DIST$); set $BEST = DIST$;
3. If $BESTp > BEST$ set $SOL = PIVOT$ and $BESTp = BEST$
4. Else if $BESTp = BEST$ set $SOL = \{PIVOT, SOL\}$
5. Prune the intervals whose lower bound is bigger than $BEST$, and remove the Pivot from the set of cases (see figure 3)
6. Back to step 2.

The choice of the most suitable representative case in the initialization phase is crucial to speed up the algorithm.

We have made some first tests by defining the representative case as the mean case, i.e. the one whose average dissimilarity to all the objects in the database is minimal. Other choices based on heuristics can be considered as well.

We have then implemented a more complex solution, in which we first cluster the available traces (resorting to the well-known K-Means algorithm [12]), and then select one representative case for each cluster (i.e. the cluster mean). Specifically, we perform a multi-step retrieval, in which:

– we identify the cluster the input case should be assigned to;
– we apply the PBR procedure described above to the cluster at hand (taking its mean as the initial representative case).

An extensive experimental work is foreseen in the next months, in order to test the advantages of PBR with respect to exhaustive search. Moreover, we plan to carefully evaluate the trade-off between the computational advantages of clustering-based early pruning, and the risk of loosing close neighbors of the input case, which belong to different clusters.

5 Incorporating the Tool in the ProM Framework

The ProM framework [25] is an open source environment, which offers a wide variety of process mining and analysis techniques. Process mining techniques [8] allow for extracting information recorded in traces of actions. Traces can be mined to discover models describing processes, organizations, and products. Moreover, it is possible to use process mining to monitor deviations and exceptions from the underlying process schema. Unlike classical data mining techniques, the focus is on processes and on questions that transcend the simple performance-related queries supported by classical Business Intelligence commercial tools.

ProM is platform independent as it is implemented in Java. It is easy to add new plug-ins to ProM, without the need to recode parts of the system. Moreover, ProM allows for the import from and the export to a wide variety of formats and systems, and provides advanced visualization and verification capabilities.

We are currently working at an implementation of our retrieval and clustering facilities as a pair of ProM plug-ins. Our plug-ins will be used to support an analysis of deviations from the mined process schema. In particular, they will be made available for cooperation with a set of verification plug-ins (Woflan analysis, verification of Linear Temporal Logic formulas on a log, check of conformance between a given process model and a log), and performance analysis plug-ins (basic statistical analysis and performance analysis with a given process model), already embedded in ProM. Through such interactions, our work will globally support a principled reengineering activity, in line with the objectives described in the Introduction.

In order to be completely up-to-date, we are integrating our work with the latest version of ProM, namely ProM 6.

To start, we have structured the plug-ins architecture as required by ProM 6, i.e. as a set of three separate modules:

1. a data import module;
2. an analysis module;
3. a data visualization module.

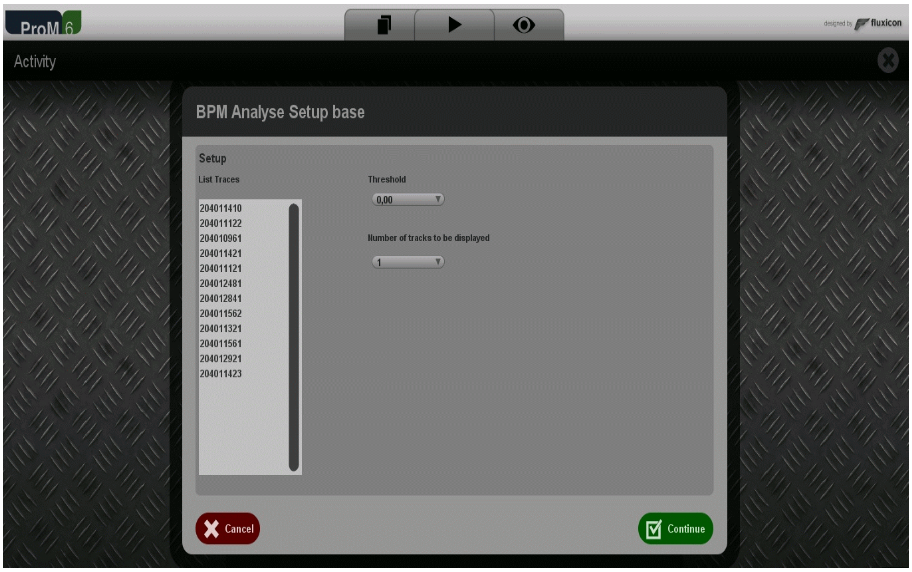
Data import (1) was actually already provided in ProM 6 (while no import facility was available in the previous version ProM 5.2); our work is correctly interfaced with such a default instrument.

As for module (2), according to ProM 6 specifics, in our architecture it is further subdivided into: (2-i) a graphical interface for parameter setting; and (2-ii) a core analysis module, which is devoted to perform retrieval (clustering, respectively), on the basis of the parameter values acquired through module (2-i).

As regards modules (2-i) and (3), in line with the advanced visualization capabilities implemented in ProM, we are realizing usable and user-friendly graphical interfaces.

As an example, figure 4 shows a snapshot of the retrieval parameter setting interface (module (2-i)).

On the other hand, figure 2 shows a snapshot of our clustering output (module (3)).



BPM Analyse Setup base

Setup

List Traces

204011410
204011122
204010961
204011421
204011121
204012481
204012841
204011562
204011321
204011561
204012921
204011423

Threshold: 0,00

Number of tracks to be displayed: 1

Fig. 4. A snapshot of retrieval parameter setting as rendered by the system graphical interface

With respect to version 5.2, ProM 6 is a little less consolidated, and less documentation is available. For this reason, our integration is requiring more time than what we originally foresaw; actually, the work is still in progress. In particular, our main ongoing effort is devoted to studying the requirements of the OpenXES library, the reference implementation of the XES standard for storing and managing event log data (http://www.xes-standard.org/openxes/start). As a matter of fact, we aim at being completely compliant with such requirements in traces management and elaboration (module (2)). Moreover, we are working at adapting our graphical modules (modules (3) and (2-i)), in order to make them correctly invokable within the ProM 6 environment. We plan to complete the implementation of our ProM 6 plug-ins in the near future.

A view of how our framework is meant to be used is reported in figure 5. Namely, as explained in the Introduction, end users and process engineers can exploit the tool in order to retrieve/cluster health care data. Clustering results, in particular, can highlight frequent anomalies, which could lead process engineers to identify problems in patient management, to be formally verified through other ProM functionalities. Moreover, process engineers may want to define a new version of the process schema, able to incorporate frequent changes. The data produced by following to the new schema will then be collected and analyzed as well, in a cyclical fashion.

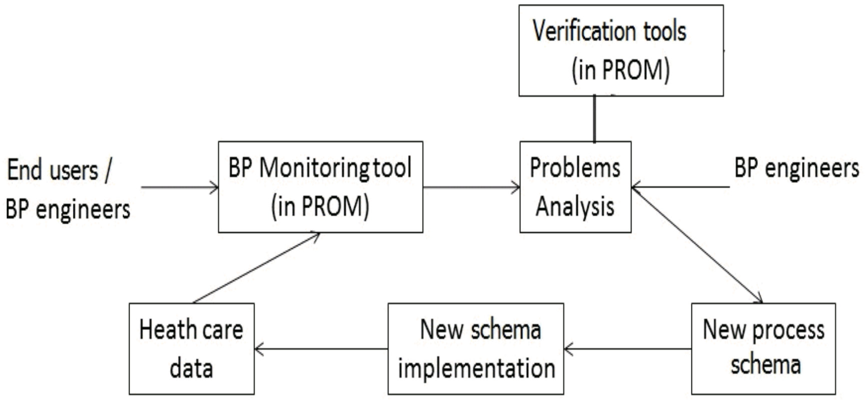


Fig. 5. A view of how the framework is meant to be exploited by end users and process engineers

6 Discussion and Conclusions

In this paper, we have described a case retrieval and clustering approach to BP monitoring, and we have introduced some recent research directions we are following, in order to improve our work.

Our main methodological contribution consists in the definition of a proper distance function, which is then resorted to both for retrieval and for clustering purposes, thus supporting end users as well as process engineers in their tasks.

In the literature a number of distance measure definitions for process instances exist. However, these definitions typically require further information in addition to the workflow structure, such as semantic annotations [26], or conversational knowledge. Such approaches are usually context-aware, that is, the contextual information is considered as a part of the similarity assessment of workflows. Unfortunately, any contextual information, as well as conversational knowledge, is not always available, especially when instances of process execution are recorded as traces of actions. Starting from this observation, a rather simple graph edit distance measure [6] has been proposed and adapted for similarity assessment in workflow change reuse [14]. Our approach somehow moves from the same graph edit distance definition. However, with respect to the work in [14], by focusing just on traces of execution we do not need to deal with control flow elements (such as alternatives and iterations). As a matter of fact, traces are always linear, i.e. they just admit the sequence control flow element. From this point of view, our approach is thus simpler. On the other hand, when focusing on linear traces our approach is more general and flexible. Indeed, we resort to taxonomic knowledge for comparing pairs of actions, so that two different actions do not always have a zero similarity. Additionally, we are able to recognize an indirect path from two actions, and to properly weight the degree of indirection in a parametrized way. Moreover, we have introduced a distance definition which also allows to properly compare action durations, and qualitative and quantitative constraints between actions. Such a capability is not provided at all in [14].

On the other hand, a treatment of temporal information in trace distance calculation has been proposed in [11]. Somehow similarly to our approach, the distance defined in that work combines a contribution related to action similarity, and a contribution related to delays between actions. As regards the temporal component, in particular, it relies on an interval definition which is very close to ours. Differently from what we do, however, the work in [11] always starts the comparison from the last two action in the traces: no search for the optimal action alignment is performed. Moreover, it stops the calculation if the distance between two actions/intervals exceeds a given threshold, while we always calculate the overall distance: as a matter of fact, even high distance values are resorted to by the clustering algorithm. Finally, the distance function in [11] does not exploit action duration, and does not rely on taxonomical information about actions, as we do. Moreover, the work in [11] does not deal with (partially) overlapping actions in the sequence: only *before* and *meets* Allen's operators [3] are treated. We thus believe that our approach is more general and potentially more flexible in practice.

Another contribution [7] addresses the problem of defining a similarity measure able to treat temporal information, and is specifically designed for clinical workflow traces. Interestingly, the authors consider qualitative temporal relations between matched pairs of actions, resorting to the neighbors-graph distance [10], as we do. However, in [7] the alignment problem is strongly simplified, as they only match actions with the same name. Our approach is thus an extension to their work.

It is also worth citing the approach in [5], which adapts edit distance to trace clustering, allowing to automatically derive the cost of edit operations. In such a work, however, temporal information is not considered. Moreover, clustering is mainly resorted to for improving process mining (by clustering instances in advance to process mining, the authors succeed in obtaining less "spaghetti like" process mining results). Process monitoring is not addressed.

As for our recent research lines, it is worth noting that the issue of avoiding exhaustive search in business process retrieval is quite a new research direction. An interesting paper addressing this topic is [4]. However, the approach followed in [4] is different from ours. First, in [4] the authors deal with graphs to represent their cases, while we resort on traces, that have a simpler structure. Second, the optimized search they propose, based on the A* algorithm, works on a partial mapping between the query case and the retrieved cases, where not all actions have been considered yet. On the other hand, the optimal (global) alignment between traces is automatically provided by our distance measure [16, 18]. Thus, their approach would not be directly applicable to our framework.

Our proposal to non-exhaustive search relies on a pivoting-based retrieval method. An extensive experimental work is foreseen in the next months, in order to test the advantages of PBR. In particular, we plan to carefully evaluate the trade-off between the computational advantages of clustering-based early pruning, and the risk of loosing close neighbors of the input case, which belong to different clusters.

Remarkably, our tool is also being incorporated in the ProM framework. The incorporation in ProM will allow our work to be used as a support to a principled reengineering activity, and will allow us to access plenty of new data, in order to complete our experimental work, in different application domains. Moreover, it will make the facility available to the BP management scientific community, thus enhancing the visibility of our CBR work.

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: foundational issues, methodological variations and systems approaches. AI Communications 7, 39–59 (1994)
2. Adams, H.P., Adams, R.J., Brott, T., del Zoppo, G.J., Furlan, A., Goldstein, L.B., Grubb, R.L., Higashida, R., Kidwell, C., Kwiatkowski, T.G., Marlerand, J.R., Hademenos, G.J.: Guidelines for the early management of patients with ischemic stroke. Stroke 34, 1056–1083 (2003)
3. Allen, J.F.: Towards a general theory of action and time. Artificial Intelligence 23, 123–154 (1984)
4. Bergmann, R., Gil, Y.: Retrieval of Semantic Workflows with Knowledge Intensive Similarity Measures. In: Ram, A., Wiratunga, N. (eds.) ICCBR 2011. LNCS, vol. 6880, pp. 17–31. Springer, Heidelberg (2011)
5. Jagadeesh Chandra Bose, R.P., Van der Aalst, W.: Context aware trace clustering: towards improving process mining results. In: Proc. SIAM Int. Conference on Data Mining, pp. 401–412. Springer (2000)
6. Bunke, H., Messmer, B.T.: Similarity Measures for Structured Representations. In: Wess, S., Richter, M., Althoff, K.-D. (eds.) EWCBR 1993. LNCS, vol. 837, pp. 106–118. Springer, Heidelberg (1994)
7. Combi, C., Gozzi, M., Oliboni, B., Juarez, J.M., Marin, R.: Temporal similarity measures for querying clinical workflows. Artificial Intelligence in Medicine 46, 37–54 (2009)
8. Van der Aalst, W., van Dongen, B., Herbst, J., Maruster, L., Schimm, G., Weijters, A.: Workflow mining: a survey of issues and approaches. Data and Knowledge Engineering 47, 237–267 (2003)
9. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification. Wiley-Interscience, New York (2001)
10. Freska, C.: Temporal reasoning based on semi-intervals. Artificial Intelligence 54, 199–227 (1992)
11. Kapetanakis, S., Petridis, M., Knight, B., Ma, J., Bacon, L.: A Case Based Reasoning Approach for the Monitoring of Business Workflows. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010. LNCS, vol. 6176, pp. 390–405. Springer, Heidelberg (2010)
12. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proc. 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press (1967)
13. Madhusudan, T., Zhao, J.L., Marshall, B.: A case-based reasoning framework for workflow model management. Data and Knowledge Engineering 50, 87–115 (2004)

14. Minor, M., Tartakovski, A., Schmalen, D., Bergmann, R.: Agile workflow technology and case-based change reuse for long-term processes. International Journal of Intelligent Information Technologies 4(1), 80–98 (2008)
15. Montani, S.: Prototype-based management of business process exception cases. Applied Intelligence (published online in February 2009), doi: 10.1007/s10489-009-0165-z
16. Montani, S., Leonardi, G.: A Case-Based Approach to Business Process Monitoring. In: Bramer, M. (ed.) IFIP AI 2010. IFIP AICT, vol. 331, pp. 101–110. Springer, Heidelberg (2010)
17. Montani, S., Leonardi, G.: Trace retrieval and clustering for business process monitoring. Technical Report TR-INF-2012-03-01-UNIPMN, University of Piemonte Orientale (2012) (submitted to Information Systems journal)
18. Montani, S., Leonardi, G., LoVetere, M.: Case retrieval and clustering for business process monitoring. In: Proc. Process-Oriented Case-Based Reasoning Workshop, International Conference on Case Based Reasoning (ICCBR 2011), Greenwich (2011)
19. Palmer, M., Wu, Z.: Verb Semantics for English-Chinese Translation. Machine Translation 10, 59–92 (1995)
20. Portinale, L., Torasso, P., Magro, D.: Selecting Most Adaptable Diagnostic Solutions Through Pivoting-Based Retrieval. In: Leake, D.B., Plaza, E. (eds.) ICCBR 1997. LNCS, vol. 1266, pp. 393–402. Springer, Heidelberg (1997)
21. Saposnik, G., Black, S.E., Hakim, A., Jiming, F., Tu, J.V., Kapral, M.K.: Age disparities in stroke quality of care and delivery of health services. Stroke 40, 3328–3335 (2009)
22. Sharan, R., Shamir, R.: CLICK: A clustering algorithm for gene expression analysis. In: Proc. International Conference on Intelligent Systems for Molecular Biology, pp. 260–268 (2000)
23. Socorro, R., Mico, L., Oncina, J.: A fast pivot-based indexing algorithm for metric spaces. Pattern Recognition Letters 32, 1511–1516 (2011)
24. Sokal, R., Michener, C.: A statistical method for evaluating systematic relationships. University of Kansas Science Bulletin 38, 1409–1438 (1958)
25. van Dongen, B., Alves De Medeiros, A., Verbeek, H., Weijters, A., Van der Aalst, W.: The proM framework: a new era in process mining tool support. In: Ciardo, G., Darondeau, P. (eds.) Knowledge Mangement and its Integrative Elements, pp. 444–454. Springer, Heidelberg (2005)
26. van Elst, L., Aschoff, F.R., Berbardi, A., Maus, H., Schwarz, S.: Weakly-structured workflows for knowledge-intensive tasks: An experimental evaluation. In: Proc. of 12th IEEE International Workshops on Enabling Technologies (WETICE), Infrastructure for Collaborative Enterprises, pp. 340–345. IEEE Computer Society, Los Alamitos (2003)
27. Weber, B., Wild, W.: Towards the Agile Management of Business Processes. In: Althoff, K.-D., Dengel, A.R., Bergmann, R., Nick, M., Roth-Berghofer, T.R. (eds.) WM 2005. LNCS (LNAI), vol. 3782, pp. 409–419. Springer, Heidelberg (2005)
28. Yip, A.M., Chan, T.F., Mathew, T.P.: A Scale Dependent Model for Clustering by Optimization of Homogeneity and Separation. CAM Technical Report 03-37. Department of Mathematics, University of California, Los Angeles (2003)

A Case-Based Approach to Cross Domain Sentiment Classification

Bruno Ohana, Sarah Jane Delany, and Brendan Tierney

Dublin Institute of Technology, Kevin Street, Dublin 8, Ireland
bruno.ohana@student.dit.ie, {sarahjane.delany,brendan.tierney}@dit.ie

Abstract. This paper considers the task of sentiment classification of subjective text across many domains, in particular on scenarios where no in-domain data is available. Motivated by the more general applicability of such methods, we propose an extensible approach to sentiment classification that leverages sentiment lexicons and out-of-domain data to build a case-based system where solutions to past cases are reused to predict the sentiment of new documents from an unknown domain. In our approach the case representation uses a set of features based on document statistics, while the case solution stores sentiment lexicons employed on past predictions allowing for later retrieval and reuse on similar documents. The case-based nature of our approach also allows for future improvements since new lexicons and classification methods can be added to the case base as they become available. On a cross domain experiment our method has shown robust results when compared to a baseline single-lexicon classifier where the lexicon has to be pre-selected for the domain in question.

Keywords: case-based reasoning, sentiment classification, sentiment lexicons.

1 Introduction

Subjective text reflecting people's opinions is now widely available in online forums, product comparison sites, in social media websites and many other user-generated content outlets. Such repositories can provide a new valuable layer of sentiment information to business intelligence applications, and devising computational methods for their efficient use is the realm of sentiment analysis research. In particular, for a given piece of text, one can ask whether its sentiment can be considered generally positive or negative, favorable or unfavorable. Sentiment classification is the area of sentiment analysis concerned with predictive methods for determining the sentiment orientation of subjective text. It is often characterized as a binary classification problem where possible outcomes are positive or negative sentiment. Alternatively a sentiment classifier may attempt to place sentiment along a range of possible values, including neutral, as is the case for example on a star-rating system of film reviews.

Supervised learning methods for sentiment classification have been extensively studied in the past decade having achieved considerable success and have been applied to various domains [1–3]. One downside of such methods however is the requirement for labelled in-domain data for training. Considering the wide range of potential areas sentiment classification can be applied to - product and content reviews on every possible industry for example - the cost of compiling data sets for each target domain becomes prohibitively expensive. Moreover, it is known that sentiment clues derived from supervised learning models are strongly associated with the domain used in training, and not easily reusable on a different domain [4]. This has encouraged research on cross domain approaches that minimize the requirement for in-domain training data.

Current research in cross domain sentiment classification methods have focused broadly on two approaches: one is to use out-of-domain data to build supervised learning models capable of performing well on other domains. The second class of methods essentially amounts to an unsupervised approach where documents are evaluated with the assistance of pre-existing knowledge, requiring no training data. The unsupervised sentiment classifier takes into account a document's linguistic clues, and often relies on a sentiment lexicon: a language resource that associates terms with sentiment information. For example, a lexicon would encode a priori knowledge of sentiment contained on words such as excellent, good or terrible. With a sentiment lexicon, an unsupervised classifier can make predictions by identifying opinion-bearing terms in a target document and making decisions according to the overall orientation of terms found.

Large numbers of sentiment lexicons are available in the literature, and various unsupervised techniques for using them in sentiment classification have been proposed [5–7]. However, the choice of lexicon and how it is going to be used is generally fixed in advance in a classification problem. We claim that in a multi domain setting there is no single fixed lexicon technique that will consistently generate good predictions for all documents. Instead, a more flexible approach would be to determine, out of all the available lexicons, which ones obtain good predictions for specific documents, and use those documents to build a case base of past examples for later reuse.

In this study we propose a case-based approach to document sentiment classification where out-of-domain labelled data is reused to make predictions on unseen documents as follows: a case base is built by evaluating labelled documents using various sentiment lexicons, and recording which lexicons yield correct predictions as the case solution. A case is represented by a set of features reflecting the structure and statistics which describe the document. A prediction is made by retrieving the most similar cases and reusing the lexicons found in the cases' solutions. This paper shows how a case-based approach can produce results comparable to single-lexicon unsupervised approaches while removing the need to determine the best lexicon in advance. In addition, the case-based approach is easily extensible, allowing for the addition of new cases where new sentiment lexicons and ways of applying them can be incorporated in the future as they become available.

In the remaining sections we discuss the research literature of sentiment classification, lexicons and the challenges of cross domain sentiment classification. We describe the design of our case base, and present the results of a sentiment classification experiment using product reviews from different domains. We discuss the results obtained with our approach and propose avenues for future work.

2 Related Work

Supervised learning methods for sentiment classification using in domain training data have been extensively studied in the past decade: early work from Pang, Lee and Vaithyanathan [1] presents the results of different classifiers using features based on word n-grams on a data set of film reviews. In [8] this model is improved by eliminating objective sections from raw documents prior to training, while a similar approach seen in [9] builds multiple classifiers based on types of sentences found in a document. Extending the feature sets with document statistics and punctuation information is seen in [2].

The performance of supervised learning methods is strongly linked to the domain data used during training. Experiments seen in [10] and [4] illustrate how poor results can be obtained on combinations where domains used for training and evaluation have little in common. In the latter study more general methods to overcome this drawback suggest using out-of-domain data to build classifier ensembles, and extending training data with in-domain unlabelled documents. The use of small amounts of labelled and unlabelled in-domain data is also seen in [11, 12].

Sentiment lexicons are language resources that associate a vocabulary term with opinion polarity – positive, negative or neutral, often by means of a numeric score indicating certainty or opinion strength. Lexicons can be obtained via manual annotation of words, the *General Enquirer* being a well known example [13], however to overcome the limitations in size and cost of manual annotation, research has sought ways of creating lexicons by expanding a small set of seed terms using a pre-existing knowledge resource. Corpus based methods are first seen in the work of [14] where expansion is based on terms found near connectors such as "and", "or" and "but". Term proximity is also explored using larger corpora in [15] and more recently in a lexicon derived from web documents in [16]. Expansion via thesaurus relationships is explored in lexicons presented in [17]. The *SentiWordNet* lexicon [18] uses the WordNet database [19] as the source of information and is built first by exploring direct term relationships such as synonym and antonym information, and then performing a second step that uses a semi supervised method for detecting sentiment from term glosses. The *SentiFul* lexicon [20] finds new words via morphological rules that relate them to a word with known sentiment.

When considering their use in sentiment classification, sentiment lexicons appear as an additional source of information for engineering features on cross domain classifiers as seen in the use of SentiWordNet [21]. Alternatively sentiment lexicons are typically used in unsupervised approaches in conjunction with an algorithm for scanning a document and extracting a document sentiment score based on lexicon information and linguistic clues. Multi domain classification has been explored using custom built lexicons [6, 5] and using multiple lexicons and a majority based scheme to obtain predictions [7]. Unsupervised methods based on lexicons have the advantage of requiring no training data. However, before applying them to a classification problem, the choice of lexicon and how they are to be applied needs to be determined and fixed. This may lead to sub optimal results where better suited combinations exist for a yet unseen target domain.

Case-based reasoning [22] is a problem solving approach aiming at the reuse of similar past examples to determine the outcome of a new unseen instance. To date, we found little evidence of this approach being used in sentiment classification. One example can be found in the literature in [10] where a document repository of labelled cases is indexed for keyword based searches. Predictions are made by first retrieving cases using a free-text search based on terms found in a target document. The sentiment of a target document is determined by the labels of similar documents and their rank in search results.

3 The Case-Based Approach

Cases in our case base are derived from a training set of labelled out-of-domain opinionated documents, and have two essential components: the *case description* is a document signature used for later retrieval, while the *case solution* stores details about successful predictions obtained for the document during training. The case description attempts to broadly capture a document's characteristics into a set of features, leaving aside any potential domain specific aspects such as particular term presence. To that end we propose the use of an n-dimensional feature vector derived from document, sentence and term-level statistics, part of speech information, punctuation and other indicators of document complexity. These are described in Table 1.

Table 1. Features describing a case

Category	**Metrics**
Document Statistics	Total words, tokens and sentences. Average sentence size. Part of speech frequencies.
Writing Style	Spacing Ratio. Stop words ratio. Average syllable count. Monosyllable ratio. Word to token ratio. Ratio of unique words.

The spacing ratio is the rate of empty spaces to characters, and the ratio of stop words is based on the SMART system stop word list [23]. There are 17 features in total, and all features are numeric and normalized with min-max normalization based upon values from the training set.

For the case solution we record all the sentiment lexicons that made a correct prediction during training on the document represented by the case. We use 5 different lexicons from the literature: The General Inquirer (GI) lexicon [13], a small manually annotated lexicon often used as a gold standard on sentiment analysis research; The Subjectivity Clues lexicon (Clues) [24] is also an annotated lexicon that includes words from the General Inquirer and other sources; SentiWordNet (SWN) [18] is an automatically built lexicon based on WordNet term relationships and gloss information. The Moby lexicon [7] is built from the Moby thesaurus by expanding from a set of seed terms. Finally the MSOL lexicon [17] is based on the expansion of the Macquaire thesaurus from a set of word pairs with opposing sentiment.

To make predictions we use an unsupervised classifier that takes a sentiment lexicon as input and computes a document sentiment *score* by querying the lexicon for sentiment information of terms found in the document. When a term is found in the lexicon, its sentiment orientation is retrieved. By convention, this is stored as a pair (pos, neg) of numeric values indicating positive and negative sentiment. We note that some lexicons such as SentiWordNet may record values for both positive and negative sentiment on a single term due to the process by which the scores are derived. The overall sentiment score of a document is also a pair of numeric values containing the accumulated scores of all individual terms obtained from the lexicon, and a prediction of document sentiment is based on the higher of the two values in the document score.

The unsupervised classifier works with the help of linguistic clues to improve its accuracy: a document part-of-speech tagger marks the grammatical role of each word in the document. To process documents in our experiment we use the *Stanford POS Tagger*[1]. Based on results from preliminary experiments, we select only terms tagged as adjectives and verbs during document scoring. In addition, certain lexicons are segmented by part of speech, thus a part-of-speech tagging pre-processing step also improves the accuracy of lexicon queries.

Negation detection is also another important element when detecting sentiment, for example, the sentences "I think this book is not good" and "I think this book is good" have opposing sentiment orientation, despite the presence of the largely positive term "good" on both. We apply a variation of the *NegEx* algorithm [25] to identify opinion in negated sentences. Our implementation works by scanning a document for known negating n-grams and inverting sentiment orientation of nearby terms in the same sentence.

Finally, when calculating document sentiment from the sum of individual term scores, we want to reduce the effect of terms frequently occurring in English from having a dominant effect on the predictions. To mitigate this effect we implement an adjustment factor based on relative term frequencies introduced by [7]. The adjustment is given by the formula:

$$S_{adj}(w) = S(w) * (1 - \sqrt{freq(w)}) \tag{1}$$

[1] http://nlp.stanford.edu/software/tagger.shtml

where $s(w)$ is the term score from the source lexicon and $freq(w)$ is the term frequency (valued between 0 and 1) of w relative to the most frequent term found in the lexicon. The relative frequencies are calculated for each lexicon according to term frequency information from a separate document corpus.

3.1 Populating the Case Base

The case base is built from a set of out-of-domain labelled documents. For each training document, we attempt to make predictions using each of the available sentiment lexicons and the unsupervised classifier described previously. If no lexicons can correctly predict a document, the document is discarded. Otherwise a new case is added where the case solution is the list of all lexicons that yielded a correct prediction. Algorithm 1 describes our method for populating the case base at training time.

Algorithm 1. Populating the case base

```
Input:
  – D, set of labelled out-of-domain documents for training.
  – L, set of all available sentiment lexicons.
  – f(L,d), unsupervised document sentiment classifier using lexicon $L$ as input.
Output:
  – CB, the populated case base.
$CB \leftarrow \{\}$
for all document d in D do
  $S \leftarrow \{\}$
  for all $L_i$ in L do
    make prediction using $f(L_i, d)$
    if prediction is correct then
      $S \leftarrow S \cup L_i$
    end if
  end for
  if $S <> \{\}$ then
    compute case description $x(d)$
    $CB \leftarrow CB \cup (x(d), S)$
  end if
end for
```

We note that our approach allows for future expansion since the case base can grow iteratively as more out-of-domain data becomes available, while additional lexicons can be obtained and added to the algorithm in the future.

3.2 Case Retrieval and Reuse

Using Euclidean distance as the similarity measure, we retrieve the k cases nearest to an unseen instance. The solutions from the k-nearest cases are used as follows: where *k=1* we can directly apply the obtained lexicons from the case solution to make a prediction. For larger values of k we establish a ranking of lexicons by counting their frequency of occurrence out of the k cases retrieved. By inspecting the ranking, we then obtain the most frequently found lexicons and use them on predictions, as illustrated in the example in Table 2.

Table 2. Example ranking of solutions using kNN and *k=3*

Retrieved Solutions (k=3)	**Ranking (Count)**	**Lexicons**
case A: $\{L_1, L_2, L_5\}$	L_1 (3)	$\{L_1, L_2\}$
case B: $\{L_1, L_2\}$	L_2 (3)	
case C: $\{L_1, L_2, L_3\}$	L_3 (1)	
	L_5 (1)	

The outcome of retrieval and ranking may yield more than a single lexicon since a case solution may record multiple lexicons, and ties can occur. In this case we separately calculate the document sentiment scores using each lexicon and the unsupervised classifier, and aggregate all positive and negative scores. The accumulated scores are then used to make a prediction as before: the highest of the scores determines document sentiment.

4 Evaluation

We evaluate our proposed case-based approach on a cross domain sentiment classification experiment using 6 datasets of user generated product and film reviews in plain text: the IMDB dataset of film reviews [1], the hotel reviews dataset from [26], and product reviews for apparel, music, books and electronics from Amazon.com [27]. Each data set has an equal number of positive and negative documents and is detailed in Table 3.

Table 3. Customer review datasets

Dataset	**No. of Reviews**	**Avg. Size (tokens)**	
		positive	**negative**
Film	2000	803.2	721.4
Hotels	2874	215.0	228.7
Electronics	2072	237.6	194.9
Books	2034	284.7	202.2
Apparel	566	137.1	110.5
Music	5902	246.4	195.6

4.1 Experiment Methodology

To assess our method for sentiment classification using out-of-domain data we created 6 distinct case bases by training on datasets of all but one of the domains.

The case base is then used to classify documents on the hold out domain. The composition of each case base in terms of class distribution is presented in Table 4 along with the percentage of discarded documents. The case base names reflect the hold out dataset.

Table 4. Case base class distribution and discarded ratio

Case base	**Size**	**Pos. %**	**Neg. %**	**Discarded %**
Books	9683	53.3	46.7	27.8
Electronics	9592	53.6	46.4	28.2
Film	9614	54.1	45.9	28.6
Music	6173	52.6	47.4	25.1
Hotels	11516	53.5	46.5	7.8
Apparel	11002	53.4	46.6	28.9

All of the case bases contain more documents with a positive rather than negative orientation, with negative documents being excluded more often during the case base population stage. This could be attributed to a relative difficulty in predicting negative sentiment compared to positive ones using lexicon based term scoring techniques - a behavior also noted on past research [6]. We also see a considerable number of discarded cases at training time, for which a correct classification could not be found using any of the available lexicons. The ratio of discarded entries stays around 25-28% on most case bases but shows a distinct lower ratio when the hotels dataset is left out, indicating this dataset makes a substantial contribution to the total of discarded cases.

Figure 1 shows, for each case base, the distribution of cases by the number of lexicons used in the solution. We note that across all case bases a total of 27-28% of cases were predicted correctly using just one or two of the available lexicons.

Cases with a single-lexicon solution reflect the situation where no other lexicon could make a correct prediction while building the case base. The average distribution of these cases over all case bases is given below to illustrate how a specific lexicon can sometimes be uniquely capable of making correct predictions on certain documents: GI: 0.9%; SWN: 2.4%; Clues: 0.7%; Moby: 7.7%; MSOL: 2.4%.

4.2 Results

In Table 5, we present accuracy results for the case-based method using the six hold out data sets and their corresponding case bases. For comparison we include baseline results from best single-lexicon accuracies using a similar scoring approach and the same data sets [7]. We marked in bold the experiment results that outperformed the single-lexicon baseline.

When compared to the best single-lexicon baseline, our results at *k=1* gave improved accuracies in four of the six domains tested. Using the Wilcoxon signed-ranks test for comparing results over multiple data sets [28] at confidence *p=0.05*,

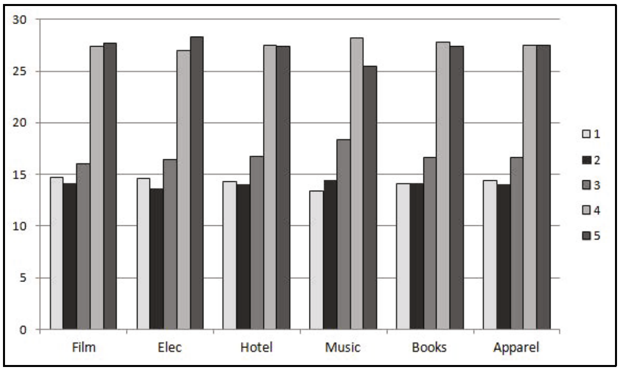


Fig. 1. Distribution of cases by number of lexicons in solution (% of case base size)

Table 5. Accuracy on the hold out data set for each case base

Case base	**k=1**	**k=3**	**k=5**	**Baseline Accuracy**	**Lexicon**
Film	67.88	66.73	66.78	68.18	Clues
Electronics	**68.06**	65.16	65.6	67.19	SWN
Hotel	**72.58**	71.43	70.7	71.67	SWN
Music	64.62	64.78	64.28	65.04	SWN
Books	**64.27**	62.54	62.0	63.73	Clues
Apparel	**66.96**	66.07	63.42	65.54	Clues

we find the results for the case-based method and the baseline are not statistically significant. We note however that the baseline presents best accuracies from separate single-lexicon experiments run on each domain, and that the lexicon yielding best results is not necessarily the same on all experiments. Thus, our approach produces results similar to the baseline, but without the requirement for fixing a lexicon in advance.

4.3 Ranking Behavior

For different values of k, there is a slight reduction in performance as k grows and we begin applying the solution ranking algorithm. This trend can be seen in accuracies plotted on Figure 2 (note that the y axis is partial).

At the same time, we see a distinct trend in the number of lexicons used in making a prediction as k grows: At $k = 1$ no solution ranking takes place and all lexicons obtained from the solution of the single nearest case are used, while for higher values of k ranking selects the most frequent lexicons out of all solutions retrieved. Figure 3 shows how the number of lexicons used in a prediction are distributed as k changes for the film review data set. For higher values of k ranking tends to favour predictions with fewer lexicons, and to rely less on aggregated predictions from many lexicons.

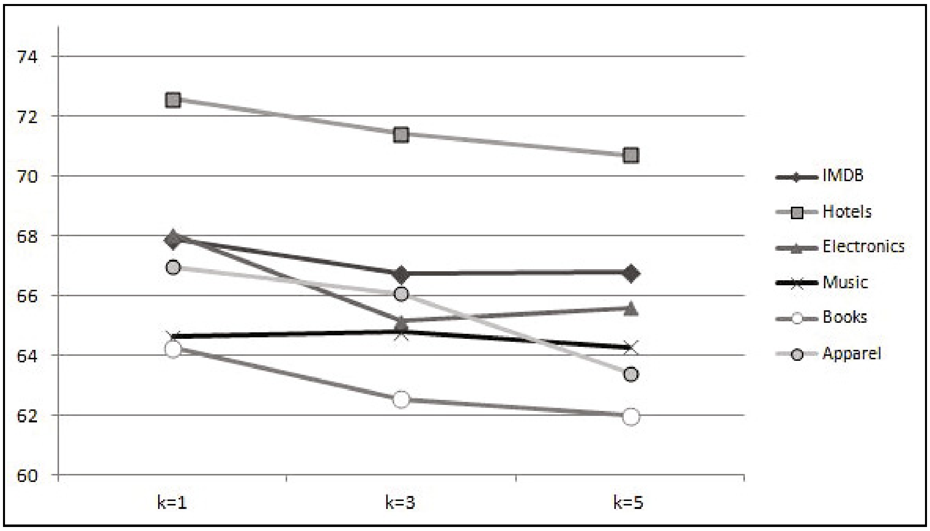


Fig. 2. Accuracies for varying k

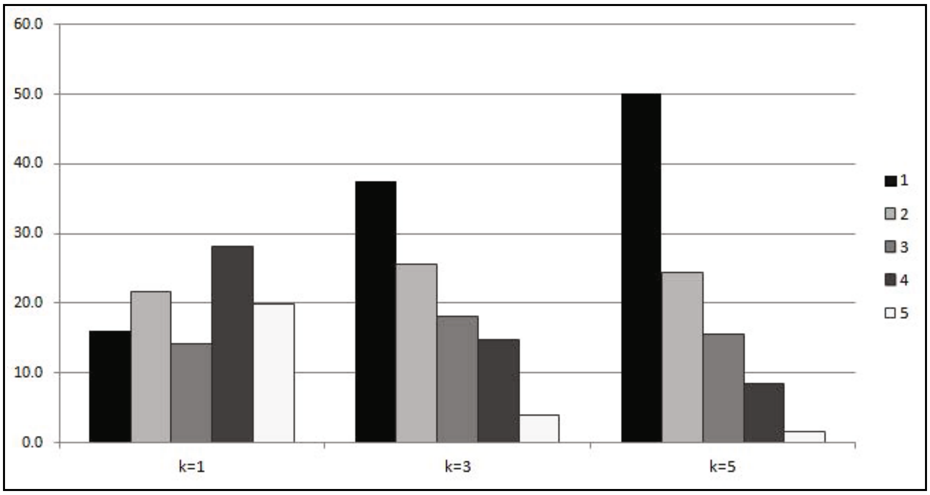


Fig. 3. Distribution of total number of lexicons used in prediction for varying k (film reviews case base)

5 Conclusions and Future Work

In this study we present an case-based approach for performing sentiment classification. The case description is a feature vector based on document statistics, and the case solution contains all lexicons that made correct predictions during training. W e evaluated our approach on user generated reviews from six different domains. When compared to a baseline using the best result from a single-lexicon classifier, our results remain competitive with no statistically significant performance difference while producing more robust results by eliminating the need to fix a lexicon prior to making predictions. This illustrates the potential of case-based methods as an important component in unsupervised sentiment classification.

We see the following areas as interesting paths to further improve the case-based sentiment classifier: in this study we have restricted the solution search space to five sentiment lexicons while populating the case base, and saw that a considerable number of cases were discarded as no correct predictions could be found. Our approach can be easily be extended to consider more lexicons during the case base population stage, and this can help reducing the discard ratio by recording more cases for later reuse. Additional unsupervised algorithms can be added during training in a similar way.

Case reuse in our experiment relies on a ranking step to select the most frequently found lexicons from all solutions retrieved. The predictions of each selected lexicon are then aggregated to determine document sentiment. We have seen that, as the number of retrieved solutions grows ($k > 1$), ranking tends to favour single lexicon predictions causing a slight deterioration on performance. Experimenting with other ranking methods that benefit from aggregating predictions from many lexicons while still being able to produce scalable results for large search spaces is an interesting problem in developing better case-based approaches.

Finally, easily adding new cases from additional training data is a beneficial aspect of case-based methods. Investigating how this can be best achieved by using case base maintenance policies for addition and deletion is also an interesting path for future research, in particular when considering large volumes of out-of-domain training data with uneven label and domain distribution.

References

1. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of 2002 EMNLP, pp. 79–86. ACL (2002)
2. Dave, K., Lawrence, S., Pennock, D.M.: Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In: Proceedings of the 12th WWW, p. 528. ACM (2003)
3. Abbasi, A., Chen, H., Salem, A.: Sentiment analysis in multiple languages: Feature selection for opinion classification in Web forums. ACM Transactions on Information Systems (TOIS) 26(3), 12 (2008)
4. Aue, A., Gamon, M.: Customizing sentiment classifiers to new domains: a case study. In: Proceedings of Recent Advances in Natural Language Processing (RANLP) (2005)
5. Kennedy, A., Inkpen, D.: Sentiment classification of movie reviews using contextual valence shifters. Computational Intelligence 22(2), 110–125 (2006)
6. Taboada, M., Voll, K., Brooke, J.: Extracting sentiment as a function of discourse structure and topicality. Technical Report TR 2008-20, School of Computing Science, Simon Fraser University, Burnaby, BC, Canada (2008)

7. Ohana, B., Tierney, B., Delany, S.J.: Domain Independent Sentiment Classification with Many Lexicons. In: Mining and the Web Workshop, AINA 2011, pp. 632–637. IEEE (2011)
8. Pang, B., Lee, L.: A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of the 42nd Annual Meeting of the ACL, p. 271. ACL (2004)
9. Li, S., Huang, C.R., Zhou, G., Lee, S.Y.M.: Employing personal/impersonal views in supervised and semi-supervised sentiment classification. In: 48th Annual Meeting of the ACL, pp. 414–423. ACL (2010)
10. Sood, S., Owsley, S., Hammond, K., Birnbaum, L.: Reasoning through search: A novel approach to sentiment classification. Technical Report NWU-EECS-07-05, Northwestern University (2007)
11. Tan, S., Wu, G., Tang, H., Cheng, X.: A novel scheme for domain-transfer problem in the context of sentiment analysis. In: Proceedings of the Sixteenth CIKM, pp. 979–982. ACM (2007)
12. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In: Annual Meeting ACL, vol. 45, p. 440 (2007)
13. Stone, P.J., Dunphy, D.C., Smith, M.S., Ogilvie, D.M., et al.: The general inquirer: A computer approach to content analysis. MIT Press, Cambridge (1966)
14. Hatzivassiloglou, V., McKeown, K.R.: Predicting the semantic orientation of adjectives. In: 35th Annual Meeting of the ACL, ACL 1998, pp. 174–181. Association for Computational Linguistics (1997)
15. Turney, P.D., Littman, M.L.: Unsupervised learning of semantic orientation from a hundred-billion-word corpus. Technical Report ERB-1094, National Research Council of Canada (2002)
16. Velikovich, L., Blair-Goldensohn, S., Hannan, K., McDonald, R.: The viability of web-derived polarity lexicons. In: HLT 2010: Annual Conference of the North American Chapter of the ACL, pp. 777–785. ACL (2010)
17. Mohammad, S., Dunne, C., Dorr, B.: Generating high-coverage semantic orientation lexicons from overtly marked words and a thesaurus. In: Proceedings of EMNLP 2009, pp. 599–608. ACL (2009)
18. Esuli, A., Sebastiani, F.: SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining. In: LREC 2006, pp. 417–422 (2006)
19. Miller, G.A.: WordNet: a lexical database for English. Communications of the ACM 38(11), 41 (1995)
20. Neviarouskaya, A., Prendinger, H., Ishizuka, M.: SentiFul: A Lexicon for Sentiment Analysis. IEEE Transactions on Affective Computing (99), 1 (2011)
21. Denecke, K.: Are SentiWordNet scores suited for multi-domain sentiment classification?. In: ICDIM 2009, pp. 1–6. IEEE (2009)
22. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications 7(1), 39–59 (1994)
23. Salton, G., Lesk, M.: The smart automatic document retrieval systems an illustration. Communications of the ACM 8(6), 391–398 (1965)
24. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis. In: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT/EMNLP 2005, p. 354. ACL (2005)
25. Chapman, W., Bridewell, W., Hanbury, P., Cooper, G., Buchanan, B.: A simple algorithm for identifying negated findings and diseases in discharge summaries. Journal of Biomedical Informatics 34(5), 301–310 (2001)

26. Baccianella, S., Esuli, A., Sebastiani, F.: Multi-facet Rating of Product Reviews. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 461–472. Springer, Heidelberg (2009)
27. Jindal, N., Liu, B.: Opinion spam and analysis. In: Proceedings of the Conference on Web Search and Web Data Mining (WSDM 2008), pp. 219–230. ACM (2008)
28. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research 7, 1–30 (2006)

GENA: A Case-Based Approach to the Generation of Audio-Visual Narratives

Santiago Ontañón[3], Josep Lluís Arcos[1], Josep Puyol-Gruart[1], Eusebio Carasusán[2], Daniel Giribet[2], David de la Cruz[1], Ismel Brito[1], and Carlos Lopez del Toro[1]

[1] IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra (Spain)
{arcos,puyol,davdela,ismel,clopez}@iiia.csic.es
[2] Televisió de Catalunya
Carrer de Jacint Verdaguer,
Sant Joan Despí, 08970 (Spain)
{ecarasusan.q,dgiribet.g}@tv3.cat
[3] Computer Science Department
Drexel University
Philadelphia, PA, 19104 (USA)
santi@cs.drexel.edu

Abstract. This paper presents GENA, a case-based reasoning system capable of generating audio-visual narratives by drawing from previously annotated content. Broadcast networks spend a large amount of resources in covering many events and many different types of audiences. However, it is not reasonable for them to cover smaller events or audiences, for which the cost would be greater than the potential benefits. For that reason, it is interesting to design systems that could automatically generate summaries, or personalized news shows for these smaller events or audiences. GENA was designed in collaboration with *Televisió de Catalunya* (the public Catalan broadcaster) precisely to address this problem. This paper describes GENA, and the techniques that were designed to address the complexities of the problem of generating audio-visual narrative. We also present an experimental evaluation in the domain of sports.

1 Introduction

Broadcast networks spend a large amount of resources in covering many events such as soccer matches or Formula 1 races, and personalizing content for many different types of audiences. In order to cover such events, editors spend lots of hours performing repetitive tasks, such as annotating and selecting video segments to create event reports. Additionally, broadcast networks also have to create many specific reports and news shows for different regions, which include local highlights amongst the general interest ones. The work presented in this paper aims at automating some of those tasks, in order to bring down the cost, and freeing editors from mechanical tasks, allowing them to focus in the more creative ones.

The GENA (Generation of Audio-visual NArrative) system was designed in collaboration with *Televisió de Catalunya* (TVC) (the public Catalan broadcast network) precisely to address this problem. Specifically, GENA was designed to generate different kinds of sport event summaries, and localized news shows, although the experiments reported in this paper only cover the sports domains. GENA generates an audio-visual narrative, which in this context means content that can be broadcast through a TV channel, IPTV, or the web.

In order to address this problem GENA uses a CBR approach. TVC kindly provided us with a collection of audio-visual narratives (soccer game summaries, Formula 1 reports, news shows) generated by professional reporters, together with the complete original assets from where the narratives were generated. Each of these narratives was captured as a case. For example, in the case of soccer summaries, a case consists of the complete original soccer game (complete audio-visual content plus metadata), a description of the type of summary desired for the game, and the actual summary. Given a new request to generate a summary, GENA retrieves similar cases of summaries generated by professional reporters, and generates a new candidate summary ready to be directly broadcast.

In this paper, we present GENA, including the knowledge representation used in order to capture audio-visual narratives, and the specific retrieval and adaptation procedures used to deal with such complex data. From an application point of view, GENA is a novel CBR solution to a real life problem. From a theoretical point of view the main contributions of GENA are a new similarity measure specially designed for narrative, which is a generalization of the Jaccard similarity [9], and a new adaptation approach based on generating solutions that exhibit similar statistical properties as the solutions in the retrieved cases.

The remainder of this paper is organized as follows. Section 2 describes in detail the specific problem that GENA tries to address. Then, Section 3 describes the GENA system, including retrieval and adaptation. Section 4 reports our experimental results in the domains of Soccer, and Formula 1. Finally, Section 5 compares GENA with existing work in the literature.

2 Audio-Visual Narrative Generation

The main goal behind GENA is to generate audio-visual narratives, like sport summaries, or personalized news shows, from the existing data in the repositories of a broadcast network, while requiring minimal additional manual intervention. For that reason, in this section we will first describe the structure of the data already available in the repositories of a broadcast network, and then formulate the specific problem that GENA was designed to solve.

2.1 Available Data

Information in the repositories of Broadcast networks is stored as a series of layers, some of them containing the original audio-visual form (videos of sport

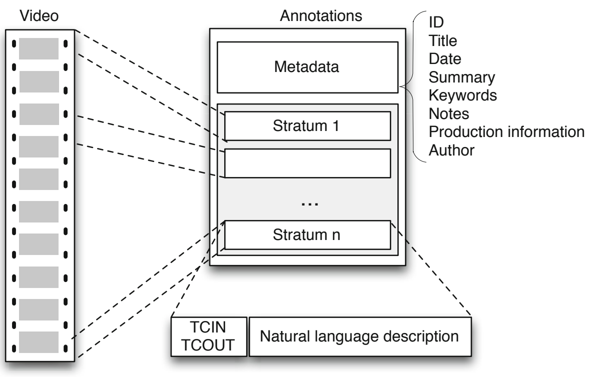


Fig. 1. Structure of the available information for sport events

events, news clips, etc.), and some others contain annotations like keywords, natural language descriptions and other metadata in order to make the information available when reporters need to look for it. When a reporter or an editor wants to create, for example, a report on the best moments of a given sport event, they query the repositories of information using the annotations (such as keywords) to access the audio-visual information, and then they splice parts of the audio-visual content together in order to form a good summary. This is precisely the task GENA is designed to perform.

Figure 1 shows the structure of the available information from which GENA needs to generate audio-visual narratives. Specifically, we have applied GENA to two different domains: sport events (soccer and Formula 1) and news shows, but this paper only reports experiments concerning the sports domain. Figure 1 shows the structure of the information that TVC stores for sport events, as composed of two main parts: video and annotations. The current version of GENA is not equipped with any video processing reasoning capabilities (although a video processing module is being worked on). Thus, all the reasoning performed by GENA is done at the level of the annotations.

In the case of sports, Figure 1 shows that the video of a sport event is annotated with two main kinds of information: metadata and *strata* list. The metadata contain general information such as the ID, title, date and keywords of the video. The strata information is the most useful for GENA and consists of a list of individual records, called strata (commonly also known as "Time Segment Annotations"). Each of these strata contains a natural language description of a fragment of video (specified by a start time, TCIN, and an end time, TCOUT). For example, a typical stratum in the soccer domain could contain the sentence: "Leo Messi scores a goal from midfield", and correspond to some seconds of video that capture the moment in which Leo Messi scored a goal. The list of strata can be used by GENA and by the human editors to identify and search over the information contained in the video. Moreover, the list of strata might not be sorted, and strata can totally or partial overlap. For example, there might be a stratum labelled “Replay of the best moments of the first half of the game”, and each of those individual moments will be described by their respective strata.

Specifically, during the development of the GENA project, TVC provided (only for the specific purposes of this project) 18 soccer games (from the 2009 - 2010 Spanish national league) and the whole set of Formula 1 races from the 2010 championship. In the soccer domain, TVC also made available all the different summaries that had been generated from those games (long summaries, short summaries, best goals, etc.). In the Formula 1 domain, we had general summaries as well as summaries focused on specific drivers.

2.2 Problem Statement

The problem that GENA is designed to solve is the following: given the existing data from sport events and news stories described in the previous section, is it possible to create an artificial intelligence system that can generate different types of sport summaries or personalized news shows automatically?

In order to address that problem, we decided to use a case-based reasoning approach, and captured the available information in the form of cases. A case in GENA captures an example of how did a human expert generate a summary or a news show from a given original asset (from a soccer game, Formula 1 race, etc.). Therefore, a case in GENA is composed of the following parts:

- Problem Description:
  - *Original asset*: the original soccer game or Formula 1 race
  - *Generation parameters*: what kind of narrative had to be generated from the original asset (a summary, a report of a specific player/driver, a summary for a specific audience, etc.).
- Solution:
  The *Generated narrative*, represented as the subset of strata that were selected from the original asset, and the particular order in which they were sequenced.

The generation parameters depend on the domain. For example, in soccer, they consist of a “summary type” from the following list: long summary, short summary, first part summary, second part summary, goals, and summary focused on a specific player. In case the summary type is ”summary focused on a specific player”, then there is a second parameter specifying which player to focus on. Analogously, the Formula 1 domain has similar narrative types.

Moreover, notice that this case representation makes the assumption that human reporters and editors generate summaries or news shows by just selecting and sequencing strata from the original assets. In reality, humans tend to sometimes trim the strata (remove some parts of it), or extend them (add additional video footage not in the strata) in order to achieve desired effects. Since GENA cannot reason about the content in the video, this part was left out as future work. However, the strata granularity level is enough to generate interesting and useful summaries, and news shows to be broadcast directly.

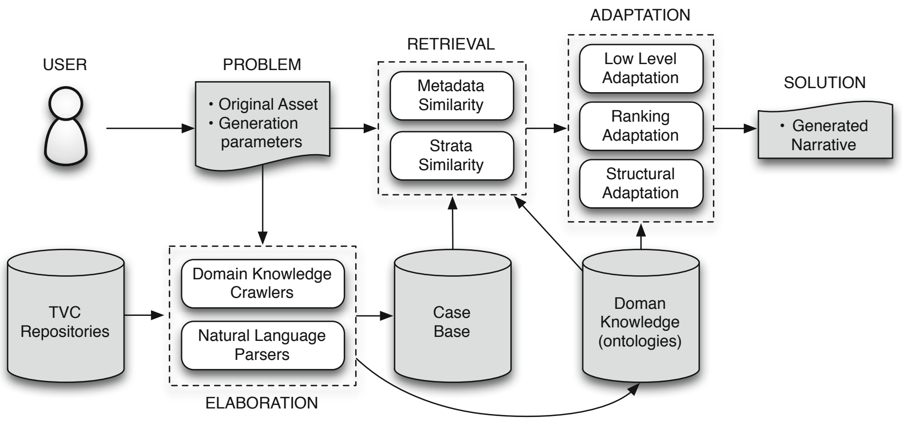


Fig. 2. High-level view of the GENA narrative generation architecture

GENA contains a case-base with 59 soccer cases (59 different summaries generated from the 18 soccer games we had available), and 47 Formula 1 cases (47 different summaries generated from 15 races).

Problems contain only two parts: an original asset, and the generation parameters. The goal of GENA upon receiving a problem is to generate a narrative from the original asset constrained by the generation parameters. An example problem would be:

– Original asset: soccer game "Barça - Mallorca (4-2)".
– Generation parameters: "summary focused on a specific player", "Leo Messi" (meaning that GENA needs to generate a summary focusing on Leo Messi).

Upon receiving a problem, GENA would retrieve one or more cases containing similar types of summaries from the soccer domain, and then generate a narrative of the "Barça - Mallorca (4-2)" game in the same way as the narrative was generated in the retrieved case. The next section describes the retrieval and adaptation techniques we designed for the GENA system.

3 GENA

Figure 2 shows a high level overview of the GENA system architecture, with its three main components: elaboration, retrieval and adaptation. The role of the elaboration module is to 1) process the data in the broadcast network repositories (TVC repositories) and turn them into cases and additional domain knowledge (such as ontologies for the different domains), and 2) process problems coming from the user so GENA can work with them (e.g. perform any kind of natural language analysis needed). Once problems have been elaborated, the retrieval module retrieves one or several cases that are relevant to the problem at hand by combining several similarity measures. Finally, the adaptation module reuses the solution(s) in the retrieved cases to generate the desired narrative (e.g. a summary of a soccer game). We have implemented three different adaptation modules of increasing complexity, which will be compared in Section 4.

The following subsections describe each one of these three modules in detail.

3.1 Elaboration

GENA contains two main explicit knowledge containers: the case-base and the domain knowledge repository. The former is a plain list of cases, and the second one contains a collection of ontologies for the different domains GENA can deal with. In particular, in the version reported in this paper, GENA has three ontologies: a generic one, one for soccer, and one for Formula 1.

The *elaboration* module preprocesses the data coming into GENA to feed these two knowledge containers, and so that retrieval and adaptation can be performer effectively. Therefore, the elaboration module has two main goals:

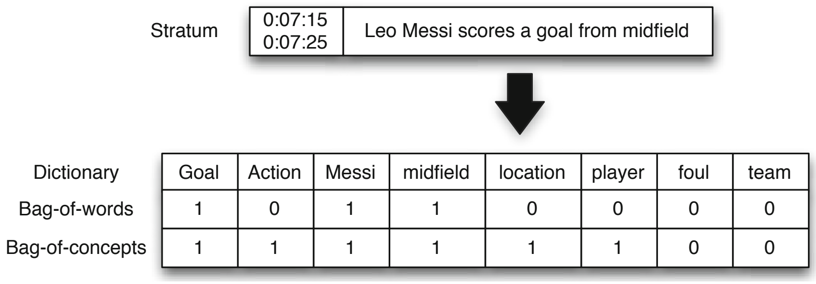
– Transform all the available knowledge into a unified representation so GENA can reason about it (domain knowledge). This goal is mainly achieved by the *domain knowledge crawlers* in GENA.
– Transform all the suitable available knowledge into episodic knowledge (problems and cases). In order to achieve this goal, the most complex process is to perform natural language processing in the strata descriptions of the assets.

Given that the data available in the broadcast network repositories was through for human consumption, rather than for being used by an artificial intelligence system, there are no formal domain ontologies available for GENA to use. However, there are glossaries of terms and thesauri used by the documentarists in the network to classify all the assets, containing thousands of terms in a semi-structured way. The first domain knowledge crawler in GENA contrasts these lists of terms with the words appearing in the natural language description of the strata in the available assets. Those terms appearing with enough frequency in the assets, are terms that are likely to be useful to GENA. Thus, the crawler compiles a list of the useful terms for each of the three domains GENA was applied to. Specifically, the resulting ontologies for Soccer and Formula 1 contain 390 and 85 unique (i.e. non synonymous) concepts respectively.

In order to organize the set of terms obtained by the previous crawler, we defined a generic ontology with the basic concepts of narrative (story, discourse, existents, events, actions, happenings, characters and props) as identified by Chatman [2], and later semi-automatically classified all the terms obtained by the first crawler into the concepts of this generic ontology.

A final crawler mined information about the different teams, soccer players, Formula 1 drivers, cities, countries, politicians, etc. available in the different data repositories of TVC in order to populate the resulting ontologies.

Once the knowledge crawlers have populated all the ontologies in GENA, the next step in the elaboration process is to process the soccer games, Formula 1 races and news shows available in order to create cases. The most complex part



Dictionary	Goal	Action	Messi	midfield	location	player	foul	team
Bag-of-words	1	0	1	1	0	0	0	0
Bag-of-concepts	1	1	1	1	1	1	0	0

Fig. 3. Illustration of the BoW and BoC generation for the description of a stratum

of this process is to turn the natural language descriptions in the strata into some computer-understandable representation that GENA can reason with. In particular, we opted for a *bag-of-concepts* (BOC) [11] representation.

Given a predefined dictionary of words, the standard *bag-of-words* (BOW) [4] representation, represents a stratum as a vector with one position per word in the dictionary. Each position is 1 or 0 depending on whether the corresponding word appears in the stratum or not. A bag-of-concepts representation extends this representation in a simple but significant way: rather than having a dictionary of words, the dictionary contains *concepts* organized in an *is-a* hierarchy. If a given concept c appears in a stratum, then, not only c, but all the super-concepts of c are also added to the bag-of-concepts. This is illustrated in Figure 3, where the bag-of-words and bag-of-concepts for a stratum with the description "Leo Messi scores a goal from midfield" using a small dictionary are shown.

We used the FreeLing [7] natural language parser to analyze the natural language text from each stratum. We then cross the result of the analysis with the available ontologies in order to determine which of the concepts in the ontologies of GENA is present in each stratum, and build the bag-of-concepts of each stratum. Once the BOC of each stratum in the cases is generated, they are added to the case-base of GENA.

Each time a new problem arrives, the same natural language analysis is performed, before GENA attempts to solve the problem.

3.2 Retrieval

Retrieval in GENA works as a 2 step process: In the first *filtering* process, a subset of candidate cases of the case-base is selected as those cases that belong to the appropriate domain (soccer, Formula 1 or news) are annotated with the same generation parameters as the problem (e.g. same kind of summary). Then, in a second step, GENA follows a standard k-nearest neighbor algorithm using a specially designed similarity measure to retrieve the k most similar cases to the problem at hand. This section focuses on the similarity measure used for the second step of the retrieval process.

The most basic similarity measure used in GENA is similarity between two strata, for which we experimented with several options. In the experiments reported in this paper, we defined the similarity between two strata, a and b as the *cosine* similarity between their two bag-of-concepts, $BOC(a)$, and $BOC(b)$:

$$S_{cos}(a,b) = cos(\theta) = \frac{BOC(a) \cdot BOC(b)}{|BOC(a)||BOC(b)|}$$

For example, consider the two strata: $a =$ "Leo Messi scores a goal from midfield" and $b =$ "Messi scores a goal". We compute their bag-of-concepts using the dictionary shown in Figure 3, and we obtain $BOC(a) = (1,1,1,1,1,1,0,0)$, $BOC(b) = (1,1,1,0,0,1,0,0)$. Their cosine similarity is:

$$S_{cos}(a,b) = \frac{(1,1,1,1,1,1,0,0) \cdot (1,1,1,0,0,1,0,0)}{|(1,1,1,1,1,1,0,0)||(1,1,1,0,0,1,0,0)|} = \frac{4}{\sqrt{6}\sqrt{4}} = 0.816$$

GENA can use the strata similarity to assess similarity between complete narratives (e.g. between soccer matches, F1 races, etc.). Since the problem of comparing complete narratives is very complex, GENA uses two simplification assumptions in order to make case retrieval feasible: a) narratives can be represented as sets of strata, b) an approximation of the similarity measure will be computed, rather than the exact measure.

The first assumption is that narratives can be represented as sets of strata, i.e. GENA ignores the order in which the strata are sequenced in a narrative for the purposes of case retrieval (since order is important in the formation of narratives, the order is indeed taken into account during adaptation, but not during retrieval). Similarity between sets of elements can be approximated using the Jaccard similarity [9], which estimates the similarity between two sets A and B as follows:

$$S_{Jaccard}(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

The Jaccard similarity returns 1 when the two sets are identical, and 0 when they are disjoint. In general, the larger the intersection, the higher the similarity. Moreover, in the case of GENA, each strata is practically unique[1], and thus, the intersection between the sets of strata of two narratives is likely to be empty. For that reason, we defined a generalization of the Jaccard similarity, as follows (assuming $|A| < |B|$):

$$S_{GJ}(A,B) = \max_{m \in M} \frac{\sum_{a \in A} S_{cos}(a, m(a))}{|A| + |B| - \sum_{a \in A} S_{cos}(a, m(a))}$$

[1] Journalists annotate events in natural language plus some common tags, it is very unlikely that two strata have exactly the same tags and natural language annotation.

Intuitively, this new measure works as follows. Assume m is an injective mapping from the elements in A to the elements in B, so that if $a \in A$ then $m(a) \in B$. The numerator is the sum of similarities of the elements in A with their corresponding elements in B (which is an approximation of their intersection), and the denominator is just the sum of elements from A and B minus the approximation of their intersection. In this way, the numerator is bounded between 0 and $|A|$, and the denominator between $|B|$ and $|A| + |B|$, since we assumed that $|A| \leq |B|$, the similarity is always in the interval $[0, 1]$. Moreover, the more similar the strata in A to their corresponding strata in B, the higher the similarity. The final step is to select the mapping m that maximizes this similarity from the set M of all possible injective mappings from A to B.

It is easy to prove that if the strata similarity metric used is the identity function, the previous measure corresponds exactly to the Jaccard similarity, and thus, the proposed measure is a generalization of it. Moreover, since computing all the possible mappings is an expensive operation, we will only approximate it by the following measure:

$$S_{SGJ}(A, B) = \frac{\sum_{a \in A} \max_{b \in B} S_{cos}(a, b)}{|A| + |B| - \sum_{a \in A} \max_{b \in B} S_{cos}(a, b)}$$

This measure has a polynomial time complexity and is the one used for case retrieval in GENA.

Finally, using the previously defined narrative similarity measure, GENA defines the similarity between a problem and a case as follows. A problem defines an original asset from which we want to generate a narrative, and a case contains both an asset and a generated narrative. Thus, there are two similarities that can be assessed:

1. The similarity between the asset in the problem and the original asset in the case: which gives as a measure of how similar was the problem solved in the case to the problem at hand.
2. The similarity between the asset in the problem and the generated narrative in the case: which gives us a measure of how similar are the strata that were selected to form the target narrative in the case to the ones in the problem at hand. In other words, this gives us a measure of how easy would it be to generate a target narrative similar to the one in the case using the strata in the problem at hand, i.e. this is a measure of adaptability.

Following ideas from *adaptation-guided retrieval* [12], GENA combines the previous two similarities to obtain a final score for each case. Thus, given a problem with original asset A, and a case with an original asset C and generated narrative G, the score the GENA assigns to each case is defined as:

$$S(A, (C, G) = \alpha \times S_{SGJ}(A, C) + (1 - \alpha) \times S_{SGJ}(A, G)$$

where α is a constant that can take values in the interval $[0, 1]$.

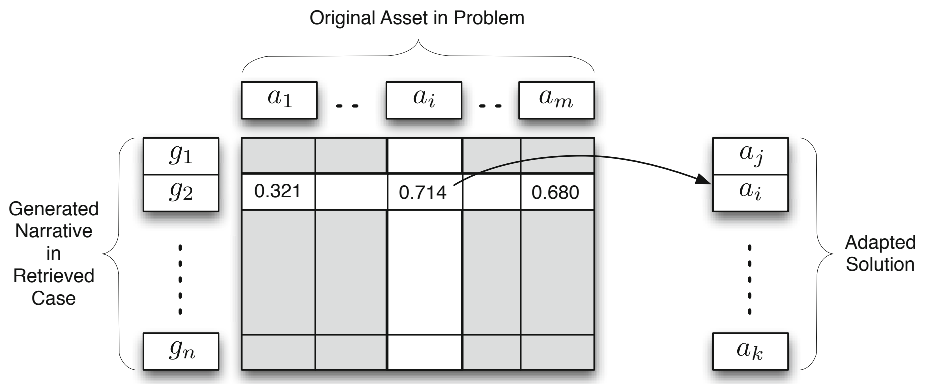


Fig. 4. Illustration of the *low-level-adaptation* module, where the numbers in the matrix represent the similarities between the strata in the generated narrative in the retrieved case and the strata in the original asset in the problem

3.3 Adaptation

GENA implements three different adaptation modules, of increasing complexity: the *low-level adaptation* (LLA) module, the *ranking adaptation* (RA) module, and the *structural adaptation* (SA) module. Let us describe each one of them in detail.

The low-level adaptation module, or LLA, adapts the solution from a single retrieved case in the following way. The solution in a case is an ordered list of strata: $[s_1, ..., s_n]$. Now, let us call $m(s_i) = a$ to the most similar stratum a in the problem at hand to the strata s_i. The solution generated by the LLA module is: $[m(s_1), ..., m(s_n)]$. In other words, the LLA generates a solution for the problem at hand, by taking the solution in the retrieved case and replacing each strata by the most similar one in the problem at hand. This idea is illustrated in Figure 4, where it can be see that the LLA just needs to compute a similarity matrix with the similarities between all the strata in the solution in the retrieved case and all the strata in the problem at hand.

The ranking adaptation module, or RA, performs a similar process to the LLA, but considering a set of k retrieved cases, rather than a single retrieved case. Given a set of cases $C_1, ..., C_k$, the RA proceeds as follows:

1. Let $S_1, ..., S_k$ be the solutions that the LLA would generate from each of the cases in $C_1, ..., C_k$ respectively.
2. Let the score of a stratum $score(a) = |\{S \in S_1, ..., S_k | a \in S\}|$ be the number of solutions from $S_1, ..., S_k$ in which a given strata a from the problem at hand appears.
3. The RA computes N as the average size of $S_1, ..., S_k$ (in number of strata), and selects the N strata from the problem at hand that have highest score.
4. The set of N strata selected in the previous step are ordered according to their average positions in $S_1, ..., S_k$ in order to form the final solution S.

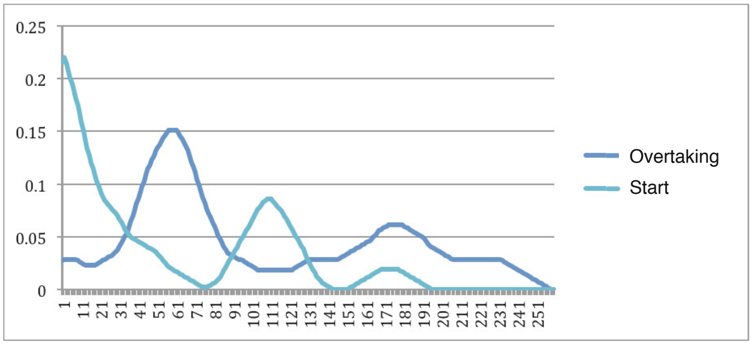


Fig. 5. Illustration of the the temporal distribution statistical indicator computed by the TDC module for the two concepts *overtaking* and *start* in the Formula 1 domain. The vertical axis represents probability of a stratum with the concept to appear, and the horizontal axis represents time.

The advantage of the RA module is that it is less brittle than the LLA to exceptional events, or to problems that require parts from more than one case. If the problem has some exceptional event, it is unlikely that the retrieved case contains it, but more likely that at least one case in the ranking has it.

Finally, the structural adaptation module, or SA, is the most complex of the adaptation modules in GENA and consists of three main processes (for the sake of space, we only provide a high-level view of this module):

1. A *Term Relevance* module assesses a relevance factor (a real number in the interval $[0, 1]$) for each concept in the ontology, by measuring how important is the fact that a given concept appears in a stratum for the strata to be selected as part of the solution of the cases with the same generation parameters as the problem at hand. For example, soccer concepts such as *goal* have a high relevance, whereas terms, like *midfield* are not very relevant.
2. A *Target Distribution Computation* (TDC) module, computes, for each concept in the ontology, a collection of statistical indicators that all cases with the given generation parameters satisfy. In particular, for each concept, it computes three indicators: percentage of all the strata with the concept that appear in the solution, percentage of all the strata not belonging to a replay and with the concept that appear in the solution, and temporal distribution of strata in the solution with a given concept. This last indicator is illustrated in Figure 5, where we can see the distribution of strata containing two given concepts in summaries of Formula 1 races. In this example, we can see that the indicator represents that in Formula 1 summaries, it is more common to have the strata that talk about the start of the race at the beginning, and that immediately after we should have strata that talk about overtaking moves.
3. Finally, the SA module generates a solution as follows: it generates an initial solution by using LLA or RA. Then, using a hill-climbing search process, it

modifies this solution by adding/removing/reordering strata trying to maximize the fit of the solution to the statistical indicators computed by the TDC module. When the solution cannot be transformed in any further way to improve the fit to the statistical indicators, it is returned to the user.

The advantages of the structural adaptation approach over the low-level or ranking adaptation approaches can be seen with this simple example. Imagine that a user asks GENA to generate the summary of a soccer game that had 5 goals. Imagine GENA retrieves a case that contained only 3 goals. Since goals in summaries are only shown once, the solution generated by the low-level adaptation will only contain three strata with goals, and will not show all 5 goals in the problem at hand. However, the structural adaptation module would realize (through the statistical indicators) that all the strata marked as goal in the original asset were selected for the solution. Therefore, it will include all 5 goals from the problem at hand in the final solution.

4 Experimental Evaluation

In order to evaluate the performance of GENA we performed a *leave-one-out* evaluation in two domains: Soccer and Formula 1. The case-bases used for the two domains had 59 and 47 cases respectively. Different cases contain different types of generation parameters (whole game summaries, summaries of the first part, reports of the performance of a specific player or driver, etc.). For each case in the leave-one-out evaluation, we asked GENA to generate a narrative and compared it against the narrative contained in the case, that had been authored by a professional editor or journalist.

In order to compare the output of GENA with the human-generated narrative, we used the following metrics:

- The Jaccard similarity, as described in Section 3.2, which measures the proportion of strata that GENA selected that were also selected by the human professional.
- Since the Jaccard similarity penalizes severely the fact that GENA might not select the exact strata the expert did, but one that is very similar, we also used the proposed generalization of the Jaccard similarity S_{SGJ}, as described in Section 3.2, which measures how similar are the strata selected by GENA to those selected by the human professional.
- A measure of whether GENA sequences the strata in the same order as the human does, defined as:

$$O(S,S') = \frac{\sum_{a \in C} \sum_{b \in (C \setminus a)} o(a,b,S,S')}{|C|(|C|-1)}$$

  where $C = S \cap S'$, and $o(a,b,S,S') = 1$ when a and b appear in the same order in S and S' (i.e. if a appears before b in S, they also must appear in that order in S', regardless of any other strata that can be in between them), and 0 otherwise.

Table 1. Experimental results in the soccer domain. All measures are normalized between 0 and 1, and higher is better.

	$\alpha = 1$			$\alpha = 0$		
Adaptation	$S_{Jaccard}$	S_{SGJ}	O	$S_{Jaccard}$	S_{SGJ}	O
Low-Level Adaptation	0.105	0.860	0.480	0.067	0.846	0.382
Ranking Adaptation	0.157	0.935	0.467	0.155	0.932	0.455
Structural Adaptation	0.147	0.906	0.550	0.143	0.896	0.541

Table 2. Experimental results in the Formula 1 domain. All measures are normalized between 0 and 1, and higher is better.

	$\alpha = 1$			$\alpha = 0$		
Adaptation	$S_{Jaccard}$	S_{SGJ}	O	$S_{Jaccard}$	S_{SGJ}	O
Low-Level Adaptation	0.107	0.883	0.233	0.079	0.884	0.175
Ranking Adaptation	0.197	0.947	0.404	0.198	0.949	0.355
Structural Adaptation	0.145	0.913	0.420	0.140	0.899	0.343

Moreover, we report results with $\alpha = 1$ (the retrieval module only considers the similarity between the problem at hand and the original asset in the case) and with $\alpha = 0$ (the retrieval module only considers the similarity between the problem at hand and the generated narrative in the case), experiments with intermediate values of the α parameter are part of our future work.

Tables 1 and 2 show the results for Soccer and Formula 1 respectively. Considering soccer, the first thing we observe is that the Jaccard similarity severely penalizes GENA for not selecting exactly the same strata that the expert selected. However, as S_{SGJ} shows, the strata selected by GENA are very similar to the ones selected by the human expert. For example, with $\alpha = 1$, the average similarity between the strata of the output of GENA and those of the narrative by the human expert is 0.935 using the ranking adaptation method. Thus, we can conclude that GENA selects strata that are very similar to those selected by a human expert. Considering the order in which GENA sequences the strata, we see that the O measure varies from 0.382 using low-level adaptation and $\alpha = 0$ and up to 0.550 with structural adaptation and $\alpha = 1$. This means that about half of the strata are ordered in the same way as those generated by the human expert. We can also see that the choice of the adaptation procedure has a very strong impact in all measures, with both ranking adaptation and structural adaptation obtaining better results than the low-level adaptation measure. Concerning Formula 1, Table 2 shows similar trends as for soccer.

In summary, GENA can generate narratives that contain strata that are very similar to those generated by human experts. The order of the strata is highly improved by the complex adaptation procedures used in GENA, but improving it even further is part of our future work. Additionally, an evaluation of the output of GENA by human experts, manually inspecting each answer produced by GENA is also ongoing.

5 Related Work

Three areas of work are relevant for the work presented in this paper: narrative generation, textual CBR and AI applications to sports and news domains.

CBR approaches to narrative generation date back to the Minstrel system [13]. Minstrel generated King Arthur styled narratives by retrieving and adapting past story snippets. A key difference between Minstrel and GENA is that Minstrel's goal was to generate original and creative narratives, whereas in GENA, the goal is to generate summaries, or personalized news. Therefore, the content in narratives generated by GENA is selected from an original asset, while in Minstrel it is generated by adapting the stories form the retrieved cases. Similar to Minstrel, other more recent work like Mexica [8] and Riu [6] also use CBR to generate narratives, but none of them focus on generating narratives by selecting and sequencing content from an original asset, as GENA does.

GENA's case retrieval mechanism is related to work on textual CBR [3], where typically, the goal is to retrieve textual documents from a case-base that are relevant or similar to a given query document. This has been explored in depth in the CBR community; representative examples are the CR2N [1] system for identification of reusable pieces of text, and the work on the jCOLIBRI system [10] for generic textual case retrieval.

Finally, there is a recent interest in artificial intelligence applications to generate sport event reports or personalized news. For example, the *News at Seven* [5] system generates personalized news given a set of user preferences. The main difference between GENA and News at Seven, is that GENA generates content exclusively based on the input audio-visual asset in the problem at hand, while News at Seven starts with a set of preferences and crawls the Internet for relevant material. For example, News at Seven would not be capable of performing GENA's task of, given a Formula 1 race, generate a summary of the race with the best moments sequenced in the appropriate way.

6 Conclusions and Future Work

This paper has presented GENA, a case-based reasoning system capable of generating audio-visual narratives by drawing from previously annotated content generated by human experts. We described a new similarity measure for narratives composed of lists of strata and a collection of adaptation procedures to adapt narratives. GENA was evaluated in two different domains: soccer and Formula 1, demonstrating the generality of the approach.

Our experimental results show that GENA succeeds in generating narratives that are similar to those generated by human experts in that they contain strata that are very similar. However, there is still room for improvement in the order in which these strata are sequenced by GENA. For example, GENA currently doesn't fully exploit semantic relations between entities and actions.

As part of our future work, we are currently applying GENA to the news shows domain, where the main challenge is a very large vocabulary and the requirement of better natural language processing. Additionally, we are also exploring new forms of adaptation that help GENA in generating narratives that resemble even more closely those generated by humans. As part of our ongoing work, we plan to give the output of GENA to human experts in order to have their subjective impression on the quality of the generated narratives.

Acknowledgements. This research was supported by the project CENIT-BUSCAMEDIA 2009 ref. CEN20091026.

References

[1] Adeyanju, I., Wiratunga, N., Lothian, R., Sripada, S., Lamontagne, L.: Case Retrieval Reuse Net (CR2N): An Architecture for Reuse of Textual Solutions. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 14–28. Springer, Heidelberg (2009)

[2] Chatman, S.: Story and Discourse: Narrative Structure in Fiction and Film. Cornell University Press (June 1978)

[3] Lenz, M., Hübner, A., Kunze, M.: 5. Textual CBR. In: Lenz, M., Bartsch-Spörl, B., Burkhard, H.-D., Wess, S. (eds.) Case-Based Reasoning Technology. LNCS (LNAI), vol. 1400, pp. 115–137. Springer, Heidelberg (1998)

[4] Lewis, D.D.: Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 4–15. Springer, Heidelberg (1998)

[5] Nichols, N., Hammond, K.: Machine-generated multimedia content. In: Proceedings of the 2009 Second International Conferences on Advances in Computer-Human Interactions, ACHI 2009, pp. 336–341. IEEE Computer Society, Washington, DC (2009)

[6] Ontañón, S., Zhu, J.: Story and Text Generation through Computational Analogy in the Riu System. In: AIIDE, pp. 51–56. The AAAI Press (2010)

[7] Padró, L., Collado, M., Reese, S., Lloberes, M., Castelln, I.: Freeling 2.1: Five years of open-source language processing tools. In: Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., Tapias, D. (eds.) Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010), European Language Resources Association (ELRA), Valletta, Malta (May 2010)

[8] Pérez y Pérez, R., Sharples, M.: Mexica: A computer model of a cognitive account of creative writing. Journal of Experimental and Theoretical Artificial Intelligence 13(2), 119–139 (2001)

[9] Real, R., Vargas, J.M.: The probabilistic basis of jaccards index of similarity. Systematic Biology 45(3), 380–385 (1996)

[10] Recio, J.A., Díaz-Agudo, B., Gómez-Martín, M.A., Wiratunga, N.: Extending jCOLIBRI for Textual CBR. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS (LNAI), vol. 3620, pp. 421–435. Springer, Heidelberg (2005)

[11] Sahlgren, M., Cöster, R.: Using bag-of-concepts to improve the performance of support vector machines in text categorization. In: Proceedings of the 20th International Conference on Computational Linguistics, COLING 2004. Association for Computational Linguistics, Stroudsburg (2004)

[12] Smyth, B., Keane, M.T.: Adaptation-guided retrieval: questioning the similarity assumption in reasoning. Artificial Intelligence 102, 249–293 (1998)

[13] Turner, S.R.: Minstrel: a computer model of creativity and storytelling. Ph.D. thesis, University of California at Los Angeles, Los Angeles, CA, USA (1993)

On Knowledge Transfer in Case-Based Inference

Santiago Ontañón[1] and Enric Plaza[2]

[1] Computer Science Department
Drexel University
Philadelphia, PA, USA 19104
santi@cs.drexel.edu

[2] IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Catalonia (Spain)
enric@iiia.csic.es

Abstract. While similarity and retrieval in case-based reasoning (CBR) have received a lot of attention in the literature, other aspects of CBR, such as case reuse are less understood. Specifically, we focus on one of such, less understood, problems: *knowledge transfer*. The issue we intend to elucidate can be expressed as follows: *what knowledge present in a source case is transferred to a target problem in case-based inference?* This paper presents a preliminary formal model of knowledge transfer and relates it to the classical notion of analogy.

1 Introduction

In case-based reasoning (CBR), a problem is solved by first *retrieving* one or several relevant cases from a case-base, and then *reusing* the knowledge in the retrieved case (or cases) to solve the new problem. The retrieval stage in CBR has received a lot of attention in the literature, however, other aspects of CBR have received less attention and are less well understood; specifically, what knowledge can be reused from a previous case (source) to solve a new (target) case?

There is no generally agreed upon model of this process, which we will call the *knowledge transfer* process. This paper presents a model of knowledge transfer in case-based inference (CBI). Case-based inference, as described in [7] corresponds only to a part of the complete CBR cycle [1]. CBI basically accounts for the general inference process performed when predicting or characterizing a solution to a problem from a given set of cases, it does not include the process of adaptation or revision of the proposed solutions.

Consequently, in this paper, we intend to model the process of pure knowledge transfer, without intending to model the complete case reuse process, nor trying to encompass the whole variety of approaches to reuse in case-based reasoning, like rule-based adaptation. The issue we intend to elucidate can be expressed as follows: *what knowledge present in the source case is transferred to a target problem during case-based inference?*

In our model, we take a different direction from the CBI model of Hüllermeier [7], where they focus on prediction, i.e. classification and regressions tasks, since we focus on design tasks. While on prediction, the solution is selecting a solution form a set of possible solutions, on design tasks the solution is deceived by building a complex structure from "solution elements" (usually nodes and their relationships). The goal of this paper is then to give an account of what is transferred from a previous case to a solution case when it s a complex structure. Our model of knowledge transfer is based on the notions of refinement, subsumption, partial unification and amalgam, defined over a generalization space. This model is applicable to any representation formalism for which a relevant generalization space can be defined. Consequently, albeit we do take into account the notion of similarity, numerical measures of similarity are downplayed in this model, and we focus on a more symbolic notion of similarity. In our approach, it is more important to reason about *what is shared* among cases than the *degree* to which two cases share some of their content.

The work presented in this paper is an extension of the work in [15], where we introduced a preliminary version of this model. In this paper, we take one step forward, generalize the model to also cover multi-case adaptation and make more emphasis on its relation with analogical reasoning, as one of the underlying principles of case-based reasoning.

The remainder of this paper is organized as follows. First we introduce the idea of knowledge transfer in CBR in Section 2. Then, Section 3 briefly presents some necessary theoretical background for our formal model of knowledge transfer presented in Section 4. Finally, Section 5 discusses knowledge transfer in computational analogy and its relations with CBR.

2 Knowledge Transfer in CBR

In standard models of CBR, cases are typically understood as problem/solution pairs (p, s) or situation/outcome pairs. Therefore, solving a problem p' means finding finding or constructing a solution s' by adapting the solution of one or more retrieved cases. In this paper, we will consider a more general model, where cases are a single description, and where the problem and the solution are just two parts of this single description. In this view, an unsolved problem is just a partial description that needs completion.

The task of solving a problem in our view consists of two steps (in accordance to recent formal models of CBR [7]):

1. (case-based inference) finding a complete description by transferring information from retrieved cases to the problem at hand. Thus, the process of case-based inference can be further divided into two steps: case retrieval and knowledge transfer.
2. (adaptation) later performing any additional domain specific adaptations required to turn the complete description found by case-based inference into a valid solution for the domain at hand.

In the traditional CBR cycle [1], the reuse process encompasses both knowledge transfer and adaptation. The model presented in this section focuses exclusively on the process of knowledge transfer, rather than on the whole reuse process. Therefore, the outcome of the knowledge transfer process is not a valid solution, but the result of transferring knowledge from the one or more source cases to the target, which might still need to be adapted by using some domain specific rules, or any other reuse procedure. For that reason, we will refer to the result of knowledge transfer as a *conjecture*. Thus, we say that a conjecture is formed by transferring knowledge from source cases to a target problem —or, in other words, conjectures are the outcome of case-based inference. Some conjectures might constitute solutions, while some others might require adaptation.

There are multiple scenarios that define different knowledge transfer tasks:

- Transfer may be from a single or multiple retrieved cases.
- The unsolved problem description can be understood as a hard requirement (i.e. when the solved problem can only add elements to the unsolved problem description, but not change or remove anything to the problem description), or not (when the unsolved problem description just expresses some preferences of over the final solution).

For the sake of clarity, in this paper we will only provide a formalization of the hard requirements scenario. However, we will provide insights into how the soft requirement scenario can be easily modeled in our framework.

Our formalization is based on the notions of *generalization space* and that of *amalgam* and *partial unification*. For a more in-depth description of these ideas, the reader is referred to [13], here, we will just provide their intuitive ideas, sufficient to present out model of knowledge transfer.

3 Background

In this paper we will make the assumption that cases are terms in some *generalization space*. We define a generalization space as a partially ordered set $\langle \mathcal{L}, \sqsubseteq \rangle$, where $\mathcal{L}$ is a language, and $\sqsubseteq$ is a subsumption between the terms of the language $\mathcal{L}$. We say that a term ψ_1 subsumes another term ψ_2 ($\psi_1 \sqsubseteq \psi_2$) when ψ_1 is more general (or equal) than ψ_2[1]. Additionally, we assume that $\mathcal{L}$ contains the infimum element $\perp$ (or "any"), and the supremum element $\top$ (or "none") with respect to the subsumption order.

Next, for any two terms ψ_1 and ψ_2 we can define their *unification*, ($\psi_1 \sqcup \psi_2$), which is the *most general specialization* of two given terms, and their *anti-unification*, defined as the *least general generalization* of two terms, representing the most specific term that subsumes both. Intuitively, a unifier (if it exists) is a term that has all the information in both the original terms, and an anti-unifier is a term that contains only all that is common between two terms. Also, notice that, depending on $\mathcal{L}$, anti-unifier and unifier might be unique or not.

[1] In machine learning terms, $A \sqsubseteq B$ means that A is more general than B, while in description logics it has the opposite meaning, since it is seen as "set inclusion" of their interpretations.

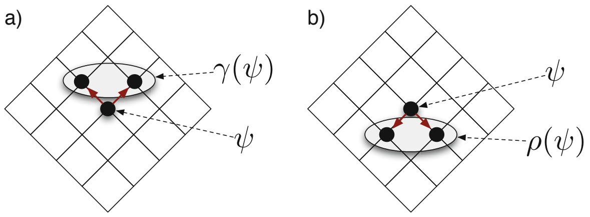


Fig. 1. A generalization refinement operator γ, and a specialization operator ρ

Let us now summarize the basic notions of *refinement operator* over partially ordered sets and introduce the concepts relevant for this paper —see [9] for a more in-depth analysis. Refinement operators are defined as follows:

Definition 1. *A* downward refinement operator ρ *over a partially-ordered set* $\langle \mathcal{L}, \sqsubseteq \rangle$ *is a function such that* $\rho(\psi) \subseteq \{\psi' \in \mathcal{L} | \psi \sqsubseteq \psi'\}$ *for all* $\psi \in \mathcal{L}$.

Definition 2. *An* upward refinement operator γ *over a partially-ordered set* $\langle \mathcal{L}, \sqsubseteq \rangle$ *is a function such that* $\gamma(\psi) \subseteq \{\psi' \in \mathcal{L} | \psi' \sqsubseteq \psi\}$ *for all* $\psi \in \mathcal{L}$.

In other words, upward refinement operators generate elements of $\mathcal{L}$ which are more general, whereas downward refinement operators generate elements of $\mathcal{L}$ which are more specific, as illustrated by Figure 1. Typically, the symbol γ is used for upward refinement operators, and ρ for downward refinement operators.

Refinement operators can be used to navigate the generalization space using different search strategies, and are widely used in Inductive Logic Programming. For instance, if we have a term representing "a German minivan", a generalization refinement operator would return generalizations like "a European minivan", or "a German vehicle". Moreover, in practice, it is preferable to have refinement operators that do not perform large generalization or specialization leaps, i.e. that make the smallest possible change in a term when generalizing or specializing, to better explore the space of generalizations as a search space [14].

3.1 Amalgams

The notion of *amalgam* can be conceived of as a generalization of the notion of unification over terms. The unification of two terms (or descriptions) ψ_a and ψ_b is a new term $\phi = \psi_a \sqcup \psi_b$, called unifier. All that is true for ψ_a or ψ_b is also true for ϕ.; e.g. if ψ_a describes "a red vehicle" and ψ_b describes "a German minivan" then their unification yields the description "a red German minivan." Two terms are not unifiable when they possess contradictory information; for instance "a red French vehicle" is not unifiable with "a red German minivan". The strict definition of unification means that any two descriptions with only one item with contradictory information cannot be unified.

An *amalgam* of two terms (or descriptions) is a new term that contains *parts from these two terms.* For instance, an amalgam of "a red French vehicle" and "a

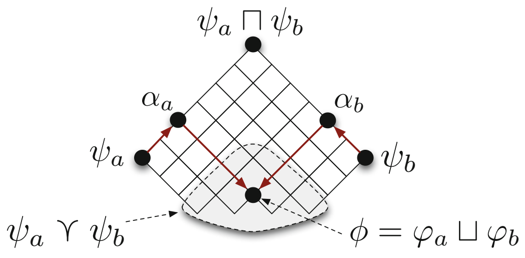

$\psi_a \sqcap \psi_b$
α_a
α_b
ψ_a
ψ_b
$\psi_a \curlyvee \psi_b$
$\phi = \varphi_a \sqcup \varphi_b$

Fig. 2. Illustration of the idea of amalgam between two terms ψ_a and ψ_b

German minivan" is "a red German minivan"; clearly there are always multiple possibilities for amalgams, since "a red French minivan" is another example of amalgam. The notion of amalgam, as a form of partial unification, was formally defined in [13]. For the purposes of this paper, we will introduce a few necessary concepts.

Definition 3. (Amalgam) *The set of* amalgams *of two terms ψ_a and ψ_b is the set of terms such that:*

$$\psi_a \curlyvee \psi_b = \{\phi \in \mathcal{L}^+ | \exists \alpha_a, \alpha_b \in \mathcal{L} : \alpha_a \sqsubseteq \psi_a \ \wedge \ \alpha_b \sqsubseteq \psi_b \ \wedge \ \phi = \alpha_a \sqcup \alpha_b\}$$

where $\mathcal{L}^+ = \mathcal{L} - \{\top\}$

Thus, an amalgam of two terms ψ_a and ψ_b is a term that has been formed by unifying two generalizations α_a and α_b such that $\alpha_a \sqsubseteq \psi_a$ and $\alpha_b \sqsubseteq \psi_b$ —i.e. an amalgam is a term resulting from combining some of the information in ψ_a with some of the information from ψ_b, as illustrated in Figure 2. Formally, $\psi_a \curlyvee \psi_b$ denotes the set of all possible amalgams; however, whenever it does not lead to confusion, we will use $\psi_a \curlyvee \psi_b$ to denote one specific amalgam of ψ_a and ψ_b.

The terms α_a and α_b are called the *transfers* of an amalgam $\psi_a \curlyvee \psi_b$. α_a represents all the information from ψ_a which is *transferred* to the amalgam, and α_b is all the information from ψ_b which is transferred into the amalgam. As we will see later, this idea of transfer is akin to the idea of *transferring* knowledge from the source to target in CBR, and also in computational analogy [4].

Intuitively, an amalgam is *complete* when all which can be transferred from both terms into the amalgam has been transferred, i.e. if we wanted to transfer more information, α_a and α_b would not have a unifier.

For the purposes of case reuse, we introduce the notion of asymmetric amalgam, where one term is fixed while only the other term is generalized in order to compute an amalgam.

Definition 4. (Asymmetric Amalgam) *The* asymmetric amalgams $\psi_s \overrightarrow{\curlyvee} \psi_t$ *of two terms ψ_s (*source*) and ψ_t (*target*) is the set of terms such that:*

$$\psi_s \overrightarrow{\curlyvee} \psi_t = \{\phi \in \mathcal{L}^+ | \exists \alpha_s \in \mathcal{L} : \alpha_s \sqsubseteq \psi_s \ \wedge \ \phi = \alpha_s \sqcup \psi_t\}$$

In an asymmetric amalgam, the target term is transferred completely into the amalgam, while the source term is generalized. The result is a form of partial unification that conserves all the information in ψ_t while relaxing ψ_s by generalization and then unifying one of those more general terms with ψ_t itself. Finally, an asymmetric amalgam is *maximal* when all knowledge in ψ_s that is consistent with ψ_t is transferred to the solution ψ'_t —i.e. $\psi'_t \in \psi_s \overrightarrow{\curlyvee} \psi_t$ is maximal iff $\nexists \psi''_t \in \psi_s \overrightarrow{\curlyvee} \psi_t$ such that $\psi'_t \sqsubset \psi''_t$.

4 A Model of Knowledge Transfer

This section provides a formalization of the idea of knowledge transfer in CBR for the scenarios of single and multi-case retrieval, but only considering problems as a *hard requirement* (see Section 2).

4.1 Knowledge Transfer with Hard Requirements

Let us define the task of knowledge transfer for single case reuse with hard requirements as follows.

Given. A case base $\Delta = \{\psi_1, \ldots \psi_m\}$ and a target description ψ_t
Find. A 'maximal' case ψ'_t such that $\psi_t \sqsubset \psi'_t$ (a *conjecture*)

Clearly, if there is some $\psi_i \in \Delta$ such that $\psi_t \sqsubset \psi_i$ then ψ_i is a solution, and the conjecture can be built simply by unifying query and solution: $\psi_t \sqcup \psi_i = \psi_i$. This specific situation is called in CBR literature "solution copy with variable substitution" [8]. Also, notice that the CBI model worries about maximal amalgams, while determining whether such case is *complete* or not corresponds to the whole CBR task and is beyond the scope of the knowledge transfer model.

In general, when there is no case such that $\psi_t \sqsubset \psi_i$, unification is not enough, and knowledge transfer requires the use of amalgams, and in particular of the asymmetric amalgam. Knowledge transfer from a source ψ_s with hard requirements produces *hard conjectures*, defined as follows:

Definition 5. ***(Hard Transfer)*** *A* hard transfer *α for target ψ_t from a source ψ_s is a term $\alpha \sqsubseteq \psi_s$ such that $\alpha \sqcup \psi_t \neq \top$, i.e. a generalization of ψ_s that unifies with ψ_t. Thus, the set of* hard transfers *for target ψ_t from a source ψ_s is: $G(\psi_s, \psi_t) = \{\alpha \in \mathcal{L} | \alpha \sqsubseteq \psi_s \wedge \alpha \sqcup \psi_t \neq \top\}$.*

Definition 6. ***(Hard Conjecture)*** *Given a* hard transfer *$\alpha \in G(\psi_s, \psi_t)$, a* conjecture *for target ψ_t is a term in $\psi_s \overrightarrow{\curlyvee} \psi_t$ where α is the transfer. The set of hard conjectures K_H for target ψ_t from a source ψ_s is $K_H(\psi_s, \psi_t) = \psi_s \overrightarrow{\curlyvee} \psi_t$.*

We will be interested in the most specific conjectures, which are the ones coming from maximal asymmetric amalgams, and as a subset of $G(\psi_s, \psi_t)$. Whether a maximal conjecture is a complete solution is discussed later in Section 4.2

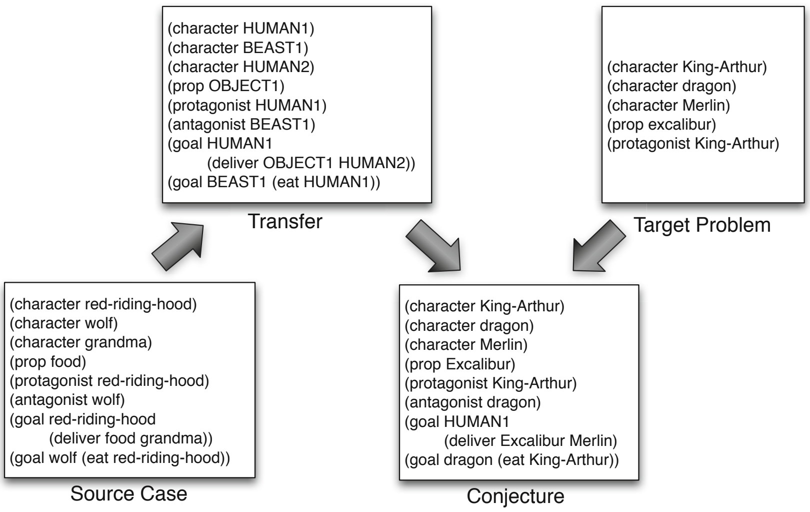


Fig. 3. Exemplification of the concepts of source, target, transfer and conjecture in a story generation domain

In order to illustrate our model with an example let us consider the task of story generation (which has been addressed using CBR by many authors [19]). In this domain, the goal is to generate a story or a story schema (decide which characters exist in the story, which props, which are the goals of the characters, which actions will they perform, etc.). The case base contains a collection of predefined stories, and a problem corresponds to a set of requirements over the story we want the system to generate. We can see, first of all, that there is no clear distinction between problem and solution. A case is just a complete story, whereas a problem is just a partially specified story. Figure 3 illustrates our model showing the following elements: a target problem consisting of an incomplete story specifying three characters (from the King Arthur fantasy world), and asking the system to generate a story that has three characters, King Arthur, Merlin and a dragon, where King Arthur is the main character and where Excalibur is involved. The system happens to retrieve a case with the story of *Little Red Riding Hood* (shown on the bottom left). We don't show the complete case, for space limitations, but in addition to the definitions shown in Figure 3, the retrieved case should contain the list of actions that constitute the plot of the story. We show the transfer, which is a generalization of the retrieved case, and a possible conjecture, which is a unification of the transfer with the target problem. In this example, the resulting story has King Arthur wanting to deliver excalibur to Merlin, while the dragon wants to eat Kind Arthur. We show one possible conjecture, but notice that many different conjectures could be formed here, by transferring different aspects from the retrieved case.

The result of CBI is a *conjecture* in the sense that it is a plausible solution for ψ_t. Notice that, (1) a conjecture may be an incomplete solution, and (2) a conjecture is not assured to be correct. Moreover, since there may be more than one conjecture, (3) the issue of which conjecture should be selected has also to be specified. Let us review them in turn.

4.2 Conjecture Incompleteness

The purpose of knowledge transfer in case reuse is to transfer to the target as much knowledge as possible (consistent with the target). This "as much as possible" is satisfied if we take as *transfer* a term α that is one of the most specific generalizations of the source that are unifiable with the target, what we called maximal amalgams in Section 3.1. Nevertheless, some information is lost in the generalization path $\psi_s \xrightarrow{\gamma} \alpha$, which corresponded to the *remainder* [14]. Specifically, the remainder $r(\psi, \alpha)$ of a term ψ and a generalization $\alpha \sqsubset \psi$ is a term ϕ such that $\alpha \sqcup \phi = \psi$ (and there is no $\phi' \sqsubset \phi$ such that $\alpha \sqcup \phi' = \psi$). That which is lost from the source case will be called *source differential* in our model.

Definition 7. ***(Source Differential)*** *The source differential ψ_D of a source term ψ_s with respect to a transfer $\alpha \in G(\psi_s, \psi_t)$ is the remainder $r(\psi_s, \alpha)$.*

Notice that, even assuming the source ψ_s to be a consistent and complete case in a case base, now we view the source as having two parts with respect to the target, namely $\psi_s = \alpha \sqcup r(\psi_s, \alpha)$, and only one of this parts (α) is transferred to the target. Therefore, we cannot assume, in general, that the result of case-based inference $\alpha \sqcup \psi_t$ (even when $\alpha \sqcup \psi_t$ is maximal) is a complete solution for the new case (that depends on what is in $r(\psi_s, \alpha)$ and what are the requirements for a solution to be 'complete').

Depending on the task a CBR system is performing, this partial solution may be enough. Classical analogy systems take this approach: the goal is to transfer (as much as possible) knowledge from source to target —there is no notion of an externally enforced task that demands some kind of completeness to solutions. Thus, our model of case-based inference encompasses maximal conjectures, but solution completeness is out of its scope, since it depends on the whole CBR process beyond case-based inference.

4.3 Conjecture Correctness

A conjecture $\psi_t \sqcup \alpha$ may be maximal, but even so this might be a correct solution or not with respect to ψ_t. If we see ψ_t as a set of requirements that the complete solved target case must satisfy, then if a conjecture $\psi_t \sqcup \alpha$ is maximal, then the conjecture $\psi_t \sqcup \alpha$ is correct. Although this supplementary assumptions makes sense in theory (if ψ_t expresses the "requirements" to be satisfied), often CBR systems operate in domains where it is not feasible to assure that ψ_t is a complete requirement on the correctness and completeness of solutions; it is more reasonable to assume that ψ_t is a partial requirement and the final acceptability or correctness is left to be assessed by the Revise process.

Therefore, knowledge transfer in case reuse produces a solution that is consistent and maximal, but possibly partial, and not assured to be correct; i.e.

produces a conjecture. Since there are multiple transfers that can produce multiple conjectures, we turn now into the issue of assessing, comparing, and ranking conjectures.

4.4 Conjecture Ordering

Multiplicity of maximal conjectures may have two causes. The first is that $\varGamma(\psi_s, \psi_t)$ might not be unique. The second is when, even if $\varGamma(\psi_s, \psi_t)$ is unique, more than one source is taken into account (as considered in the next section): a set of k precedent cases $\mathbb{P}_k = (\psi_1, \ldots, \psi_k)$ produce a set of transfers $\varPsi(\mathbb{P}_k) = \varPsi_1 \cup \ldots \cup \varPsi_k$, which in turn generates a set of conjectures.

Conjectures in $K_H(\mathbb{P}_k, \psi_t)$ may be complete, but from a practical point of view it is useful to rank them according to their estimated plausibility, their degree of completeness, or any other heuristic that can be used in a particular application domain. Typically, the Retrieve phase estimates relevance of precedent cases with some similarity measure, so we can use the similarity degrees $(s_1 \geq \ldots \geq s_k)$ of the k retrieved cases $\mathbb{P}_k = \{\psi_1, \ldots, \psi_k\}$ to induce a partial order on the set of transfers: $\langle \varPsi(\mathbb{P}_k), \geq \rangle = (\varPsi_1 \geq \ldots \geq \varPsi_k)$. Thus, the conjectures coming from transfers originating in more similar precedent cases (or those transferring more knowledge from more similar cases, in the case of multi-case reuse) are preferred to those from less similar cases.

Since conjectures are in general partial solutions, using some measure that estimates the degree of completeness of conjectures may also be used for ranking conjectures. Domain knowledge can be used to estimate conjecture completeness. In previous work [15] we proposed a measure called *preservation degree* for this purpose. This ranking can be combined with the similarity based ordering to establish a combined partial order on conjectures.

4.5 Knowledge Transfer from Multiple Cases

There are scenarios when the conjecture generated using case-based reasoning is a combination of more than one case in the case base. The intuitive idea in this scenario is that, instead of an asymmetric amalgam where a generalization of a single retrieved case (transfer) is unified with the target problem, we will have an asymmetric amalgam where a generalization of each of the source cases (one transfer per source case) is unified with the target problem. Therefore, instead of a single transfer, we will have multiple transfers (one per source case).

This process can be formally modeled again as an asymmetric amalgam.

Definition 8. ***(Hard Conjecture from Multiple Cases)*** *Given a set of source cases $\{\psi_s^1, \ldots, \psi_s^n\}$, a target ψ_t and a set of hard transfers $\alpha_1, ..., \alpha_n$, where $\alpha_i \in G(\psi_s^i, \psi_t)$, a* conjecture *for a target ψ_t is a term in $\{\psi_s^1, \ldots, \psi_s^n\} \overrightarrow{\curlyvee} \psi_t$, where $\alpha_1, ..., \alpha_n$ are the transfers. The set of hard conjectures K_H for target ψ_t is $K_H(\{\psi_s^1, \ldots, \psi_s^n\}, \psi_t) = \{\psi_s^1, \ldots, \psi_s^n\} \overrightarrow{\curlyvee} \psi_t$.*

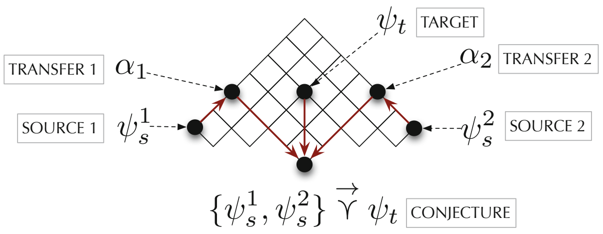


Fig. 4. An schema showing multi-case hard conjecture from two sources ψ_s^1 and ψ_s^2

This idea is illustrated in Figure 4 for the situation of two source cases.

Although $K_H(\{\psi_s^1, \ldots, \psi_s^n\}, \psi_t)$ is formally well defined, complexity clearly increases as the number of sources increases, since the number of possible conjectures grows. In practical approaches, a CBR system will typically use a small number of source cases, say 2 or 3, and will use heuristics or domain knowledge that restrict the set of amalgams to consider.

Let us illustrate the idea with the same story generation domain used before. This time, assume that two cases were retrieved: *Little Red Riding Hood* and *Star Wars*. The target problem is the same as the one shown in Figure 3. This time, there will be two different transfers, one from each case, and the conjecture will be the unification of the two transfers with the target problem. For example, if the transfer form *Little Red Riding Hood* is that the dragon wants to eat the main character, and the transfer from *Star Wars* is that the main character wants to learn how to use a sword to defeat the villain and asks another character to train him/her, the resulting story would be the following: King Arthur wants Merlin to train him in the use of Excalibur to defeat the dragon, and the dragon wants to eat King Arthur. Notice, that by transferring from more than one story, there is a wider variety of conjectures that can be formed, and thus, the chances of finding a good solution are also higher.

For the sake of space, in this paper we have only considered the scenario of seeing problems as hard requirements. This means that the conjectures proposed by a CBR system always satisfy the target problem. In the soft requirements scenario, the term representing the problem is considered to just express the preferences over the kind of solutions we want. Therefore, instead of considering asymmetric amalgams, the soft requirements scenario is modeled with the symmetric amalgams, where both the retrieved case and the target problem can be generalized in order to produce the final conjecture. That is, if the system cannot find any solution that completely satisfies the target problem, it can relax the problem, and find a solution that only partially satisfies the target problem.

5 Knowledge Transfer in Analogy

We turn now to discuss how the classic concept of analogical reasoning [4] is related to our model of knowledge transfer, and to CBR in general. It is well accepted that CBR and analogical reasoning are tightly related and share some common underlying principles [10]. In this section we will see how our model of knowledge transfer underlies both CBR and some forms of analogical reasoning, showing that CBR and analogy indeed share a common underlying formal reasoning mechanism, at least in the limited scope of knowledge transfer.

Computational models of analogy operate by identifying similarities and transferring knowledge between a source domain S and a target domain T. This process can be divided into four stages [6]: 1) *recognition* of a candidate analogous source, S, 2) *elaboration* of an analogical mapping between source domain S and target domain T, 3) *evaluation* of the mapping and inferences, and 4) *consolidation* of the outcome of the analogy for other contexts (i.e. learning). At a superficial level, those 4 processes can be likened to the 4 processes of CBR: retrieve, reuse, revise and retain, although some differences exist. For example, while the reuse process in CBR aims at generating a candidate solution for the problem at hand, the elaboration step in computational analogy limits itself to mapping a source domain to a target domain and proposing candidate inferences (conjectures, in the vocabulary used in this paper). Another piece of evidence that the 4 process of analogy can be likened to those in the CBR cycle is that CBR theoretical frameworks, such as Richter's *knowledge containers* can be applied to analyze computational analogy processes [16].

Moreover, we would like to emphasize that *analogy* is an overloaded concept. The previous 4 step process models the complete cycle of analogical reasoning as understood in cognitive science. However, the term analogical reasoning in mathematics and logic corresponds just to the elaboration step. In the remainder of this paper, we will specifically focus our attention on this elaboration step, which is the most studied in computational models of analogy like SME [4].

5.1 Analogy as a Special Case of Induction

Analogy in the logical sense is typically defined as the process of transferring knowledge or inferences from a particular source domain S to another particular target domain T. John Stuart Mill [12, Ch. XX] argued that analogy is simply a special case of induction. In his view, analogy could be reduced to: "Two things resemble each other in one or more respects; a certain proposition is true of the one; therefore it is true of the other". That is to say, analogy between two situations S and T can be interpreted as having two steps:

Inductive Step: In a first step we perform an inductive leap. Assume that S and T are similar in some aspects, we will use anti-unification $S \sqcap T$ to denote all the information shared between S and T (i.e. that in which they are similar). Now, given a proposition α which is true in S (i.e. $\alpha \sqsubseteq S$) but we don't know if it is true in T, we assume (inductive leap) that $S \sqcap T$ is the cause of α —i.e. $S \sqcap T$ implies α (which we will write as $S \sqcap T \rightarrow \alpha$).

Deductive Step: Then, in a second step, we apply the inductively derived assumption $S \sqcap T \rightarrow \alpha$ to derive that α is also true in T, and conclude $\alpha \sqcup T$ is true (i.e. the target with the added piece of knowledge α is also true).

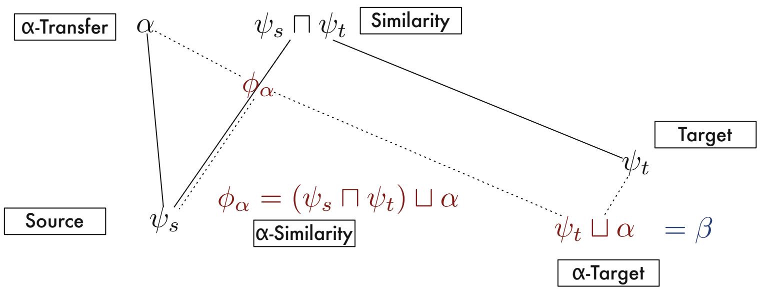


Fig. 5. Subsumption relations among the terms involved in analogy $\psi_s \underset{\psi_t}{\xrightarrow{a}} \beta$

Let us illustrate this analogical reasoning principle with a typical example. Let us consider our solar system as the source domain T, and Bohr's model of the atom as the target domain S. Both domains are similar in some aspects, $S \sqcap T =$ "There are smaller elements orbiting a larger element in the center". We know that the following statement is true for the solar system S: $\alpha =$ "there is an attraction force between the small elements and the larger element in the center". We can now use the previous model of analogical reasoning in the following way. In the first (inductive) step of analogy we reach the following assumption: $S \sqcap T \to \alpha$ (which means that the fact that there are elements orbiting is enough to conclude that there is an attraction force). In the second (deductive) step, we apply $S \sqcap T \to \alpha$ to T and conclude that there in Bohr's model of the atom there must also be an attraction force between the small elements and the larger element in the center. By adding this new piece of information to T, now we can conclude $\alpha \sqcup T$, that represents the model of the atom with the added piece of knowledge referring to the attraction force.

5.2 Knowledge Transfer in Analogy

This view of analogy can be defined as follows in a generalization space.

Definition 9. *Given two terms $\psi_s, \psi_t \in \mathcal{L}$ (called source and target respectively) a formula $\beta \neq \top$ is derived by analogy whenever:*

1. $\exists \alpha : \alpha \sqsubseteq \psi_s \wedge \alpha \not\sqsubseteq (\psi_s \sqcap \psi_t)$ *(α is true in source only)*
2. $\beta = \alpha \sqcup \psi_t$ *(knowledge α is transferred to target)*

where α is the knowledge transferred from source to target.

Since $\alpha \not\sqsubseteq \psi_s \sqcap \psi_t$ we cannot (deductively) derive that α is true in ψ_t. Therefore, this analogical reasoning requires an inductive step, which can be seen as a defeasible or conjectural inference. This "inductive analogy" model is illustrated in Figure 5. The solid lines depict sound inferences, i.e. the subsumption relationships among terms (ψ_s, ψ_t and α). Analogy makes some conjectural inferences shown as dotted lines. Specifically, if α is not inconsistent with ψ_t (i.e. $\alpha \sqcup \psi_t \neq \top$) then possibly both situations may also have α in common; this is represented as the term $\phi_\alpha = \alpha \sqcup \psi_t$ that is conjectured to be true. Now, assuming ϕ_α is true, and ψ_t is true, we can then conjecture that $\beta = \psi_t \sqcup \alpha$ is true (i.e. that α can be "transferred to" ψ_t).

This conjectural inference can be seen in two ways: induction or knowledge transfer, that nonetheless are equivalent. In the knowledge transfer approach, we derive β by conjecturing α is also true in the target (i.e. we derive $\alpha \sqcup \psi_t$); i.e. we use the idea of asymmetric amalgam to derive β by transferring α to ψ_t.

In the inductive model of analogy we conjecture that the implicit generalization should also include α as being true (that is we move from $\psi_s \sqcap \psi_t$ to ϕ_α). Later, since the target also shares $\psi_s \sqcap \psi_t$, we can (deductively) infer that α is true in the target. Figure 5 shows how both views reach the same conjecture.

5.3 Analogy and Case-Based Reasoning

The "inductive analogy" model sheds some light on the nature of analogical reasoning, and also provides insights on how to assess when the conclusions reached by analogy are stronger or weaker.

Conclusions reached by analogy are considered strong when the similarity between source and target is high. In Stuart Mill's words: "[...] it follows that where the resemblance is very great, the ascertained difference very small, and our knowledge of the subject-matter tolerably extensive, the argument from analogy may approach in strength very near to a valid induction". This follows from his inductive view of analogy, because if source and target are not very similar, then $S \sqcap T$ would contain very little information, and thus the rule $S \sqcap T \rightarrow \alpha$ reached by induction would most likely be an over generalization. Moreover, if $S \sqcap T$ contains a lot of information, then $S \sqcap T \rightarrow \alpha$ would be a rule with a very narrow scope (only applicable to those domains satisfying $S \sqcap T$), and thus more likely to be correct.

As stated above, conclusions reached by analogy can be seen as a knowledge transfer process from the source domain to the target domain. It should be now clear that the principles underlying knowledge transfer are a special case of analogical reasoning. This can be seen when we bear in mind that the assumption behind CBR, namely "similar problems have similar solutions", is just a special case of the analogical reasoning principle: "if two things resemble each other in one or more respects; a certain proposition is true of the one; therefore it is true of the other". Moreover, the second principle of analogy, stating that analogical reasoning reaches stronger conclusion when the two domains are more similar, explains the principle behind the most common approaches to case retrieval in CBR, that simply look for the most similar case to the problem at hand.

Notice that what we are stating is that knowledge transfer (and thus CBI) is a special case of analogical reasoning, not that the whole case-based reasoning paradigm is. Moreover, we also state that the goal of the retrieval step in CBR should be to provide a source domain from which the conclusions reached by analogy (knowledge transfer) are stronger. Solution adaptation, typically domain dependent, is not explained by analogical reasoning, and constitutes the main theoretical difference between CBR and computational models of analogy in cognitive science.

6 Discussion

This paper has presented a model of knowledge transfer in case-based inference based on the idea of partial unification. We have focused on cases are represented as terms in a generalization space. In our model, case reuse is seen as having two steps: a first step (case-based inference) where knowledge is transferred from one or several source cases to the target case (called a conjecture), and a second step (adaptation) where the conjecture might need to be adapted. This paper has focused on a model of the first step.

Our model of knowledge transfer offers insights on the relation between case reuse and analogical reasoning. Previous work on relating CBR with analogy has focused on superficial aspects such as CBR being typically intra-domain, whereas analogy is inter-domain [18]. In our model, we can see that analogical reasoning is related to the knowledge transfer step of case reuse rather than with the second (adaptation) step. The work of Prade and Richard [17] is an exception, and proposed a Boolean model of analogical reasoning and suggested it could be used for adaptation in CBR, following an inductive view of analogy as we presented above. An interesting line of future work is the relation of knowledge transfer with conceptual blending [5]. We have seen that analogy can be likened to an asymmetric amalgam, whereas conceptual blending could be seen as a form of symmetric amalgam.

Our work is related to existing general models of case reuse, like [2]. However, such models focus on the adaptation step, and typically obviate the process of knowledge transfer (transfer is seen as a mere “solution copy” from source to target). We believe that this oversimplification of the knowledge transfer problem is at the root of the difficulty of finding general models of multi-case reuse. The work presented in this paper is a step towards that direction, since it can easily cope with transferring knowledge from multiple sources.

Also related is the work on case-based inference [7], but they focus on prediction (classification and regression tasks where the outcome is a form of similarity-based inference). The difference with our work is that we have focused on how complex solutions and conjectures can be formed by transferring knowledge from one or multiple source cases to a partial solution of a target case.

Part of our long term goal is understanding case reuse and its relation to other forms of reasoning. We envision case-based inference as a form of conjectural or defeasible inference, like other forms of non-monotonic reasoning (induction, abduction and hypothetical reasoning). The model presented in this paper is one step towards this goal. As future work, we want to formalize the process of knowledge transfer from multiple source cases, and develop case reuse methods based on the idea of knowledge transfer.

Finally, although the model presented in this paper is theoretical, practical implementations of the underlying principles are possible. For example, our previous work on similarity measures over generalization spaces [14], and on case adaptation in multiagent systems using amalgams [11] are steps in this direction.

Acknowledgements. This research was partially supported by project Next-CBR (TIN2009-13692-C03-01).

References

[1] Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. Artificial Intelligence Communications 7(1), 39–59 (1994)

[2] Bergmann, R., Wilke, W.: Towards a new formal model of transformational adaptation in case-based reasoning. In: European Conference on Artificial Intelligence (ECAI 1998), pp. 53–57. John Wiley and Sons (1998)

[3] Cojan, J., Lieber, J.: Belief Merging-Based Case Combination. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 105–119. Springer, Heidelberg (2009)

[4] Falkenhainer, B., Forbus, K.D., Gentner, D.: The structure-mapping engine: Algorithm and examples. Artificial Intelligence 41, 1–63 (1989)

[5] Fauconnier, G.: Conceptual blending and analogy. In: Gentner, D., Holyoak, K.J., Kokinov, B.K. (eds.) The analogical mind Perspectives from cognitive science, pp. 255–285. MIT Press (2001)

[6] Hall, R.P.: Computational approaches to analogical reasoning: a comparative analysis. Artificial Intelligence 39(1), 39–120 (1989)

[7] Hüllermeier, E.: Case-Based Approximate Reasoning. Theory and Decision Library, vol. 44. Springer (2007)

[8] Kolodner, J.: Case-based reasoning. Morgan Kaufmann (1993)

[9] van der Laag, P.R.J., Nienhuys-Cheng, S.H.: Completeness and properness of refinement operators in inductive logic programming. Journal of Logic Programming 34(3), 201–225 (1998)

[10] Mántaras, R.L.D., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M.L., Cox, M.T., Forbus, K., Keane, M., Aamodt, A., Watson, I.: Retrieval, reuse, revision and retention in case-based reasoning. Knowl. Eng. Rev. 20(3), 215–240 (2005)

[11] Manzano, S., Ontañón, S., Plaza, E.: Amalgam-Based Reuse for Multiagent Case-Based Reasoning. In: Ram, A., Wiratunga, N. (eds.) ICCBR 2011. LNCS, vol. 6880, pp. 122–136. Springer, Heidelberg (2011)

[12] Mill, J.S.: The Collected Works of John Stuart Mill, vol. 7. Liberty Fund (2006)

[13] Ontañón, S., Plaza, E.: Amalgams: A Formal Approach for Combining Multiple Case Solutions. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010. LNCS, vol. 6176, pp. 257–271. Springer, Heidelberg (2010)

[14] Ontañón, S., Plaza, E.: Similarity measuress over refinement graphs. Machine Learning Journal 87, 57–92 (2012)

[15] Ontañón, S., Plaza, E.: Toward a knowledge transfer model of case-based inference. In: Proceedings of the Fifteenth International Florida Artificial Intelligence Research Society (FLAIRS). AAAI Press (2012)

[16] Ontañón, S., Zhu, J.: On the role of domain knowledge in analogy-based story generation. In: IJCAI, pp. 1717–1722 (2011)

[17] Prade, H., Richard, G.: Analogy-Making for Solving IQ Tests: A Logical View. In: Ram, A., Wiratunga, N. (eds.) ICCBR 2011. LNCS, vol. 6880, pp. 241–257. Springer, Heidelberg (2011)

[18] Seifert, C.M.: Analogy and case-based reasoning. In: Proc. of a Workshop on Case-Based Reasoning, Pensacola Beach, FL, pp. 125–129 (1989)

[19] Turner, S.R.: Minstrel: a computer model of creativity and storytelling. Ph.D. thesis. University of California at Los Angeles, Los Angeles, CA, USA (1993)

Case-Based Aggregation of Preferences for Group Recommenders

Lara Quijano-Sánchez[1], Derek Bridge[2],
Belén Díaz-Agudo[1], and Juan A. Recio-García[1]

[1] Department of Software Engineering and Artificial Intelligence,
Universidad Complutense de Madrid, Spain
[2] Department of Computer Science, University College Cork, Ireland
{lara.quijano,jareciog}@fdi.ucm.es, d.bridge@cs.ucc.ie,
belend@sip.ucm.es

Abstract. We extend a group recommender system with a case base of previous group recommendation events. We show that this offers a new way of aggregating the predicted ratings of the group members. Using user-user similarity, we align individuals from the active group with individuals from the groups in the cases. Then, using item-item similarity, we transfer the preferences of the groups in the cases over to the group that is seeking a recommendation. The advantage of a case-based approach to preference aggregation is that it does not require us to commit to a model of social behaviour, expressed in a set of formulae, that may not be valid across all groups. Rather, the CBR system's aggregation of the predicted ratings will be a lazy and local generalization of the behaviours captured by the neighbouring cases in the case base.

1 Introduction

Groups often holiday together; tour museums and art galleries together; visit historic sights together; attend concerts and other events together; dine in restaurants together; watch movies and TV programmes together; listen to music together; cook and eat together. They must select the items which they intend to consume together, ranging from holiday destinations to recipes, in a way that reconciles the different preferences and personalities of the group members. For this, they may seek the support of a recommender system. But where the majority of recommender systems suggest items based on the preferences of an individual consumer, *group recommender systems* suggest items taking into account the preferences and personalities of the members of a group [4].

Commonly, group recommender systems aggregate predicted ratings for group members [4]: for each group member, a single-person recommender system predicts a set of ratings for the candidate items; then, the group recommender aggregates the ratings. The new group recommender system that we present in this paper takes the same approach, i.e. it aggregates the preferences of the group members, but it uses Case-Based Reasoning (CBR) for the aggregation. Figure 1 is suggestive of its operation. The system has a case base of past group

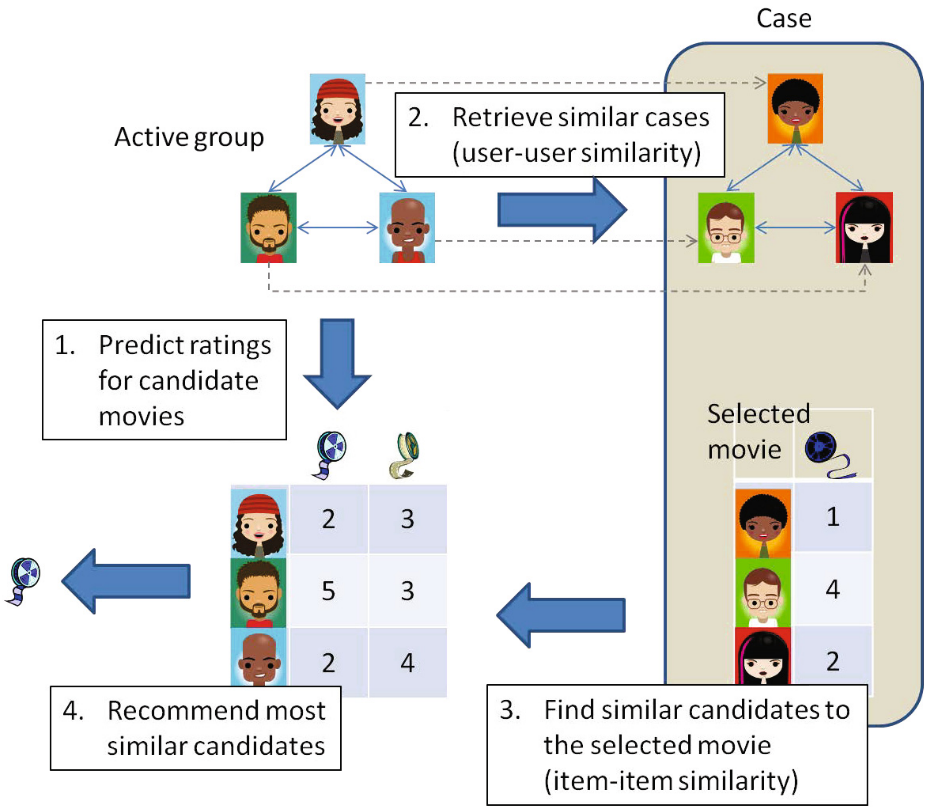

Case
Active group
2. Retrieve similar cases (user-user similarity)
1. Predict ratings for candidate movies
Selected movie
2 3
5 3
2 4
1
4
2
4. Recommend most similar candidates
3. Find similar candidates to the selected movie (item-item similarity)

Fig. 1. Overview of the case-based recommender

recommendation events. Each case (right-hand side in the diagram) records the members of the group; the candidate items; the item that the group chose to consume together, which we will call the *selected item*; and the ratings that each group member gave to the selected item after consuming it. To make a recommendation to a new active group (top-left in the diagram), the CBR system deploys a unique combination of user-user and item-item similarity, as follows:

Step 1: First, it uses a user-based collaborative recommender to predict a rating for each candidate item by each group member.

Step 2: Next, it retrieves cases, i.e. past group recommendation events, that involve groups that are similar to the active group. Case retrieval uses the user-user similarity measure, and, as a by-product, it aligns each member of the active group with a member of the group in the case (the dashed lines in Figure 1). The similarity measure compares group members on their age, gender, personality and ratings and the degrees of trust between members of each group (the solid lines between group members in the diagram).

Step 3: Then, it reuses each case that is retrieved: the contributions that each group member made in choosing the selected item are transferred to the corresponding member of the active group. This is done by scoring the new candidate items by their item-item similarity to the selected item. In this way, the retrieved cases act as implicit models of group decision-making, which are transferred to the decision-making in the active group.

Step 4: Finally, it recommends the candidate items that have obtained the highest scores.

The paper explains this more fully. Section 2 gives some background exposition that we need for later sections; Section 3 describes an existing group recommender system, which we will use for comparison purposes; Section 4 describes the new case-based group recommender; Section 5 describes an experiment that compares the new recommender with the one we developed previously; and Section 6 concludes and presents some ideas for future work.

2 User-User and Item-Item Similarity

Suppose there are n users, $U = \{u : 1 \ldots n\}$, and m items (e.g. movies), $I = \{i : 1 \ldots m\}$. Let r be a ratings matrix and $r_{u,i}$ be the rating that user u assigns to item i. Ratings are on a numeric scale, e.g. 1 = terrible and 5 = excellent, but $r_{u,i} = \perp$ signals that u has not yet rated i.

The similarity between one user and another, $u \in U, u' \in U, u \neq u'$, can be computed using Pearson Correlation [3], ρ. In effect, this computes the similarity between two *rows* in a ratings matrix like the one in the table in the lower left-hand part of Figure 1. The *user-user similarity* is:

$$\rho_{u,u'} \hat{=} \frac{\sum_{i \in I \wedge r_{u,i} \neq \perp \wedge r_{u',i} \neq \perp} (r_{u,i} - \bar{r}_u)(r_{u',i} - \bar{r}_{u'})}{\sqrt{\sum_{i \in I \wedge r_{u,i} \neq \perp \wedge r_{u',i} \neq \perp} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{i \in I \wedge r_{u,i} \neq \perp \wedge r_{u',i} \neq \perp} (r_{u',i} - \bar{r}_{u'})^2}} \tag{1}$$

$\bar{r}$ denotes a mean value and σ denotes a standard deviation, and these are computed over the *co-rated items* only ($i \in I \wedge r_{u,i} \neq \perp \wedge r_{u',i} \neq \perp$).

Suppose we want to recommend to active user u_a one or more of a set of candidate items $T_a \subseteq I$. For example, T_a could be the set of movies showing this week at u_a's local multiplex. Using user-user similarity, $\rho_{u,u'}$, we can build a *user-based collaborative recommender* [3,13]. For each $i \in T_a$, it will predict active user u_a's rating for i, $\hat{r}_{u_a,i}$. It can do this using nearest-neighbour methods: from the users for whom $\rho_{u_a,u'}$ is greater than zero, it finds the k users $u' \in U$ who have rated i and who are most similar to u_a. The predicted rating is a weighted average of the neighbours' ratings for i [12]. The recommender suggests to the user the k' items $i \in T_a$ for which the predicted ratings $\hat{r}_{u_a,i}$ are highest.

But, given a ratings matrix we can equally well compute the similarity between one item and another, $i \in I, i' \in I, i \neq i'$, the *item-item similarity*, again using

Pearson correlation. In effect, this computes the similarity between two *columns* in a ratings matrix such as the one in the lower-left of Figure 1:

$$\rho_{i,i'} \hat{=} \frac{\sum_{u \in U \wedge r_{u,i} \neq \bot \wedge r_{u,i'} \neq \bot} (r_{u,i} - \bar{r}_i)(r_{u,i'} - \bar{r}_{i'})}{\sqrt{\sum_{u \in U \wedge r_{u,i} \neq \bot \wedge r_{u,i'} \neq \bot} (r_{u,i} - \bar{r}_i)^2} \sqrt{\sum_{u \in U \wedge r_{u,i} \neq \bot \wedge r_{u,i'} \neq \bot} (r_{u,i'} - \bar{r}_{i'})^2}} \quad (2)$$

In this case, the means ($\bar{r}$) and standard deviations (σ) are computed over the *users who have rated both items* ($u \in U \wedge r_{u,i} \neq \bot \wedge r_{u,i'} \neq \bot$).

Using item-item similarity, $\rho_{i,i'}$, it is possible to build an *item-based collaborative recommender* [6,13], although we use it for a different purpose in this paper. Before presenting the case-based group recommender in detail, we present the group recommender system against whose performance we will be comparing the new recommender.

3 Social Recommendations to Groups

For the comparison, we use a group recommender that we developed previously [11,8]. With real data and, in more recent work, with a larger dataset of artificial data, we showed that, relative to simpler approaches, our group recommender improves the accuracy of predicted group ratings and the precision of group recommendations, and that is why we use it here.

Let $G_a \subseteq U$ be an active group of users, in our case a group which intends to see a movie together. The goal is to recommend k' items from a set of T_a items. As Section 1 has mentioned, the system works by aggregation of ratings, as follows:

- For each $i \in T_a$ taken in turn, the recommender does the following:
  - It predicts a rating for item i, $\hat{r}_{u_a,i}$, for each individual group member $u_a \in G_a$. It does this using the user-based collaborative technique that we described in Section 2, i.e. it averages the ratings of i given by u_a's k most similar neighbours who have rated i.
  - It applies a function, designated dbr (which stands for *delegation-based rating*), to each predicted rating. The dbr function modifies $\hat{r}_{u_a,i}$ to take into account the *personality* of the user and the strength of connections between this person and other members of the group, which we refer to as their *trust*. In this way, not all the predicted individual ratings will contribute equally in the aggregation. We explain it in detail below.
  - It aggregates the individual predicted ratings into a single group rating $\hat{r}_{G_a,i}$. Possible aggregation functions include *least misery* (where the minimum is taken), and *most pleasure* (where the maximum is taken) [7]. We experimented with both before [9], and we found *most pleasure* to give better results, and so we adopt that here:

$$\hat{r}_{G_a,i} \hat{=} \max_{u \in G_a} \text{dbr}(\hat{r}_{u,i}, G_a) \quad (3)$$

- It recommends the k' items in $i \in T_a$ for which the predicted group ratings $\hat{r}_{G_a,i}$ are highest.

The delegation-based method recognizes that a person's opinions may be based in part on the opinions of other members of the group. The formula, which we explain below, is as follows:

$$\mathrm{dbr}(\hat{r}_{u,i}, G_a) \mathrel{\hat{=}} \frac{\sum_{v \in G_a \wedge v \neq u} t_{u,v} \times (r_{v,i} + \theta_{r_{v,i}} \times (v.pers - u.per))}{\sum_{v \in G_a \wedge v \neq u} t_{u,v}} \qquad (4)$$

In Equation 4, $t_{u,v}$ denotes the trust between u and v, which is a real number between 0.0 (no connection) and 1.0 (strong connection). In a real application, such as the Facebook movie group recommender that we have built [10], $t_{u,v}$ can be based on distance in the social network, the number of friends in common, relationship duration, and so on. As you can see, for user u in group G_a, we take into account the predicted ratings, $r_{v,i}$, for each other member of the group, $v \in G_a, v \neq u$, weighted by the trust between the two users, $t_{u,v}$. This follows [2], where a method for group recommendations using trust is proposed.

In Equation 4, $u.pers$ denotes user u's personality, also a real number between 0.0 (very cooperative) and 1.0 (very selfish). In our Facebook group movie recommender, users complete a personality test on registration. The details of the test are in [15]. In Equation 4, the rating given by another group member $r_{v,i}$ is increased or decreased depending on the difference in personality, $v.pers - u.pers$. This way, users with stronger personalities will contribute more to the final score. A user v with a positive opinion of i, i.e. where $r_{v,i}$ is greater than the mid-point of the ratings scale, will want to increase u's opinion of i; but if v has a negative opinion, i.e. where $r_{v,i}$ is less than the mid-point of the scale, then v will want to decrease u's opinion. We model this through a function θ:

$$\theta_{r_{v,i}} \mathrel{\hat{=}} \begin{cases} 5 & \text{if } r_{v,i} \geq mid \\ -5 & \text{otherwise} \end{cases}$$

where *mid* is the mid-point of the ratings scale, e.g. 3 on a five-point scale. We chose the constants (5 and -5) because the mean difference in personality values is 0.2 and therefore the impact of $\theta_{r_{v,i}}$ in Equation 4 will typically be 1 or -1.

4 A Case-Based Group Recommender System

Our new group recommender takes a case-based reasoning approach. There are two motivations for a case-based approach to group recommender systems.

– Firstly, groups tend to recur: the same group (with few variations) repeats activities together. Furthermore, group structures tend to recur: in the case of movies, for example, family outings comprising two adults and two children are common, as are parties of friends in the same age range.
– Secondly, group recommenders such as the one described in Section 3, have a 'one-size-fits-all' approach to the way they combine the predicted individual ratings. This ignores the possibility that different groups might have very different dynamics, not captured by a single theory expressed in a set of

formulae that apply globally. A case-based approach does not require us to commit to a model of social behaviour and to find a way to express that model in a set of formulae. Rather, aggregation of predicted ratings will be a lazy and local generalization (in the spirit of CBR) of the behaviours captured by the neighbouring cases in the case base.

4.1 Case Representation

Assume a case base CB in which each case $c \in CB$ records a previous group recommendation event. Each case will have the following structure:

$$\langle id_c, \langle G_c, T_c \rangle, i_c, \{r_{u,i_c} : u \in G_c\} \rangle$$

- id_c is a case identification number, used to distinguish the case from others.
- The *problem description* part of the case comprises:
  - $G_c \subseteq U$, the group of users who used the recommender previously. For each user $u \in G_c$, we will know demographic information such as u's age ($u.age$) and gender ($u.gender$); u's ratings, $r_{u,i}$ for some set of items; and u's personality value, $u.pers$. And, for each pair of users $u \in G_c, v \in G_c, u \neq v$, we will know the trust value, $t_{u,v}$.
  - $T_c \subseteq I$, the set of items that the users were choosing between. In our cases, these were the movies that were at the local multiplex on the occasion when this group used the recommender.
- The *solution* part of the case contains just $i_c \in T_c$, the selected item, i.e. the item that the group agreed on. In our cases, this is the movie that the group went to see together.
- The *outcome* part of the case [1,5] is a set of ratings. These are the actual ratings r_{u,i_c} that the members of the group $u \in G_c$ gave to item i_c: for example, after a group has gone to see their selected movie, group members return and rate the movie. In practice, some members of the group will not do this. In these cases, we can use $\hat{r}_{u,i_c}$ instead, i.e. the rating that a user-based collaborative recommender (Section 2) predicts the user $u \in G_c$ will assign to i_c. However, we have not so far evaluated empirically the consequences of using predicted ratings in place of actual ratings.

We now explain how this recommender makes its recommendations.

Step 1: Predict Individual Ratings

As usual, the goal is to recommend k' items from a set of items, $T_a \subseteq I$, to an active group of users, $G_a \subseteq U$. We can write the problem statement as $PS = \langle G_a, T_a \rangle$. The first step is to predict individual ratings $\hat{r}_{u,i}$ for each candidate item $i \in T_a$ for each member of the active group $u \in G_a$. We do this using a standard user-based collaborative recommender, as described in Section 2.

Later in the process, it may be necessary to insert virtual users into G_a, i.e. ones that are not real people. We explain when and why this happens at the appropriate time. But it simplifies the later exposition if we say now how we predict the ratings of items by virtual users. Since virtual users have no actual ratings, we cannot use the user-based collaborative recommender, as we do for real users. Instead, if u is a virtual user, its predicted rating for item i, $\hat{r}_{u,i}$, is the population average rating for i: $\frac{\sum_{u \in U \wedge r_{u,i} \neq \perp} r_{u,i}}{|u \in U \wedge r_{u,i} \neq \perp|}$.

Step 2: Retrieve Cases

The next step is to find the k'' *most similar cases*. We use $k'' = 3$. The similarity between a problem statement $PS = \langle G_a, T_a \rangle$ and a case $c = \langle id_c, \langle G_c, T_c \rangle, i_c, \{r_{u,i_c} : u \in G_c\} \rangle \in CB$, $\mathrm{sim}(PS, c)$, is calculated on the basis of group similarity:

$$\mathrm{sim}(\langle G_a, T_a \rangle, \langle id_c, \langle G_c, T_c \rangle, i_c, \{r_{u,i_c} : u \in G_c\} \rangle) \doteq \mathrm{gsim}_{cbr}(G_a, G_c) \qquad (5)$$

This means that in our work case similarity only takes the groups, G_a and G_c, into account; it does not take into account the items, T_a and T_c. T_c contains the items that G_c contemplated in the past, but T_a contains items that G_a are contemplating right now, e.g. movies that have just come to town. These sets may or may not overlap. If they do, we have the basis for a refinement to the similarity we could use in case retrieval. We leave this to future work.

We denote the group similarity by gsim_{cbr}, and we emphasize that this is a new definition, richer than definitions that we have used in other work [8]. In effect, it is a form of graph similarity: users are nodes; trust relationships are weighted edges.

In our definition of group similarity, we pair each user from the active group G_a with exactly one user from the group in the case G_c and vice versa. In other words, we will be finding a *bijection* from G_a to G_c. This raises a problem when comparing groups of different sizes, where a bijection is not possible. In such situations, we could simply say that $\mathrm{gsim}_{cbr}(G_a, G_c) = 0$. However, we did not want to do this. It might force the system to retrieve unsuitable cases. Consider a case base that just happens to contain many families of four (two adults, two children), no families of five, but many parties of five friends. If the active group is a family of five (two adults, three children), it is surely not appropriate to prevent retrieval of families of four and only retrieve parties of five friends.

To enable comparisons, this is the point, prior to computing similarity, that we insert additional virtual users into either G_a or G_c, whichever is the smaller, in order to make the groups the same size.

Now, we can define the group similarity measure. Consider any pair of equal-sized groups, G and G' and a bijection, f, from G to G'. The function f will map members of G to G', and so for any $u \in G$, we can compute the similarity, psim_{cbr}, to his/her partner $f(u) \in G'$. We will do this for each user and his/her partner, and take the average:

$$\mathrm{gpsim}_{cbr}(G, G', f) \doteq \frac{\sum_{u \in G} \mathrm{psim}_{cbr}(u, f(u))}{|G|} \qquad (6)$$

But, we also have trust values for each pair of users in G, and we can compute the similarities between each of these and the trust values for the corresponding pair of users in G'. Again we take the average (dividing by the number of pairs):

$$\mathrm{gtsim}_{cbr}(G, G', f) \hat{=} \frac{\sum_{u \in G, v \in G, u \neq v} \mathrm{tsim}_{cbr}(t_{u,v}, t_{f(u),f(v)})}{|G|^2 - |G|} \tag{7}$$

We combine gpsim_{cbr} and gtsim_{cbr} in a weighted average to obtain the following definition of the similarity between any pair of equal-sized groups, G and G', given a bijection f from G to G':

$$\mathrm{gsim}_{cbr}(G, G', f) \hat{=} \alpha \times \mathrm{gpsim}_{cbr}(G, G', f) + (1 - \alpha) \times \mathrm{gtsim}_{cbr}(G, G', f) \tag{8}$$

We currently use $\alpha = 0.5$.

This definition of gsim_{cbr} (Equation 8) uses gtsim_{cbr} (Equation 7), which uses tsim_{cbr}, the similarity between two trust values, which we have not yet defined. We use their range-normalized difference:

$$\mathrm{tsim}_{cbr}(x, y) \hat{=} \mathrm{rn\_diff}_t(x, y) \tag{9}$$

where

$$\mathrm{rn\_diff}_{attr}(x, y) \hat{=} 1 - \frac{|x - y|}{range_{attr}} \tag{10}$$

There is a problem, however. If one or both of u or v (Equation 7) is a virtual user, we will not have a trust value; similarly, if one or both of $f(u)$ or $f(v)$ is virtual. In these situations, we impute an average trust value between that pair of users, which empirically we found to be 0.05.

Equally, the definition of gsim_{cbr} (Equation 8) uses gpsim_{cbr} (Equation 6), which uses psim_{cbr}, the similarity between a person u in one group G and a person v in another group G', which we have not yet defined. We make use of their ratings, age, gender and personality values. Specifically, we combine local similarities into a global similarity. The local similarities are as follows. For the users' ratings, we use the Pearson correlation (Equation 1) but normalized to $[0, 1]$, denoted here by $\rho_{[0,1]}$. For gender, we use an equality metric:

$$\mathrm{eq}(x, y) \hat{=} \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

For ages and personalities, we use the range-normalized difference. Finally, the global similarity, psim_{cbr}, is simply an average of $\rho_{[0,1]}$, eq_{gender}, $\mathrm{rn\_diff}_{age}$ and $\mathrm{rn\_diff}_{pers}$.

Again we have the problem of virtual users, who do not have ages, genders, personalities, or ratings. If either user is a virtual user, we simply take psim_{cbr} to be the mid-point of the similarity range. Empirically, this is 0.6. This means that there is neither an advantage nor a disadvantage to being matched with a virtual user and, since everyone must be paired with someone, this seems appropriate.

While this completes the definition of $\mathrm{gsim}_{cbr}(G, G', f)$, it assumes that we give it a particular bijection, f, which pairs members of G with members of G'.

But, for the similarity, we want to consider *every* such bijection and settle on the *best one*, the one that gives the best alignment between the group members (their ages, genders, personalities, ratings) and the trust values. We must compute $\mathrm{gsim}_{cbr}(G, G', f)$ for each bijection.

Let $\mathcal{B}(A, B)$ denote all bijections between equal-sized sets A and B. For example, if A is $\{a, b, c\}$ and B is $\{x, y, z\}$, then one bijection is $\{a \mapsto x, b \mapsto y, c \mapsto z\}$, another is $\{a \mapsto y, b \mapsto x, c \mapsto z\}$, and so on. Our definition of the similarity of group G and G' is based on finding the bijection, out of all the possible bijections, that maximizes $\mathrm{gsim}_{cbr}(G, G', f)$:

$$\mathrm{gsim}_{cbr}(G, G') \hat{=} \max_{f \in \mathcal{B}(G,G')} \mathrm{gsim}_{cbr}(G, G', f) \tag{12}$$

Think of this as finding the pairing that maximizes total similarity. It does mean that a person in G might not be paired with the person who is most similar in G': it optimizes total similarity (over all group members and all trust values).

If G (and G') are of size n, then there are $n!$ bijections, and all must be considered. There is cause to be concerned whenever a computation requires consideration of $n!$ objects, because of the way that factorial grows with n. But, fortunately, the groups that most recommenders will deal with will be small enough to keep this manageable. For example, of 525 movie-going events reported to us through a Facebook poll, 21 were of size seven or a little above seven. Those that were of size seven would require consideration of $7! = 5040$ bijections, which remains manageable. If there are group recommenders where the number of bijections becomes too large, then some sort of sampling or greedy heuristic can be used, with the cost that the optimal bijection might be missed.

Step 3: Reuse Cases

At this point, we have explained our similarity measure, which is used to retrieve the k'' most similar cases. We must now explain how we reuse the cases that we have retrieved. To simplify the explanation, we will first consider the reuse of a single retrieved case, denoted $c = \langle id_c, \langle G_c, T_c \rangle, i_c, \{r_{u,i_c} : u \in G_c\} \rangle$.

Immediately, there is an issue that we must resolve. We want to predict G_a's ratings for each $i \in T_a$. But in case c, the selected item (e.g. the movie which the members of G_c went to see), was chosen from among T_c, which in most cases will not be equal to T_a: group G_a is going to the movies this week, whereas group G_c describes a previous outing to the movies, when it is probable that a different set of movies were on show. How can we transfer the contributions that the members of G_c made to the selection of $i_c \in T_c$ to the new situation where members of G_a must select an item from T_a?

The key to this is item-item similarity, which we described in Section 2. With item-item similarity, we can find the item $i \in T_a$ that is, for these users, most similar to $i_c \in T_c$. But there remains a problem. The Pearson correlation between two items i and i' is computed over the users who have rated both i and i' (Equation 2). There is no guarantee that there will be any user in either G_a or G_c who has rated both $i \in T_a$ and $i_c \in T_c$. But this is where the bijection f found

	Predicted ratings for candidate movies	
G_a	**Shrek**	**Hulk**
Ann	2	3
Ben	5	3
Col	2	4

	Actual ratings for the selected movie
G_c	**Twilight**
Dee	1
Edd	4
Flo	2

(a) No users in common

	Predicted & actual ratings		
Aligned users	**Shrek**	**Hulk**	**Twilight**
Ann+Dee	2	3	1
Ben+Edd	5	3	4
Col+Flo	2	4	2

(b) Using the bijection

Fig. 2. How item-item similarity is used

in Equation 12 can be used again. When comparing a rating from a user $u \in G_a$ for an item $i \in T_a$, we can use the rating $r_{f^(u),i_c}$ made by the corresponding user $f^(u) \in G_c$ for the item $i_c \in T_c$. It is by this means that we transfer the contributions that users in c made in their group decision to the group decision for $\langle G_a, T_a \rangle$.

But there is still a problem. The users $u \in G_a$ are unlikely to have a rating $r_{u,i}$ for the items $i \in T_a$, because T_a contains the candidate items that the group is choosing between. Instead, we use their predicted ratings $\hat{r}_{u,i}$, which we computed previously (Section 4.1) or, in the case of virtual users, the population average rating for the item.

Figure 2 contains an example. Suppose Ann, Ben and Col are in active group G_a, and that Dee, Edd and Flo are in case G_c. Figure 2a shows that we are unable to compute the item-item similarity between the selected movie from the case, Twilight, with the candidate movies, Shrek and Hulk. The movies have no users in common. For the active group, we have the predicted ratings for the candidate items; for the group in the case, we have the actual ratings for the selected movie. But suppose that, by the bijection, Ann maps to Dee, Ben maps to Edd and Col maps to Flo. Then, we can compute the item-item similarity between Shrek and Twilight by comparing Ann's predicted rating for Shrek with Dee's actual rating for Twilight, and Ben's predicted rating for Shrek with Edd's actual rating for Twilight, and so on. In effect, while there may be no users in these two groups who have rated both Shrek and Twilight, we are treating Ann & Dee as a 'single person' who has a rating for both Shrek (Ann's predicted rating) and Twilight (Dee's actual rating); see the Figure 2b.

We use Equation 2 to do this, but there are some changes. First, instead of computing the correlation over all users U, we compute it only over the users $u \in G_a$. Secondly, wherever the formula uses $r_{u,i}$, we now use u's predicted rating, $\hat{r}_{u,i}$; and wherever the formula uses $r_{u,i'}$, we now use the rating given by the user in G_c who corresponds to u, i.e. $r_{f^(u),i'}$.

We must still decide what to do if the groups are not of the same size. Consider the situation first where G_a is smaller than G_c . When we computed group similarity gsim_{cbr} earlier, we will have inserted extra virtual users into G_a. In this situation, we would not use G_a in place of U in Equation 2; rather, we would use the augmented version of G_a in place of U. That way, we can properly transfer the decision of the larger group to the smaller group: each person's contribution in the larger group is transferred to someone, either a real person from the smaller group or a virtual person who was inserted into the smaller group.

In the situation where G_a is larger than G_c, we will have earlier inserted virtual users into $G_c$in order to compute gsim_{cbr}. This time, however, we do use G_a in place of U. In other words, we compute the item-item similarity only on the ratings of the real people in G_a and their real counterparts in G_c. The virtual users were obviously not in reality present when G_c made its decision to consume i_c, so it makes no sense to transfer their contributions (i.e. none) to the decision-making of the smaller group G_a. This is achieved by simply computing item-item similarity over the real users and their counterparts, which is what Equation 2 will do if we use G_a in place of U. This does mean that, in these situations, there will be users in G_a whose opinions will be ignored (because they have no real counterparts in the smaller group, G_c).

So, we have explained how, given a retrieved case c, we can compute the similarity between i_c from c and each $i \in T_a$. We repeat this for each of the k'' retrieved cases. We can accumulate the item-item similarities and weight them by the group similarities. Formally, if C is the set of k'' cases, then the score for a candidate item $i \in T_a$ is $\sum_{c \in C} \text{gsim}_{cbr}(G_a, G_c) \times \rho_{i,i_c}$.

Step 4: Recommend Items

All the items in T_a have now received a score based on cumulating the similarities to the selected items in similar cases, weighted by the degree of similarity to those cases. So, finally, we recommend the k' items that have the highest scores.

5 Experiment

5.1 Group Recommender Dataset

We need a dataset with which we can evaluate our new system. We have built a social group recommender as a Facebook application [10]. But, at the time of writing, it cannot provide the volume of data that we need for conducting experiments. Unfortunately, neither are we aware of a public dataset for group recommenders. Hence, we created our own dataset. We have explained its construction elsewhere [8], and so we only summarize here.

We created our dataset from the MovieLens 1M dataset (`www.grouplens.org`), which gives us around 1 million ratings on a scale of 1 to 5 for around 6040 users for nearly 4000 movies. We created 100 groups from the MovieLens users, selecting group members at random but in such a way that everyone in a group falls into the same age range, and we ensured that there were at least 15 movies which are co-rated by all members of the group. When we create cases, these 15 movies will be the set T_c. We created 50 groups of size 2, 18 of size 3, 16 of size 4, 7 of size 5, 5 of size 6, and 4 where we took the size to be 7, this distribution being based on respondents to a Facebook poll that we administered.

The MovieLens dataset gives us the age, gender and ratings of each user. We had to impute personality values, which we did using the population distributions given in [15,14]. Similarly, we had to impute trust values between pairs of users in the same group. We took the trust between users u and u' to be the number of movies on whose ratings they agree as a proportion of the movies that either of them has rated. We take it that users agree if both have given the movie a rating above the ratings mid-point (which is 3) or if both have given the movie a rating below the ratings mid-point.

As we have explained, we have engineered matters so that, for each group, there is a set of 15 movies that all members of the group have rated, and we are treating these 15 movies as T_c, the set of movies that this group was choosing between. (Remember that T_c can be different for every group.) To create a case, we need to indicate which of these 15 movies the group will actually have chosen to go to see. For this, we got the opinion of four 'experts', two for each group. The experts voted on which three movies in T_c the group was most likely to select, placing movies into first, second and third position. Depending on the level of agreement between the experts, there might be ties for, e.g., first place, and so, although there were only three positions, the sets contained between three and five movies. We will designate this ordered set by E (for 'Expert') and we will use E_1 to mean movies in the first position in E, E_2 to mean movies in the first and second positions in E, and so on.

5.2 Evaluation Methodology

The dataset that we have created has 100 movie-going events, in other words 100 cases. We use a leave-one-out cross-validation methodology, where we remove each case in turn from the case base and present it to the recommenders.

We use three recommenders in these experiments: *Std*, *Soc* and *CBR*. *Std* is a simple group recommender: it uses the user-based collaborative recommender to predict the ratings each group member would give to the candidate items, and combines the ratings using the principle of *most pleasure*. *Soc* does the same but, before aggregation, it uses extra social data to modify individuals' predictions using the *delegation-based* method of Section 3. *CBR* is the new recommender, which uses cases to aggregate predicted ratings, which we described in Section 4.

Recall that each recommender recommends the top $k' = 3$ movies from the 15 candidates. Let R be the set of recommendations made by a particular recommender. Then we want to compare R with E from above. We computed total *success*@n for $n = 1, 2, 3$, where *success*@$n = 1$ if $\exists i, i \in R \wedge i \in E_n$ and is 0 otherwise. For example, when using *success*@2, we score 1 each time there is at least one recommended movie in the top two positions of E. We also computed total *precision*@n for $n = 1, 2, 3$, where *precision*@$n \hat{=} |\{i : i \in R \wedge i \in E_n\}|/n$.

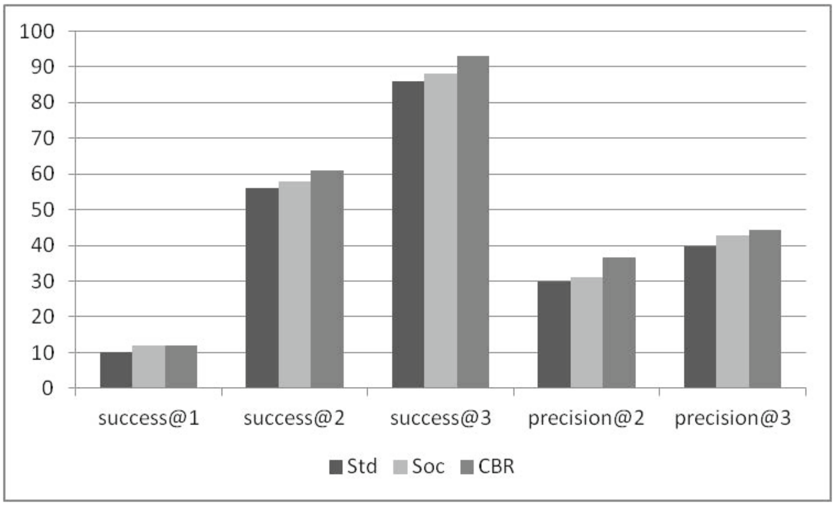


Fig. 3. Results of the experiment

For example, if no recommended movie is in the top two positions in E, then *precision*@2 $= 0$; if one recommended movie is in the top two positions in E, then *precision*@2 $= 0.5$.

5.3 Results

Figure 3 shows *success*@n for $n = 1, 2, 3$ and *precision*@n for $n = 2, 3$ (*precision*@1 = *success*@1 and is therefore not shown).

The first observation about the results is that, as n gets bigger, the results get better, e.g. *success*@2 results are better than *success*@1 results. This is not surprising: with bigger n, it is simply easier to make a recommendation that matches an expert judgement. The second observation is that results with *Soc* are better than results with *Std*: the use of the social information improves the quality of the recommendations. This is not a new result [11,8]. But what is new, our third observation, is the performance of the CBR system. It is never worse, and usually better, than both of the other systems. In detail, *CBR* has the same total *success*@1 (and *precision*@1) as *Soc*, just 12: it is very difficult for the systems to recommend the movie(s) the experts place in first position. But in all other cases, the CBR does better. For example, *Soc*'s *success*@2 $= 58$ but *CBR*'s *success*@2 $= 61$; and *Soc*'s *precision*@2 $= 31$ but *CBR*'s is 36.5. This shows the value of abandoning *Soc*'s single model of social behaviour, in favour of the lazy and local generalization that we obtain from the Case-Based Reasoning. We suspect that the differences would be even more marked in real datasets with more variability in the make-up of the groups.

6 Conclusions

We have described a new case-based group recommender system. It aggregates the predicted ratings of members of the active group but with reference to ratings of users in similar cases. A user-user similarity measure aligns members of the active group with members of the group in the case. The system uses an item-item similarity measure to transfer the contributions made to the group decision from the case to the corresponding users in the active group. One of its advantages is that preferences will be aggregated in different ways depending on how they played out in neighbouring groups, rather than according to a global, hypothesized theory of social interaction. This is borne out by our experiment, in which the CBR system is never worse, and is usually better, than a system that has a global model of group behaviour, expressed as a set of equations.

In our experiment, the selected item(s) in the cases are chosen by experts with knowledge of the actual ratings. So they are, in some sense, the absolutely best item(s). Therefore, it makes sense to run an experiment in which we see the extent to which the systems recommend such items.

But, matters are more complicated in practice. Suppose the recommender has recommended a movie to a group, and the group members have come back and rated that movie. We cannot simply retain this as a case in the case base. It may be suboptimal; it may not have been the best movie for this group. If we retain it, we will replay it in any future recommendation where it gets retrieved as a neighbour, where it may contribute to suboptimal decisions in the future.

In fact, this is not just a problem with CBR in group recommenders. It is a more general problem for the evaluation of group recommenders. It is very difficult to know whether they make good recommendations or not. If a user watches a recommended movie in a group and later gives it a low rating, this does not mean that the group recommender has done a poor job. It may even be that the group recommender predicted that this user would give a low rating. But the movie was recommended nonetheless, as it was judged to be the one that best reconciled the different tastes and personalities of the group members.

The implication is that, when group recommenders seek feedback from group members after recommended items have been consumed, they may need to solicit two types of feedback: the opinion of each individual user about whether the item satisfied him/her or not, but also the opinion of each individual user about whether the item satisfied the group as a whole or not.

Even if we get this more nuanced kind of feedback, it is not clear at this stage how to use it in evaluation of recommenders or in building case-based recommenders, not least because different group members may disagree on whether the recommendation satisfied the group or not. In case-based recommenders, the *outcome* part of the case might need to become much richer, to capture the opinions of the group members after they have consumed the item together, implying additional complexity in the kind of case-based recommender that we have described. This is a major issue for future work.

Other future work includes the use of datasets in which we explicitly arrange for the same group (or nearly the same group) to consume items together on a frequent basis, which can lead to a case base with more directly relevant cases in it. We hope too to gather more data from our Facebook group recommender and use this in future experiments.

References

1. Bridge, D.G.: The Virtue of Reward: Performance, Reinforcement and Discovery in Case-Based Reasoning. In: Muñoz-Ávila, H., Ricci, F. (eds.) ICCBR 2005. LNCS (LNAI), vol. 3620, pp. 1–1. Springer, Heidelberg (2005)
2. Golbeck, J.: Generating Predictive Movie Recommendations from Trust in Social Networks. In: Stølen, K., Winsborough, W.H., Martinelli, F., Massacci, F. (eds.) iTrust 2006. LNCS, vol. 3986, pp. 93–104. Springer, Heidelberg (2006)
3. Herlocker, J.L.: Understanding and Improving Automated Collaborative Filtering Systems. PhD thesis. University of Minnesota (2000)
4. Jameson, A., Smyth, B.: Recommendation to Groups. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 596–627. Springer, Heidelberg (2007)
5. Kolodner, J.L.: Case-Based Reasoning. Morgan Kaufmann (1993)
6. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing 7(1), 76–80 (2003)
7. Masthoff, J.: Group modeling: Selecting a sequence of television items to suit a group of viewers. User Modeling and User-Adapted Interaction 14(1), 37–85 (2004)
8. Quijano-Sánchez, L., Bridge, D., Díaz-Agudo, B., Recio-García, J.A.: A case-based solution to the cold-start problem in group recommenders. In: Díaz Agudo, B., Watson, I. (eds.) ICCBR 2012. LNCS, vol. 7466, pp. 342–356. Springer, Heidelberg (2012)
9. Quijano-Sánchez, L., Recio-García, J.A., Díaz-Agudo, B.: An architecture for developing group recommender systems enhanced by social elements. International Journal of Human-Computer Studies (in press, 2012)
10. Quijano-Sánchez, L., Recio-García, J.A., Díaz-Agudo, B., Jiménez-Díaz, G.: Happy movie: A group recommender application in facebook. In: 24th International Florida Artificial Intelligence Research Society Conference, FLAIRS 2011 (2011)
11. Quijano-Sánchez, L., Recio-García, J.A., Díaz-Agudo, B., Jiménez-Díaz, G.: Social factors in group recommender systems. In: ACM-TIST, TIST-2011-01-0013 (in press, 2011)
12. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An open architecture for collaborative filtering of netnews. In: Procs. of the Conference on Computer Supported Collaborative Work, pp. 175–186 (1994)
13. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative Filtering Recommender Systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 291–324. Springer, Heidelberg (2007)
14. Schaubhut, N.A.: Technical Brief for the Thomas-Kilmann Conflict Mode Instrument. CPP Research Department (2007)
15. Thomas, K.W., Kilmann, R.H.: Thomas-Kilmann Conflict Mode Instrument. Tuxedo, N.Y. (1974)

A Case-Based Solution to the Cold-Start Problem in Group Recommenders

Lara Quijano-Sánchez[1], Derek Bridge[2],
Belén Díaz-Agudo[1], and Juan A. Recio-García[1]

[1] Department of Software Engineering and Artificial Intelligence,
Universidad Complutense de Madrid, Spain
[2] Department of Computer Science, University College Cork, Ireland
{lara.quijano,jareciog}@fdi.ucm.es, d.bridge@cs.ucc.ie,
belend@sip.ucm.es

Abstract. We extend a group recommender system with a case base of previous group recommendation events. We show that this offers a potential solution to the cold-start problem. Suppose a group recommendation is sought but one of the group members is a new user who has few item ratings. We can copy ratings into this user's profile from the profile of the most similar user in the most similar group from the case base. In other words, we copy ratings from a user who played a similar role in some previous group event. We show that copying in this way, i.e. conditioned on groups, is superior to copying nothing and also superior to copying ratings from the most similar user known to the system.

1 Introduction

Restaurants; tourist attractions; vacation destinations; movies, music & TV when broadcast in shared spaces. All these are examples of items that can benefit from *group recommender systems*, i.e. recommender systems whose suggestions take into account the preferences of the members of a group of people who will consume the items together [4]. Group recommenders typically work by either (a) merging the recommendations that would be made to the group members, (b) aggregating the predicted ratings of the group members, or (c) constructing a group preference model from the preferences of the group members [4].

In this paper, in the context of movie recommendation to groups of friends, we consider a group recommender system that takes the second of these approaches. It runs, and aggregates the results of, a *single-person recommender system* for each member of the group. Specifically, it runs a *user-based collaborative recommender system* [3] to predict movie ratings for each member of the group. It finds a neighbourhood of users who have similar movie ratings to those of the active user; it predicts user ratings for candidate movies that neighbours have rated but which the active user has not rated. The group recommender aggregates the predicted ratings for each group member to arrive at ratings and thence suggestions that it can make to the group as a whole. Methods for aggregating ratings are reviewed in [5] and it is the *most pleasure* principle (see Section 3) that we use.

It is well-known that collaborative recommenders suffer from *cold-start problems* [3,13]. In particular, a user-based collaborative recommender finds it difficult to make good predictions for new users for whom it has few ratings: it cannot reliably find neighbours who have similar ratings to those of the new user. The group recommender inherits this problem too because it aggregates the predicted ratings for each group member. Solutions to the cold-start problem for single-person recommenders are summarized in [13]. Solutions include: non-personalized recommendations for cold-start users using population averages; intelligent ways to solicit more ratings (e.g. [2,11]); and hybrid recommenders that resort to content-based recommendations when there are insufficient ratings to make collaborative recommendations (e.g. [1,6]).

The contribution of this paper is to introduce and evaluate a *case-based reasoning* (CBR) solution to this problem. We use a case base in which each case records a previous group movie recommendation event. When a group requests a new recommendation but where one or more of the group members is in cold-start, we find a case that describes a previous recommendation event where there are users who are not in cold-start but who play similar roles in their group to the roles the cold-start users play in the active group. We copy ratings from the users in the case to the corresponding users in the active group and only then proceed to run the single-person recommender and to aggregate its results. It is natural to use a CBR approach because, in the movie domain and similar domains, similar events recur: the same group (perhaps with some small variations) repeats activities together; and some age, gender and personality distributions will tend to recur too (e.g. two adults with two children, or several friends in the same age range).

Case-based reasoning (CBR) has been used in recommender systems before (e.g. [12]) and explicit parallels between CBR and user-based collaborative recommenders have been drawn (e.g. [7]). But we are unaware of any previous use of CBR in group recommenders or in solutions to the cold-start problem.

Section 2 defines a single-person user-based collaborative recommender system; Section 3 describes two group recommenders that aggregate the predictions made for each group member by the single-person recommender; Section 4 describes how we have extended these group recommenders to use a case base of previous group recommendation events to solve the cold-start problem; Section 5 proposes systems against which the case-based system can be compared; Section 6 describes the dataset that we have used in our experiments; Section 7 presents our experimental method; Section 8 contains results; and Section 9 concludes and presents some ideas for future work.

2 Single-Person User-Based Collaborative Recommenders

As we have explained, our group recommender runs a user-based collaborative recommender for each person that is a member of the active group. Although the operation of user-based collaborative recommenders is well-known, we summarize it here in order to be explicit and to introduce some notation.

Suppose there are n users, $U = \{u : 1 \ldots n\}$, and m items (e.g. movies), $I = \{i : 1 \ldots m\}$. Let $r_{u,i}$ be the rating that user u assigns to item i. Ratings are on a numeric scale, e.g. 1 = terrible and 5 = excellent, but $r_{u,i} = \perp$ signals that u has not yet rated i.

Suppose we want to recommend to active user u_a one or more of a set of candidate target items $T_a \subseteq I$. For example, T_a could be the set of movies showing this week at u_a's local multiplex. The user-based collaborative recommender that we use works as follows [3,13]:

– For each $i \in T_a$,
  - The similarity between the active user u_a and each other user $u \neq u_a$ who has rated i, is computed using Pearson Correlation [3], ρ.
  - After computing the similarity between u_a and each other user u who has rated i, the k nearest neighbours are selected, i.e. the k for whom $\rho_{u_a,u}$ is highest. In our work, we use $k = 20$ and we only include neighbours for whom $\rho_{u_a,u} > 0$.
  - A predicted rating $\hat{r}_{u_a,i}$ for active user u_a and target item i is computed from the neighbours' ratings of i as follows:

$$\hat{r}_{u_a,i} \hat{=} \bar{r}_{u_a} + \frac{\sum_{u=1}^{k}(r_{u,i} - \bar{r}_u)\rho_{u_a,u}}{\sum_{u=1}^{k} \rho_{u_a,u}} \tag{1}$$

– Having computed $\hat{r}_{u_a,i}$ for each $i \in T_a$, the system recommends to the active user the k' items from T_a whose predicted ratings are highest. We use $k' = 3$.

3 Group Recommenders

Let $G_a \subseteq U$ be an active group of users, in our case a group who intend going to see a movie together. The goal again is to recommend k' items from a set of T_a items. We will do this by computing a predicted rating $\hat{r}_{G_a,i}$ for active group G_a and each target item $i \in T_a$ and then recommending the k' items in T_a that have the highest predicted ratings.

3.1 Standard Group Recommenders

As we have explained, a common approach to group recommendation, and the one that we follow, is to aggregate the predicted ratings of the members of the group, $\hat{r}_{u_a,i}$ for each $u_a \in G_a$ for the various i in T_a. Possible aggregation functions include *least misery* (where the minimum is taken) and *most pleasure* (where the maximum is taken). We experimented with both before [8], and we found *most pleasure* to give better results, and so we adopt that here:

$$\hat{r}_{G_a,i} \hat{=} \max_{u_a \in G_a} \hat{r}_{u_a,i} \tag{2}$$

We compute $\hat{r}_{G_a,i}$ for each $i \in T_a$ and recommend the k' with the highest aggregated predicted rating. We will designate this recommender by *Std*.

3.2 Social Group Recommenders

Our previous work showed an improvement in the accuracy of predicted group ratings by taking into account the *personality* of the users in the group and the strength of their connections, which we refer to as their *trust* [10]. We refer to our recommender that takes this extra social information into account as being *social* and the method it uses as being *delegation-based*

We obtain the personality of each user u, denoted $u.pers$, by making group members complete a personality test on registration with the recommender. The details of the personality test are in [15]. In a real application, such as the Facebook social group recommender that we have built [9], trust between users u and v $(u \in U, v \in U, u \neq v)$, $t_{u,v}$, can be based on distance in the social network, the number of friends in common, relationship duration, and so on.

Using the *most pleasure* principle again, we have:

$$\hat{r}_{G_a,i} \hat{=} \max_{u_a \in G_a} \text{dbr}(\hat{r}_{u_a,i}, G_a) \tag{3}$$

Here the *most pleasure* principle is not applied directly to individual predicted ratings, $\hat{r}_{u_a,i}$. The ratings are modified by the dbr function, which takes into account personality and trust values within the group G_a to compute what we call a delegation-based rating (dbr).

Space limitations preclude a detailed description of the operation of dbr but it is described in [10]. In essence, it is a weighted average of multiple copies of $\hat{r}_{u_a,i}$, one copy for each other member of $u \neq u_a$ in group G_a. The weights are based on the trust between u_a and u, $t_{u_a,u}$, and a value that is computed from the difference in their personalities, $u_a.pers - u.pers$.

The recommender recommends the k' items i from T_a for which $\hat{r}_{G_a,i}$ is highest. We will designate this recommender by *Soc*.

4 Using CBR in Recommenders for Users in Cold-Start

As we have explained, an active user with few ratings is said to be in cold-start. The problem that this causes for the kind of recommenders that we have been discussing is that it becomes difficult to find a reliable neighbourhood of similar users from which predictions can be made. One solution is to copy some ratings into the profile of the active cold-start user from a similar user who has additional ratings. Similarity in this case (i.e. for finding a user from whom ratings can be copied) would be measured using demographic information (age, gender, etc.) [13] because the active user has insufficient ratings to find a similar user using Pearson correlation, ρ. Let v be the user who is similar to u_a and from whom ratings will be copied. Then u_a obtains ratings for all items i that v has rated $(r_{v,i} \neq \perp)$ but that u_a has not $(r_{u_a,i} = \perp)$.

A group recommender can take the same approach when members of the group are in cold-start: prior to predicting individual ratings, it can augment the ratings profiles of group members who are in cold-start with ratings that are copied from the profiles of similar users. But in a group recommender, we can go further than using just demographic information for finding the most similar users from whom ratings will be copied. In our work, we investigate how to reuse ratings from similar users in similar groups in a case-based fashion.

4.1 Case Representation

Assume a case base CB in which each case $c \in CB$ records a previous group movie recommendation event. Each case will have the following structure:

$$\langle id_c, \langle G_c, T_c \rangle, i_c \rangle$$

– id_c is a case identification number, used to distinguish the case from others, but otherwise not used by the CBR.
– The *problem description* part of the case comprises:
  - $G_c \subseteq U$, the group of users who used the recommender previously. For each user $u \in G_c$, we will know demographic information such as u's age ($u.age$) and gender ($u.gender$); u's ratings, $r_{u,i}$ for some set of items; and u's personality value, $u.pers$. And, for each pair of users $u \in G_c, v \in G_c, u \neq v$, we will know the trust value, $t_{u,v}$.
  - $T_c \subseteq I$, the set of items that the users were choosing between. In our case, these were the movies that were at the local multiplex on the occasion when this group used the recommender.
– The *solution* part of the case contains just $i_c \in T_c$, the item that the group agreed on. In our case, this is the movie that the group went to see together.

Cases could also contain some of the numbers calculated when making the recommendation to the group, for example, the predicted individual ratings, $\hat{r}_{u,i}$ for each $u \in G_c$ and for each $i \in T_c$. Or, cases could also contain the *actual* ratings that users assign to item i_c. In other words, having gone to see movie i_c, users may come back to the system and give an actual rating, r_{u,i_c}. We leave the possible exploitation of this additional information to future work.

4.2 CBR for Cold-Start Users in Groups

We will summarize the process by which the case base is used for cold-start users. Details of the similarity measures will be given in subsequent sections. As usual, the goal is to recommend k' items from a set of items, $T_a \subseteq I$, to an active group of users, $G_a \subseteq U$. The recommender will recommend the k' for which the predicted group rating, which is aggregated from the predicted individual ratings, is highest. Of course, if none of the users in G_a is in cold-start, then the system will work either in the fashion described in Section 3.1 or in the fashion described in Section 3.2.

But suppose, on the other hand, that one or more members of G_a are in cold-start. We define this simply using a threshold, θ: a user u_a is in cold-start if and only if the number of items s/he has rated is less than θ, $|\{i : r(u_a, i) \neq \perp\}| < \theta$. In this case, we need to use the CBR. For each user who is in cold-start, we will copy ratings from the *most similar user in the most similar group* in the case base. The details follow.

Case Retrieval. We can write the problem statement as $PS = \langle G_a, T_a \rangle$. We will find the *most similar case*, c^*, in the case base:

$$c^* \hat{=} \arg\max_{c \in CB} \text{sim}(PS, c) \tag{4}$$

The similarity between a problem statement $PS = \langle G_a, T_a \rangle$ and a case $c = \langle id_c, \langle G_c, T_c \rangle, i_c \rangle \in CB$, $\text{sim}(PS, c)$, is calculated on the basis of group similarity:

$$\text{sim}(\langle G_a, T_a \rangle, \langle id_c, \langle G_c, T_c \rangle, i_c \rangle) \hat{=} \text{gsim}(G_a, G_c) \tag{5}$$

This means that in our work case similarity only takes the groups, G_a and G_c, into account; it does not take into account the items, T_a and T_c. T_c contains the items that G_c contemplated in the past, but T_a contains items that G_a is contemplating right now, e.g. movies that have just come to town. These sets may or may not overlap. If they do, we have the basis for a refinement to the similarity we could use in case retrieval. We leave this to future work.

Case Reuse. Next, for each user u_a in G_a who is in cold-start, we find the *most similar user* u^* in case c^* who has rated movies that u_a has not. Let G^* be the group of people described in case c^*. We find:

$$u^* \hat{=} \arg\max_{u \in G^* \wedge \exists i, r_{u_a,i} = \perp \wedge r_{u,i} \neq \perp} \text{psim}_{CB}(u_a, G_a, u, G^*) \tag{6}$$

In the case of more than one such user, we choose the one from whom we can copy the most ratings, i.e. the one who has most ratings for movies that u_a has not rated. Then, temporarily (for the purposes of making u_a's prediction for the items in T_a), we copy into u_a's profile the rating for each item i that u^* has rated ($r_{u^*,i} \neq \perp$) that u_a has not ($r_{u,i} = \perp$).

With each cold-start user's profile augmented in this way, we can then proceed to compute group recommendations in the fashion described in Section 3.1, which we will designate by *Std-CB*, or in the fashion described in Section 3.2, which we will designate by *Soc-CB*. But, it should now be less problematic finding neighbourhoods for the users who are in cold-start because they now have augmented user profiles.

4.3 The Most Similar Group

As we saw above, case retrieval in this system finds the most similar case to the problem statement, which is the one that contains the group that is most similar to G_a. This requires a definition of group similarity, gsim. We compute the similarity of any pair of groups, G and G', from the similarity of the users in the two groups, $\text{psim}_{CB}(u, G, u', G'), u \in G, u' \in G'$. We will define $\text{psim}_{CB}(u, G, u', G')$ in the next subsection.

So, the similarity of G to G' is the average similarity of each user u in G to his/her most similar user in G':

$$\text{gsim}(G, G') \hat{=} \frac{\sum_{u \in G} \text{psim}_{CB}(u, G, u^*, G')}{|G|} \tag{7}$$

where

$$u^* \hat{=} \arg\max_{u' \in G'} \text{psim}_{CB}(u, G, u', G') \tag{8}$$

Note that the mapping from users $u \in G$ to users $u' \in G'$ is not bijective, meaning we do not prevent two or more people from G being associated with the same user $u' \in G'$. This fact allows us to easily compare groups of different sizes without further complications. It does mean that, if two or more users from G_a are in cold-start, they may all copy ratings from the same user $u' \in G$. (We could have taken the option of requiring bijective mappings, either by only comparing equal-sized groups or by introducing 'virtual' users to make groups equal-sized, and we have done this in on-going work. But it seemed an unnecessary and costly complication in our work on cold-start.)

4.4 The Most Similar User

Our CBR solution to the cold-start problem in group recommenders requires a definition of the similarity between two users, u and u', in different groups, $\text{psim}_{CB}(u, G, u', G')$ where $u \in G$ and $u' \in G'$. This plays two roles in the CBR. First, as Section 4.3 explains, it is used in *case retrieval*, since *the most similar user* is part of the definition of *the most similar group*. Second, as Section 4.2 explains, it is used in *case reuse*, since ratings are copied to each cold-start user from his/her corresponding *most similar user* in the most similar case.

To define $\text{psim}_{CB}(u, G, u', G')$, the similarity between two users in groups, we make use of their ratings, their demographic information (age and gender) and the social information (personality and trust). Specifically, we compute local similarities for each of these, and then combine them into a global similarity.

The local similarities are as follows. For their ratings, we use the Pearson correlation but normalized to $[0, 1]$, denoted here by $\rho_{[0,1]}$. For gender, we use an equality metric and for ages and personalities, we use the range-normalized difference:

$$\text{eq}(x, y) \hat{=} \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases} \qquad \text{rn\_diff}_{attr}(x, y) \hat{=} 1 - \frac{|x - y|}{range_{attr}} \tag{9}$$

For trust values, we compute the average trust value between user u and all other members of his group, $v \in G, u \neq v$, which we will denote by $\bar{t_u}$. Similarly, we compute the average trust value for the other user, $\bar{t_{u'}}$, and we use rn_diff to give the similarity of these two values. We do the same for the standard deviations of the trust values, σ_{t_u} and $\sigma_{t_{u'}}$. The global similarity, psim_{CB}, is simply an average of $\rho_{[0,1]}$, eq_{gender}, rn_diff_{age}, rn_diff_{pers}, $\text{rn\_diff}_{\bar{t}}$ and $\text{rn\_diff}_{\sigma_t}$.

5 Other Recommenders for Users in Cold-Start

An obvious question is whether it makes a difference that our case-based solution to the cold-start problem in group recommenders works on a group basis at all.

Why copy ratings from the most similar user in the most similar group? Why not copy ratings simply from the most similar user in the case base as a whole? Or why not copy ratings from the most similar user known to the system? Systems that work in these different ways will be useful for comparisons in our experiments, hence we define both of these more precisely now.

Consider the set of users who appear in at least one case in the case base:

$$U_{CB} \hat{=} \{u : \exists c = \langle id_c, \langle G_c, T_c\rangle, i_c\rangle \in CB \wedge u \in G_c\} \tag{10}$$

When trying to predict group G_a's rating for an item $i \in T_a$, then for any user $u \in G_a$ who is in cold-start, we could find, and copy ratings from, the most similar user in U_{CB}:

$$u^* \hat{=} \underset{u \in U_{CB} \wedge \exists i, r_{u_a,i} = \perp \wedge r_{u,i} \neq \perp}{\arg\max} \text{psim}_{U_{CB}}(u_a, u) \tag{11}$$

This is different from first finding the most similar case (in other words, the most similar group) and then, for each active user in cold-start, copying ratings from the most similar user in that group. Our case-based approach is conditioned on the groups; this alternative is not. Note that this alternative needs a new definition of the similarity between two people, $\text{psim}_{U_{CB}}$ in place of psim_{CB}. Above, we were able to compute and compare the average and standard deviations of the trust values between a user and all other members of his/her group. In this new setting, this no longer makes sense, since we are ignoring the groups. Hence, the global similarity $\text{psim}_{U_{CB}}$ will be the average of just $\rho_{[0,1]}$, eq_{gender}, rn_diff_{age} and rn_diff_{pers}. We will designate this recommender by *Std-UCB* (where it works in the fashion described in Section 3.1) and by *Soc-UCB* (where it works in the fashion described in Section 3.2).

The second of our two alternative cold-start recommenders ignores the case base altogether. It simply finds, and copies ratings from, the most similar user in U (the entire set of users), wholly ignoring whether they have previously participated in group recommendations or not. Hence,

$$u^* \hat{=} \underset{u \in U \wedge \exists i, r_{u_a,i} = \perp \wedge r_{u,i} \neq \perp}{\arg\max} \text{psim}_{U}(u_a, u) \tag{12}$$

Note that for the experiments in this paper, this requires yet another definition of the similarity between users, psim_U. This is because we only have personality values for users who have participated in group recommendation events. Hence, the global similarity psim_U will be the average of just $\rho_{[0,1]}$, eq_{gender} and rn_diff_{age}. We will designate this recommender by *Std-U* (where it works as per Section 3.1) and by *Soc-U* (where it works as per Section 3.2).

6 Group Recommender Dataset

We need a dataset with which we can evaluate our case-based solution to the cold-start problem in group recommenders. We have built a social group recommender as a Facebook application [9]. But, at the time of writing, it cannot

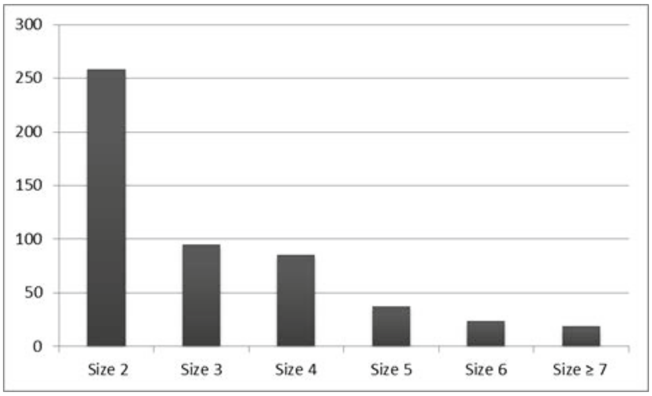


Fig. 1. Group sizes for 525 real movie-going events

provide the volume of data that we need for conducting experiments. Unfortunately, neither are we aware of a public dataset for group recommenders. Hence, we created our own dataset, and we explain how we did this here.

Base Dataset. We have used the MovieLens 1M dataset (`www.grouplens.org`). It gives us around 1 million ratings on a scale of 1 to 5 for around 6040 users for nearly 4000 movies. Each user has at least 20 ratings. The dataset also gives a small amount of demographic information about each user. In particular, we use the user's gender and age range (under 18, $18-24$, $25-34$, and so on).

Groups. We created 100 groups from the MovieLens dataset. Group members are chosen at random from all users in the MovieLens dataset but subject to the following restrictions:

– In a group, users are distinct (but a user may be in more than one group).
– In a group, we ensure that all the users are in the same age range.
– In a group, we ensure that there are at least 15 movies which are co-rated by all members of the group. When we create cases, these 15 movies will be the set T_c. These ratings themselves are withheld from the recommender, because it would not in general know a user's actual ratings for the movies that the group was choosing from.

We conducted a Facebook poll in which we asked respondents to tell us, for the last five times that they went to the cinema in a group, how large the group was. There were 105 respondents and so we learned the group size for 525 events (although we we cannot be certain that they were all distinct events). Figure 1 shows the distribution. We used the frequencies from this distribution to create our 100 groups. Hence, we have 50 groups of size 2, 18 of size 3, 16 of size 4, 7 of size 5, 5 of size 6, and 4 where we took the size to be 7.

Personality Values. We had to impute personality values to the users in the groups. The personality test that we have described in previous work is the

Thomas-Killmann Conflict Mode Instrument (TKI) [15]. Questions on the test reveal the extent to which a person uses each of five modes for dealing with conflict, including "competing", "compromising", "avoiding" and so on. These five modes can be summarized to give scores on two dimensions, "assertiveness" and "cooperativeness", from which we define a single numeric value, $u.pers$, in the range $[0, 1]$, where 0 signals a very cooperative person and 1 signals a very selfish person [10].

To impute personalities to users in our dataset, we make use of the population norms that the TKI Technical Brief provides [14]. We randomly give to each user five scores, one for each mode, based on the distributions given in the Brief. We calculate $u.pers$ from these.

We recognize that this is imperfect. Although the distribution of the five modes among our users will reflect the distribution in the population, the distribution within groups may not reflect reality. Because of the randomness, we might end up with a group of, for example, four very selfish people, where perhaps this rarely occurs in reality. We should be able to take a more informed approach in the future, once our Facebook application generates more data.

Trust Values. As we have discussed, in our Facebook application, trust is computed from Facebook data (distance in the social network, etc.), but that is not available to us for the users in the MovieLens dataset. Rather than simply imputing trust values at random, we have chosen to base them on ratings. For these experiments, the trust between users u and u' is the number of movies on whose ratings they agree as a proportion of the movies that either of them has rated. Agreement here is defined quite loosely: they agree if both have given the movie a rating above the ratings mid-point (which is 3) or if both have given the movie a rating below the ratings mid-point. The formula is as follows:

$$t_{u,u'} \hat{=} \frac{|\{i : (r(u,i) > 3 \wedge r(u',i) > 3) \vee (r(u,i) < 3 \wedge r(u',i) < 3)\}|}{|\{i : r(u,i) \neq \perp \vee r(u',i) \neq \perp\}|} \tag{13}$$

Hence, in our dataset, trust is based on the degree of shared taste.

This does not mean that, when psim_{CB} combines $\rho_{[0,1]}$ with $\text{rn\_diff}_{\bar{t}}$ and $\text{rn\_diff}_{\sigma_t}$, it is counting the same shared ratings twice. $\rho_{[0,1]}$ compares ratings between members of different groups (*inter-group*); it aligns a person in one group with someone in the other group who has the same tastes. But $\text{rn\_diff}_{\bar{t}}$ and $\text{rn\_diff}_{\sigma_t}$ compare ratings within groups (*intra-group*) to give trust values, which are then compared between groups; they align a person in one group with someone who has similar trust relationships in the other group.

The Chosen Movie. So far, we have described how we have created 100 groups. As we have explained, we have engineered matters so that, for each group, there is a set of 15 movies that all members of the group have rated (although we withhold the ratings from the recommender), and we are treating these 15 movies as T_c, the set of movies that this group was choosing between. (Remember that T_c can be different for every group.) To create a case, we need to indicate which of these 15 movies the group will actually have chosen to go to see. But we cannot ask random groups of MovieLens users to work out which of their 15 candidate movies they would have gone to see together.

We used four human 'experts' who were given all the information about a group's members G_c and the candidate movies T_c (including the actual ratings by the members of G_c for the items in T_c) and were asked to decide which of the movies the group would be most likely to settle on. Each expert evaluated 50 cases, hence each of the 100 groups was evaluated by two experts (not always the same two experts). Experts were asked to give an ordered list of the three movies from T_c that they thought the members of G_c would agree on.

Since each case is being decided by two experts, we needed a voting scheme to reconcile their judgements. A movie that an expert placed in first position was given three votes; a movie placed in second position was given two votes; and a movie placed in third position was given one vote. By adding up and ranking movies by their votes, we obtain a final ordered list of the movies that G_c would be most likely to see. For example, if both experts placed a movie $i \in T_c$ in first place, then it would receive six votes and would come first in the final combined ordering. But if one expert placed i in first position and $j \neq i$ in second position, but the other expert placed them in the opposite order, then both get five votes. The final ordered set will contain a minimum of three movies (where the experts agreed on the same set of three movies from T_c) and a maximum of six movies (where the two experts disagreed entirely). In fact, the latter never happened; final ordered sets are roughly evenly-split between those of size three and those of size four, plus a handful of size five. We will designate this ordered set by E (for 'Expert') and we will use E_1 to mean movies in the first position in E, E_2 to mean movies in the first and second positions in E, and so on.

7 Evaluation Methodology

The dataset that we have created has 100 movie-going events, in other words 100 cases. We use a leave-one-out cross-validation methodology, where we remove each case in turn from the case base and present it to the recommenders. We compare their recommendations with the judgements of the experts.

We use eight recommenders in these experiments: *Std*, *Soc*, *Std-CB*, *Soc-CB*, *Std-UCB*, *Soc-UCB*, *Std-U* and *Soc-U*. *Soc* (social) indicates that, before aggregation, the recommender uses extra social data to modify individuals' predictions using the *delegation-based* method of Section 3.2, whereas *Std* (standard) indicates that they do not as in Section 3.1. The second part of the name, if there is one, indicates how the recommenders handle cold-start users. The four options here are: they do nothing for cold-start users; they copy ratings from the most similar user in the most similar case (*-CB*, Section 4); they copy ratings from the most similar user from any case (*-UCB*, Section 5); or they copy ratings from the most similar user in the whole dataset (*-U*, also Section 5).

Recall that each recommender recommends the top $k' = 3$ movies from the 15 candidates. Let R be the set of recommendations made by a particular recommender. Then we want to compare R with E from above, the ordered set

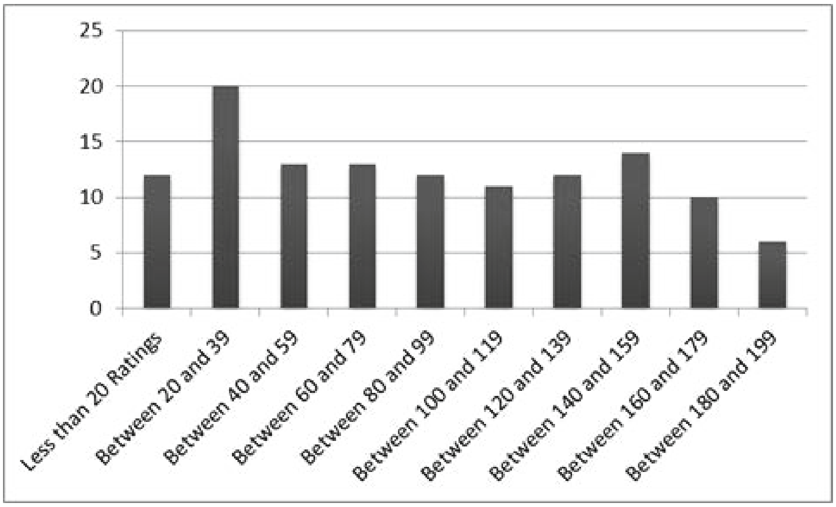


Fig. 2. Number of users in cold-start

of movies that the experts judged to be correct. We computed total *success*@n for $n = 1, 2, 3$, where *success*@$n = 1$ if $\exists i, i \in R \wedge i \in E_n$ and is 0 otherwise. For example, when using *success*@2, we score 1 each time there is at least one recommended movie in the top two positions of E. We also computed total *precision*@n for $n = 1, 2, 3$, where *precision*@$n \hat{=} |\{i : i \in R \wedge i \in E_n\}|/n$. For example, if no recommended movie is in the top two positions in E, then *precision*@$2 = 0$; if one recommended movie is in the top two positions in E, then *precision*@$2 = 0.5$.

We repeat the experiments with different cold-start thresholds. Figure 2 shows how many users are affected. We see that with $\theta = 20$, just over ten users are in cold-start; with $\theta = 40$, an additional twenty users are in cold-start; and then as θ goes up by 20, the number of users in cold-start grows by about an additional ten each time. (The threshold excludes the 15 ratings for T_a, which are withheld from the recommender.)

8 Results

Figure 3 shows *success*@n for $n = 1, 2, 3$ and *precision*@n for $n = 2, 3$ (*precision*@1 = *success*@1 and is therefore not shown) for cold-start threshold $\theta = 20$.

The first observation about the results is that, as one would expect, as n gets bigger, results improve but differences between systems become less pronounced: with bigger n it is simply easier to make a recommendation that matches an expert judgement. The next observation comes from looking at pairs of bars. The first bar in each pair is a system that does not use social data, and the second is one that does. Consistently throughout all our results, systems that use social data out-perform their counterparts that do not, which shows the value of using

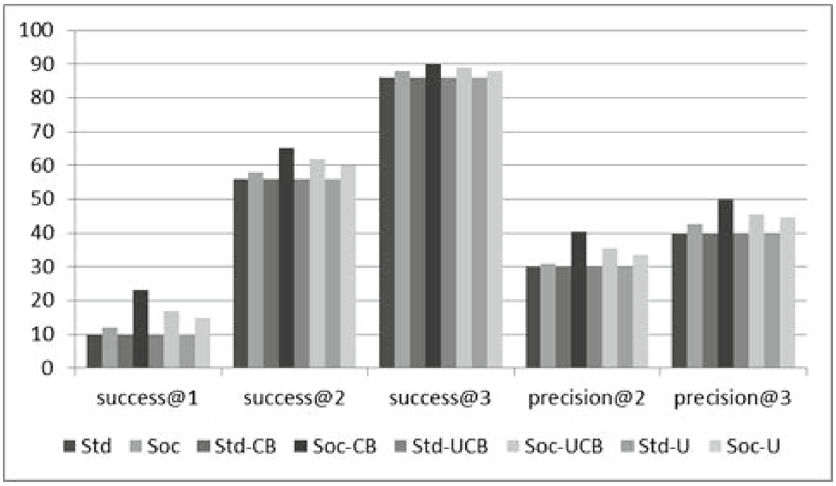


Fig. 3. Results for $\theta = 20$

personality and trust information. This is something we had already established in our previous work (e.g. [10,8]), but it is good to see the result confirmed on our new dataset. A final (and the most important) observation is that the *Soc-CB* system out-performs the *Soc-UCB* system, which out-performs the *Soc-U* system, which out-performs the *Soc* system. In other words, a cold-start strategy that is conditioned on groups (from cases) copies ratings in a more informed and successful way than strategies that copy without regard to groups, and copying ratings is more successful than having no cold-start solution at all.

We tried out a similar cold-start solution in the context of a single-person recommender, where a single active user seeks movie recommendations. If the active user was in cold-start, we copied ratings from a similar user in U. Interestingly, doing so made no or almost no change to the *success*@n and *precision*@n results (not shown here) across several definitions of similarity. We conclude that, for our movie data, conditioning on groups really does seem to be the most effective way to use this cold-start solution.

Figure 4 shows the effects of varying θ from 20 to 200. In other words, more and more users are regarded as being in cold-start and are given ratings from other users. We only show systems that use social data because, as we have already said, they are better. The results for *Soc* itself remain the same for all values of θ because this system has no cold-start strategy. For the other systems, we see that results improve and then fall off as θ increases. For example, for *Soc-CB*, results improve until $\theta = 100$. For this system, 100 is the cut-off point: users with fewer than 100 ratings are ones we should regard as being in cold-start. A higher threshold treats so many users as being in cold-start that the tastes of the active group are swamped by the ratings copied from other users, causing system performance to decrease. The graph is for *precision*@2 but we observed the same pattern of results for all other measures.

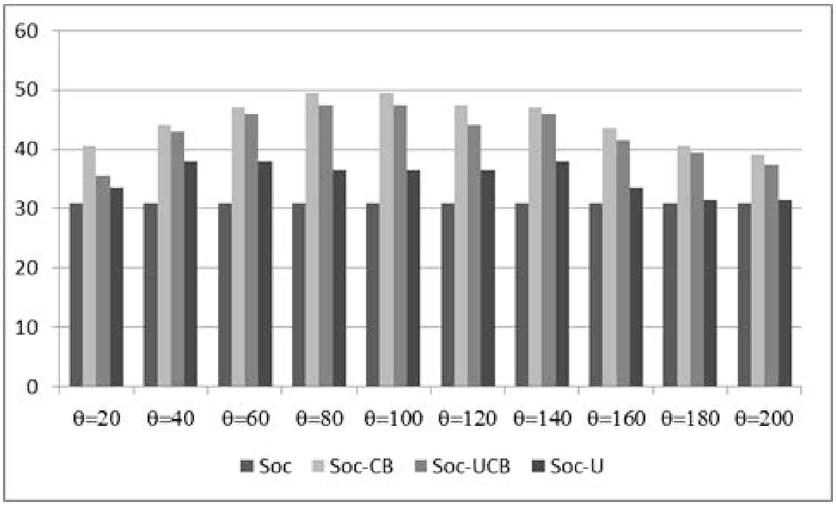


Fig. 4. Results for *precision*@2

9 Conclusions

We have presented a new solution to the cold-start problem in a collaborative group recommender. We use a case base of group recommendation events and copy ratings into the profile of users who are in cold-start from their most similar user in the most similar group in the case base. Our experiments on movie data show that, for users with fewer than 100 ratings, this strategy improves the quality of the group recommendations. The experiments also confirm, using new data, the results of our previous work, viz. a group recommender that uses social data, such as user personality and inter-personal trust, produces higher quality recommendations than one that does not use this data. A side-product of the research has been the construction of a dataset for group recommender research.

There is much that can be done to take this work forward. For us, the next step is to consider a case base in which we more explicitly arrange that there be cases (e.g. movie-going events) that involve groups whose members have a high degree of overlap with the members of the active group, so that we can experiment with the situation where the same group (or nearly the same group) consumes items together on a frequent basis. We also intend to consider richer case representations to take into account such things as timestamps, predicted and actual ratings from group members, and the dynamics of reaching a consensus (e.g. changes in group membership and changes in the selected item). We hope too to gather more data from our Facebook application and use this data as the basis for future experiments.

References

1. Balabanovic, M., Shoham, Y.: Fab: Content-based, collaborative recommendation. Comms. of the Association of Computing Machinery 40(3), 66–72 (1997)
2. Golbandi, N., Koren, Y., Lempel, R.: Adaptive bootstrapping of recommender systems using decision trees. In: Fourth ACM International Conference on Web Search and Data Mining (2011)
3. Herlocker, J.L.: Understanding and Improving Automated Collaborative Filtering Systems. PhD thesis. University of Minnesota (2000)
4. Jameson, A., Smyth, B.: Recommendation to Groups. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 596–627. Springer, Heidelberg (2007)
5. Masthoff, J.: Group modeling: Selecting a sequence of television items to suit a group of viewers. User Modeling and User-Adapted Interaction 14(1), 37–85 (2004)
6. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: Procs. of the Eighteenth National Conference on Artificial Intelligence (AAAI), pp. 187–192 (2002)
7. Sullivan, D.O., Wilson, D.C., Smyth, B.: Improving Case-Based Recommendation: A Collaborative Filtering Approach. In: Craw, S., Preece, A.D. (eds.) ECCBR 2002. LNCS (LNAI), vol. 2416, pp. 278–291. Springer, Heidelberg (2002)
8. Quijano-Sánchez, L., Recio-García, J.A., Díaz-Agudo, B.: An architecture for developing group recommender systems enhanced by social elements. International Journal of Human-Computer Studies (in press, 2012)
9. Quijano-Sánchez, L., Recio-García, J.A., Díaz-Agudo, B., Jiménez-Díaz, G.: Happy movie: A group recommender application in facebook. In: 24th International Florida Artificial Intelligence Research Society Conference, FLAIRS 2011 (2011)
10. Quijano-Sánchez, L., Recio-García, J.A., Díaz-Agudo, B., Jiménez-Díaz, G.: Social factors in group recommender systems. In: ACM-TIST, TIST-2011-01-0013 (in press, 2011)
11. Rashid, A.M., Albert, I., Cosley, D., Lam, S.K., McNee, S.M., Konstan, J.A., Riedl, J.: Getting to know you: Learning new user preferences in recommender systems. In: Proceedings of the 2002 International Conference on Intelligent User Interfaces, pp. 127–134 (2002)
12. Ricci, F., Arslan, B., Mirzadeh, N., Venturini, A.: ITR: A Case-Based Travel Advisory System. In: Craw, S., Preece, A.D. (eds.) ECCBR 2002. LNCS (LNAI), vol. 2416, pp. 613–641. Springer, Heidelberg (2002)
13. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative Filtering Recommender Systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 291–324. Springer, Heidelberg (2007)
14. Schaubhut, N.A.: Technical Brief For The Thomas-Kilmann Conflict Mode Instrument. CPP Research Department (2007)
15. Thomas., K.W., Kilmann, R.H.: Thomas-Kilmann Conflict Mode Instrument, Tuxedo, N.Y. (1974)

Opponent Type Adaptation for Case-Based Strategies in Adversarial Games

Jonathan Rubin and Ian Watson

Department of Computer Science,
University of Auckland, New Zealand
jrub001@aucklanduni.ac.nz, ian@cs.auckland.ac.nz
http://www.cs.auckland.ac.nz/research/gameai

Abstract. We describe an approach for producing exploitive and adaptive *case-based strategies* in adversarial games. We describe how *adaptation* can be applied to a precomputed, static *case-based strategy* in order to allow the strategy to rapidly respond to changes in an opponent's playing style. The exploitive strategies produced by this approach tend to *hover* around a precomputed solid strategy and adaptation is applied directly to the precomputed strategy once enough information has been gathered to classify the current *opponent type*. The use of a precomputed, *seed* strategy avoids performance degradation that can take place when little is known about an opponent. This allows our approach an advantage over other exploitive strategies whose playing decisions rely on large, individual opponent models constructed from scratch. We evaluate the approach within the experimental domain of two-player Limit Texas Hold'em poker.

1 Introduction

A *case-based strategy* selects actions based on previous decisions that have been successful when applied to similar situations in the past. An action selected by a case-based strategy can either be applied directly or can be modified/adapted to better account for the current situation. *Adaptation* is typically required when the current environment and similar past environments are no longer identical, but instead differ in particular ways.

Case-based strategies have been successfully applied within the domain of adversarial games [1–6]. In some gaming environments, making use of knowledge recorded about an adversary can dramatically improve overall performance. In this work, we present an approach that records information about particular *opponent types* and uses the information gathered to affect the selection of actions via adaptation.

Rather than employing adaptation to handle moderate differences in gaming environments, we apply adaptation based on the playing style of a current opponent. In this way we are able to generate case-based strategies that are both exploitive and adaptive i.e. they can rapidly respond to changes in the playing style of a current opponent and they do so in a way that attempts to improve overall performance.

To begin with we construct an offline model that captures information about a set of opponent types. The information contained within this model becomes the basis for informing future adaptation based on the type of opponent currently being challenged. The exploitive strategies produced by this approach *hover* around a precomputed solid strategy and adaptation is applied directly to the precomputed strategy once enough information has been gathered about the current opponent *type*.

Our approach has two major advantages over exploitive strategies that construct large opponent models in real-time:

1. Firstly, our approach relies on opponent *type* classification and not on the construction of a fully-fledged, real-time opponent model. As such, the time required to collect data to classify an opponent is substantially less than the time required to build an accurate opponent model from scratch.
2. Second, exploitive strategies whose playing decisions rely on detailed opponent models require an expensive exploration phase during which sparsely populated opponent models can negatively impact performance. On the other hand, *seeding* a game playing agent with a robust, precomputed strategy, avoids the performance degradation that can take place when little is known about an opponent.

We apply this approach within the adversarial game of poker. In particular, we present results in the domain of two-player, limit Texas Hold'em. We comparatively evaluate a collection of adaptation strategies against a range of both known (opponents used during construction of the offline model) and unknown opponents (novel opponents never before been challenged) and record the overall effect adaptation has on performance compared with applying no adaptation at all.

2 Related Work

In the domain of computer poker, exploitive strategies can be produced by performing either *implicit* or *explicit* opponent modelling. *Implicit opponent modelling* does not attempt to decipher the details of an adversary's strategy, but instead attempts to select an appropriate response from a range of strategies based on performance information against the opponent. On the other hand, *explicit opponent modelling* does concern itself with discovering details about an opponent's strategy and, in particular, any weaknesses within that strategy. This information is then used to alter the agent's own strategy in an attempt to improve overall performance. Johanson et al. [7] investigated exploitive strategies via implicit opponent modelling in the domain of two-player limit Texas Hold'em. In [8] individual imitation-based strategies were selected at runtime based on their performance against a particular opponent in both limit and no-limit domains. The work described in this paper differs from that in [7] and [8], as rather than implicitly exploiting opponents by selecting from a range of static strategies, exploitation instead occurs in a more explicit manner by modifying the details of a single strategy directly via adaptation, given a certain opponent type.

Alternative approaches that perform explicit opponent modelling construct exploitive strategies by starting with an empty opponent model and populating the model in real-time during game play. Two such efforts that use this approach are described in [9] and [10]. Both approaches use imperfect information game tree search in order to construct adaptive and exploitive strategies. They build specific models about an opponent and use these to inform exploitive betting actions. A disadvantage of this approach is that the opponent models required are typically very large and constructing the opponent model can initially take some time, which will negatively impact performance when little is known about the current opponent. The work described in this paper overcomes this problem by effectively seeding the agent with a precomputed, solid strategy and later applying adaptation to this strategy once enough information has been gathered about the current opponent's playing style. Moreover, a fully-fledged opponent model is not required to be constructed at runtime. Instead, all that is required is opponent *type* information, captured by a collection of summary statistics. Gathering this information at runtime is quick, resulting in rapid classification, which allows exploitive adaptation to take effect earlier.

Adaptation also plays an important role in the area of case-based planning [11]. Instead of constructing plans from scratch, case-based planning retrieves previously stored plans based on how similar they are to the current environment. Both case-based planning and case-based plan adaptation [12] are areas of research that have received considerable attention. Case-based plan adaptation has proved successful in adversarial, gaming environments, such as real-time strategy games [2, 1], where rather than simply re-using a previously stored plan, retrieved plans are modified to better suit the environment an agent currently finds itself in. The adaptation that we perform in this work differs from that of case-based plan adaptation as we do not rely on plans to determine an agent's actions. Moreover, case-based plan adaptation typically does not consider the specific type of opponent when determining how to adapt plans, instead adaptation is affected only by differences between the present environment and similar past environments. In this work the adaptation performed varies in response to the specific opponent type encountered.

3 Adaptation

Within a case-based strategy, case features are used to capture the current state of the environment and a case's solution dictates which actions are appropriate to apply, given the current state. Within our experimental domain of computer poker, case solutions are represented by probabilistic action vectors that specify the proportion of the time to select a particular action. A probabilistic action vector for two-player limit Texas Hold'em looks as follows: (f, c, r) and specifies the probability of making the following decisions:

Fold (f): When a player discards their personal cards and is out of the current hand.
Check/Call (c): When a player commits the minimum possible funds to continue playing. Typically regarded as a defensive action.
Bet/Raise (r): When a player commits extra funds to the pot. A bet or raise is regarded as an aggressive action.

Adaptation is applied directly to the case's solution vector. Adapting a solution vector simply consists of shifting probabilities from one action to another. By shifting probability values, solution vectors can be made more aggressive (*aggressify*) or more defensive (*defensify*). A vector is made more aggressive by reducing the probability of playing defensive actions, such as checking or calling, and increasing the probability of more aggressive actions, such as betting or raising. Consider the following example that *aggressify's* an initial probability vector by 10%. The vector involves three possible actions (*fold*, *call*, or *raise*).

$$(0.1, 0.4, 0.5) \xrightarrow{aggressify\{0.1\}} (0.1, 0.36, 0.54)$$

In the example above, 10% of the call portion of the vector has been shifted over to the *raise* portion, resulting in a strategy that will act more aggressively.

Solution vectors can also be made more defensive, as in the following example:

$$(0.1, 0.4, 0.5) \xrightarrow{defensify\{0.1\}} (0.1, 0.45, 0.45)$$

here the probability of performing an aggressive action is lessened in favour of performing a more defensive check or call action.

3.1 Offline Opponent Type Model Construction

Given that we have a straightforward procedure for adapting probabilistic solution vectors, we now describe the construction of an *offline opponent type model* that employs the adaptation procedure.

3.2 Aggression Response Trends

The adaptation procedure described above allows an existing solid strategy to lay the foundation for an eventual exploitive strategy. While the details of *how* the adaptation is performed are straightforward, it is less obvious exactly *what* adaptation should occur, i.e. should the action vector be *aggressified* or *defensified*, and if so, by how much? In order to answer these questions it was necessary to construct an initial model of what impact adaptation has on performance against a collection of opponents with diverse playing styles. We refer to the set of opponents used to construct the offline model as the *known* set of opponents.

Figure 1 shows how performance can be altered against certain opponent types by systematically varying the type and amount of adaptation applied to solution vectors. We refer to the performance trends produced in Figure 1 as *aggression trends*. The aggression trends capture important information about the effect adaptation has on strategy performance against particular types of players. The aggression trends in Figure 1 were derived by blindly adapting action vectors within the spectrum of -0.1 to +0.1, where a negative sign represents *defensifying* an action vector and a positive sign indicates *aggressifying* the vector. In other words, the leftmost point in each figure depicts the result of *defensifying* the action vectors by 10% and the right most point depicts the result of *aggressifying* vectors by 10% where all gradations in between vary by 1%. *Null adaptation* (i.e. an adaptation factor of 0) is represented by the middle point in the figures. The aggression trends shown in Figure 1 highlight the fact that different adaptation strategies are required against different types of opponents. In particular, the leftmost trend in Figure 1 depicts a response against a particular type of player that is inclined towards increased aggression. The middle trend depicts a situation where it is not appropriate to adapt solution vectors at all against this opponent as *null adaptation* performs best. Finally, the rightmost trend depicts the opposite situation to the first trend, where it is appropriate against this opponent to further *defensify* solution vectors.

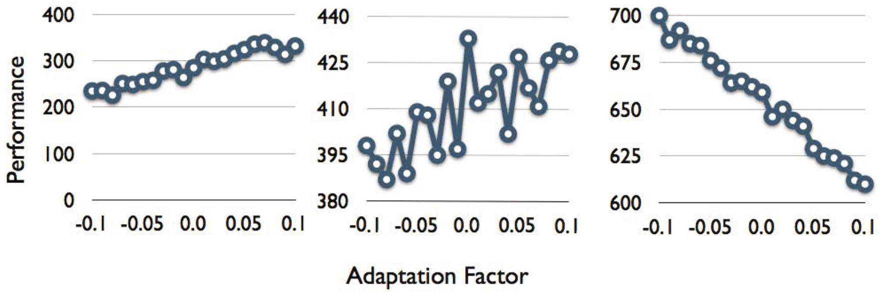


Fig. 1. A subset of aggression trends computed during an initial training phase. Adaptation was performed within the spectrum of -0.1 to +0.1, with stepped gradations of 0.01 in between.

3.3 Opponent Type

Obtaining knowledge of *aggression response trends* associated with particular players is beneficial in a number of respects. First of all, we have evidence that the adaptation procedure described previously can have a beneficial impact on overall performance. Secondly, when challenging the same opponent again we can

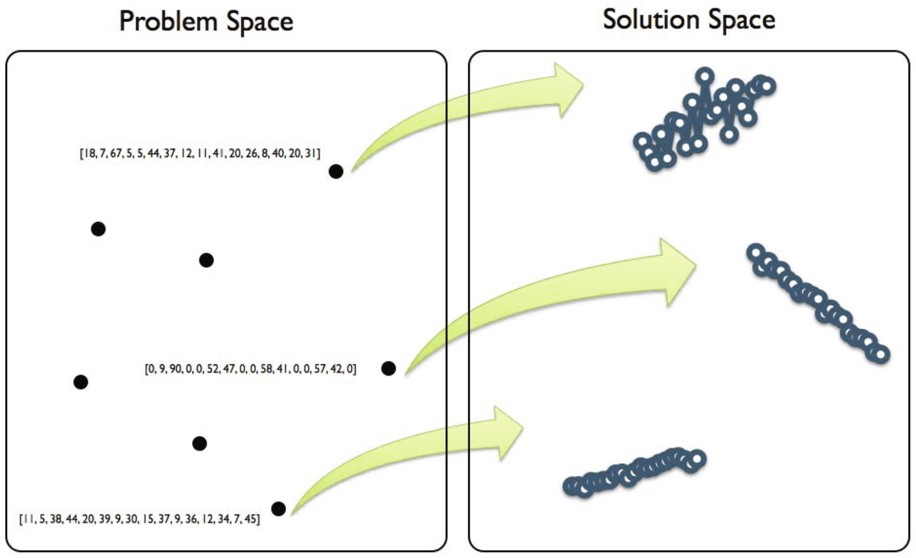


Fig. 2. Problem-Solution Space Mapping for Opponent Types

expect to improve overall performance by adapting solution vectors according to the resultant aggression trend. However, we would be severely restricted if we could only ever use the computed aggression trends against the exact same opponent. Instead, we wish to be able to use the information we have gathered to exploit further opponents that have never been encountered before. In order to do this we need to associate each aggression trend not with a particular opponent, but with a particular *type* of opponent.

There are various ways opponents could be classified to belong to a certain type. One approach that human online poker players employ to classify opponents is to keep track of statistical information about an opponent, such as the amount of money they voluntarily commit to the pot, the proportion of times they perform the *raise* action in a certain round, or their overall aggression factor. This information attempts to capture the general flavour of an opponent's style of play, such as how aggressive or defensive their strategy is. By recording action frequency information about an opponent, salient information can be captured about their particular style of play.

Within our current model, we capture information about *opponent types* by recording (for each of the four betting rounds) frequency information about *fold*, *check*, *call* and *bet* actions that were taken by the opponent. Therefore, in total each *opponent type* is described by a vector of 16 numerical values. By representing opponent types in this way we are able to perform similarity assessment between opponent types using Manhattan distance.

Taken together these opponent type action frequencies and their corresponding aggression trends compose an *offline opponent type model* that can be used to adapt solution vectors when challenging novel opponents by computing similarity between opponent type vectors. Figure 2 illustrates the idea of a mapping between opponent types (the problem space) and their corresponding aggression trends (solution space) that can be used to adapt vectors against that type of opponent.

4 Online Adaptation

4.1 Adapted Strategies

Once the *opponent type* model has been constructed we use the information contained within the model to inform how solution vectors should be adapted. Equation 1 uses information from the *opponent type* model to probabilistically select an appropriate adaptation factor for each hand of play:

$$\forall_{i \in 1 \ldots N}, \quad P(x_i) = \frac{max\{outcome(x_i) - outcome(x_{pivot}), 0\}}{\sum_i^N max\{outcome(x_i) - outcome(x_{pivot}), 0\}} \tag{1}$$

In Equation 1, x_i is an adaptation factor that dictates the proportion of actions that should be *defensified/aggressified* within the original vector, $outcome(x_i)$ refers to the result of applying adaptation factor x_i against this particular type of opponent, which is retrieved from the model, $outcome(x_{pivot})$ is the result of applying *null adaptation* against this opponent type (once again retrieved from the model) and finally, N refers to the total number of adaptation factors to consider per hand, i.e. not all adaptation factors are considered for use, only a subset of the top N factors.

In other words, Equation 1 selects adaptation factors based on how much they were able to improve upon not applying adaptation at all, as specified by the outcome values in our model. When $N = 1$, the adaptation factor that achieved the largest improvement in outcome is selected with probability 1.0, whereas when $N = 5$, the top 5 (profitable) adaptation factors are selected probabilistically. In the case where $\forall_{i \in 1 \ldots N}, P(x_i) = 0$, null adaptation is performed.

4.2 Adaptation during Game Play

During game play, betting frequencies (*fold*, *check*, *call* and *bet*) for all four rounds of play are recorded about the current opponent and used to classify them as a certain *type*. However, instead of keeping track of every action the opponent has taken, only a subset of actions are recorded that affect opponent type classification. This subset includes only the latest M hands of play, where M is some specified constant[1]. This ensures that our adapted strategies can respond

[1] Within our current implementation we use a value of $M = 300$.

to recent changes in an opponent's playing style. For the first M set of hands played against an opponent no adaptation is performed and actions are selected directly from the precomputed strategy. Once M hands have been played, adaptation can occur and an appropriate *aggression trend* is selected from the model by retrieving the opponent type with the most similar bet frequency information, using Manhattan distance.

5 Methodology

In order to evaluate our adapted strategies we require a series of opponents, both for training the initial model and for challenging once the model has been constructed. At the 2011 AAAI Computer Poker Competition (ACPC) a number of computer poker agents were submitted to the limit Texas Hold'em event. The agents submitted to the AAAI competition represent some of the top artificial computer poker players in the world, so it would be beneficial to evaluate opponent type adaptation against this set of opponents. While it is not possible to challenge the original agents submitted to the competition (as none of them are publicly available), we are able to make use of the publicly available hand history information to create *expert imitators* that attempt to imitate and generalise their style of play. By making use of the expert imitation based framework described by [4] and training on the subset of decisions made by a particular *expert* agent, we constructed a set of 20 opponents that imitate each of the agents submitted to the 2011 AAAI Computer Poker Competition. These *imitation*-based agents were used within our experimental set up.

These 20 opponents were challenged against 5 adaptation strategies given by Equation 1, where N varied from 1 to 5, plus 1 *null-adapted* strategy that involved no adaptation. From the set of original opponents two groups were created – a *known* set of opponents and an *unknown* set. Of the 20 original opponents 75% (15 opponents) were randomly assigned to the *known* group and were used to derive the model that would inform adaptation, the remaining 25% (5 opponents) were assigned to the *unknown* group that were left out of the model construction.

Results were gathered against both the *known* and *unknown* set of opponents, where all 5 adapted strategies and the null-adapted strategy played 10 seeded duplicate matches against each opponent. Each duplicate match consisted of 6000 hands in total, where 3000 hands are initially played, after which players switch seats and the same 3000 hands are played again, so that each player receives the cards their opponent would have received before. This type of duplicate match set up was used to decrease variance. In order to further decrease overall variance, common seed values were used for duplicate matches for each separate opponent match-up that occurred.

In the first experiment, adapted strategies are challenged against the *known* set of opponents. This represents a situation where previous knowledge about an opponent can inform present playing decisions and is akin to a scenario where a human poker player challenges a known opponent who they may have challenged before. Both human and artificial poker players are also required to challenge opponents they have never encountered before. The second experiment, involving the *unknown* set of opponents, represents this scenario, where no aggression trend or previous bet frequency information is known about the opponent and instead adaptation must occur by comparing how similar a novel opponent is to a previously *known* opponent style.

6 Experimental Results

Table 1 presents the results against the first set of known opponents (i.e. opponents whose exact aggression trends have been computed). The outcome and 95% confidence interval of each match is depicted in milli-big blinds per hand from the perspective of the row player (i.e. NoAdapt, Adapt1, Adapt2, . . ., Adapt5). Milli-big blinds record the average number of big blinds won per hand, multiplied by 1000. The opponent challenged is given as the column heading and the table is split into two sections (due to spatial limitations). The outcomes presented in the table are colour-coded as follows: grey cells represent the outcome against an opponent when no adaptation was performed. Darkly shaded cells indicate statistically significant improvements (green) or degradations (red), after applying adaptation. Lightly shaded cells indicate the observed outcome, after adapting case solutions, was statistically insignificant (once again light green for improvement and light red for degradation in performance).

Table 2 presents the results against the second set of unknown opponents (i.e. opponents where no aggression trend information is previously known). The same colour-coding scheme is used.

7 Discussion

A large proportion of Table 1 consists of darkly shaded green cells, indicating significant improvement in performance after solution adaptation. On average (the bottom right column in Table 1), all adaptation strategies ($N = 1 \ldots 5$) were able to improve on the performance of no adaptation and in all but one of these cases ($N = 2$) the improvement witnessed was statistically significant with 95% confidence. The results suggest that the adaptation information recorded within the opponent type model is appropriate and can be used to improve overall performance against a set of opponents that have been previously challenged.

In Table 2, less of the squares are darkly shaded green than in Table 1, however there is still more improvement (both significant and insignificant) witnessed than there is degradation in performance. The results in Table 2 show that even though no information was previously known about this set of opponents, our approach still results in an improvement in performance for all adaptation strategies ($N = 1 \ldots 5$) and in two of those cases ($N = 4$ and $N = 5$) the outcome is a statistically significant improvement in performance with 95% confidence. These results suggest that our approach has made an appropriate determination in regard to the adaptation to apply when challenging a novel opponent, given the similarity of the opponent to a known opponent.

Table 1. Opponent type adaptation results against first set of known opponents

	AAIMontybot	**Calvin**	**Feste**	**GBR**	**GGValuta**	**Hyperborean-tbr**	**POMPEIA**	**Patience**
NoAdapt	312 ± 4	548 ± 4	42 ± 2	433 ± 4	80 ± 2	25 ± 2	659 ± 3	56 ± 3
Adapt1	312 ± 3	553 ± 5	32 ± 2	429 ± 4	82 ± 2	37 ± 1	698 ± 4	57 ± 3
Adapt2	309 ± 4	549 ± 4	40 ± 3	417 ± 4	72 ± 2	28 ± 2	696 ± 3	41 ± 2
Adapt3	314 ± 4	552 ± 4	42 ± 3	421 ± 2	78 ± 3	31 ± 3	690 ± 3	57 ± 4
Adapt4	316 ± 3	566 ± 5	36 ± 2	412 ± 4	75 ± 3	26 ± 3	688 ± 2	44 ± 2
Adapt5	319 ± 4	575 ± 6	35 ± 2	409 ± 2	76 ± 3	21 ± 1	690 ± 3	49 ± 3
	RobotBot	**Slumbot**	**TellBot**	**Tiltnet**	**Tiltnet-Adaptive**	**Zbot**	**player-zeta**	**Average**
NoAdapt	285 ± 3	17 ± 3	886 ± 4	696 ± 4	673 ± 5	28 ± 2	593 ± 3	355 ± 3
Adapt1	324 ± 4	29 ± 2	900 ± 4	716 ± 4	682 ± 3	47 ± 3	596 ± 3	366 ± 3
Adapt2	330 ± 3	23 ± 4	888 ± 4	716 ± 4	674 ± 3	36 ± 2	591 ± 3	361 ± 3
Adapt3	327 ± 4	28 ± 3	892 ± 4	718 ± 2	684 ± 4	50 ± 2	590 ± 4	365 ± 3
Adapt4	326 ± 4	21 ± 2	897 ± 4	710 ± 2	705 ± 3	44 ± 2	590 ± 3	364 ± 3
Adapt5	334 ± 4	18 ± 2	888 ± 4	721 ± 4	680 ± 3	57 ± 1	600 ± 4	365 ± 3

Table 2. Opponent type adaptation results against second unknown set of opponents

	Entropy	**Hyperborean-iro**	**2Bot**	**LittleRock**	**Calamari**	**Average**
NoAdapt	436 ± 4	25 ± 2	61 ± 3	6 ± 2	-4 ± 2	105 ± 2
Adapt1	456 ± 5	25 ± 3	65 ± 3	-4 ± 3	-1 ± 1	108 ± 3
Adapt2	454 ± 4	21 ± 2	61 ± 2	-5 ± 2	1 ± 2	106 ± 2
Adapt3	448 ± 4	20 ± 2	56 ± 3	9 ± 3	8 ± 2	108 ± 3
Adapt4	463 ± 4	23 ± 2	57 ± 3	18 ± 2	1 ± 2	112 ± 2
Adapt5	462 ± 4	23 ± 1	70 ± 3	2 ± 2	-4 ± 2	111 ± 3

8 Conclusions and Future Work

In conclusion, we have presented an approach for adapting case-based strategies within adversarial environments. Rather than adapting case solutions based on differences between environmental states our adaptation procedure takes into account information about opponent types and adapts solutions in an attempt to exploit an opponent currently being challenged. This was achieved by first constructing an offline opponent type model that recorded appropriate aggression trends in response to a set of opponent types. The information captured within this model was later extrapolated during game play in order to inform probabilistic adaptation strategies. Our experimental results show that altering a precomputed strategy via adaptation was able to successfully improve overall performance, compared to not adapting the precomputed strategy. Moreover, the adaptation strategies produced were shown to be successful against both known and unknown opponents. In the future we wish to further investigate the mapping used between opponent types and aggression trends. While our results indicate that the mapping described is appropriate, it is likely that better mappings exist that could be used to further improve performance.

References

1. Ontañón, S., Mishra, K., Sugandh, N., Ram, A.: On-line case-based planning. Computational Intelligence 26(1), 84–119 (2010)
2. Sugandh, N., Ontañón, S., Ram, A.: On-line case-based plan adaptation for real-time strategy games. In: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, pp. 702–707 (2008)
3. Floyd, M.W., Esfandiari, B., Lam, K.: A case-based reasoning approach to imitating robocup players. In: Proceedings of the Twenty-First International Florida Artificial Intelligence Research Society Conference, pp. 251–256 (2008)
4. Rubin, J., Watson, I.: Similarity-Based Retrieval and Solution Re-use Policies in the Game of Texas Hold'em. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010. LNCS, vol. 6176, pp. 465–479. Springer, Heidelberg (2010)
5. Auslander, B., Lee-Urban, S., Hogg, C., Muñoz-Avila, H.: Recognizing the Enemy: Combining Reinforcement Learning with Strategy Selection Using Case-Based Reasoning. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 59–73. Springer, Heidelberg (2008)

6. Gillespie, K., Karneeb, J., Lee-Urban, S., Muñoz-Avila, H.: Imitating Inscrutable Enemies: Learning from Stochastic Policy Observation, Retrieval and Reuse. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010. LNCS, vol. 6176, pp. 126–140. Springer, Heidelberg (2010)
7. Johanson, M., Zinkevich, M., Bowling, M.H.: Computing robust counter-strategies. In: Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems (2007)
8. Rubin, J., Watson, I.: On combining decisions from multiple expert imitators for performance. In: IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, pp. 344–349 (2011)
9. Billings, D., Davidson, A., Schauenberg, T., Burch, N., Bowling, M., Holte, R.C., Schaeffer, J., Szafron, D.: Game-Tree Search with Adaptation in Stochastic Imperfect-Information Games. In: van den Herik, H.J., Björnsson, Y., Netanyahu, N.S. (eds.) CG 2004. LNCS, vol. 3846, pp. 21–34. Springer, Heidelberg (2006)
10. Schauenberg, T.: Opponent modelling and search in poker. Master's thesis. University of Alberta (2006)
11. Hammond, K.J.: Case-based planning: A framework for planning from experience. Cognitive Science 14(3), 385–443 (1990)
12. Muñoz-Avila, H., Cox, M.T.: Case-based plan adaptation: An analysis and review. IEEE Intelligent Systems 23(4), 75–81 (2008)

Exploiting Extended Search Sessions for Recommending Search Experiences in the Social Web

Zurina Saaya, Markus Schaal, Maurice Coyle, Peter Briggs, and Barry Smyth

CLARITY: Centre for Sensor Web Technologies
School of Computer Science and Informatics
University College Dublin, Ireland
{zurina.saaya,markus.schaal,maurice.coyle,
peter.briggs,barry.smyth}@ucd.ie
http://www.clarity-centre.org

Abstract. HeyStaks is a case-based social search system that allows users to create and share case bases of search experiences (called *staks*) and uses these staks as the basis for result recommendations at search time. These recommendations are added to conventional results from Google and Bing so that searchers can benefit from more focused results from people they trust on topics that matter to them. An important point of friction in HeyStaks is the need for searchers to select their search context (that is, their active stak) at search time. In this paper we extend previous work that attempts to eliminate this friction by automatically recommending an active stak based on the searchers context (query terms, Google results, etc.) and demonstrate significant improvements in stak recommendation accuracy.

Keywords: social search, community recommendation.

1 Introduction

Over the past few years there has been a growing interest in the application of case-based reasoning techniques to the type of *experiences* and *opinions* that are routinely captured as web content. The modern web is characterized by a proliferation of user-generated content. On the one hand we are all familiar with user generated content in the form of blog posts, online reviews, comments and ratings. On the other hand there is an equally rich tapestry of *implicit* experiential signals created by the actions of web users: the links people follow as their navigate, the results we select when we search, the pages we bookmark and share, and the movies and music we play. Collectively this content, and our actions as we consume and share it, encode our experiences and these experiences constitute the raw material for reuse as evidenced by recent work in the area of *WebCBR* and the *Experience Web* [16], [22].

We are also interested in the experience web, specifically in the search experiences of users and the opportunity to reuse these experiences in order to improve the effectiveness of mainstream web search. Modern search engines continue to struggle when it comes to delivering the right results to the right users at the right time. This is particularly acute in today's culture of sophisticated search engine optimization (SEO) techniques and so-called *content farming* strategies, which are designed to boost the rank of targeted results, often to the detriment of the individual searcher. Much has been written about the need for a more *personalized approach to web search*, see [5], [7], [17]. One particular approach to improving web search that has been gaining traction recently is evidenced by recent moves by mainstream search engines to introduce an element of so-called *social search* into their workflow, by incorporating results that have originated from the searcher's social network (e.g. Twitter, FaceBook, Google+), borrowing ideas from work on *collaborative web search*, see [20], [15], [2].

In this work we will focus on one particular approach to collaborative web search as presented previously in work by [21] and implemented in a system called HeyStaks. HeyStaks integrates collaborative web search into Google and Bing via a browser plugin. HeyStaks is further informed by the recent interest in curation on the web as evidenced by the emergence of content curation services such as Pinterest, Clipboard, ScoopIt etc; see [14]. Briefly, HeyStaks allows users to curate and share collections of search experiences called *staks*; each stak is essentially a case base of search experiences on a given topic. As a stak member searches, in the context of a given stak, any results they select are added to the stak. Then, in the future, when other members search for similar queries they may be recommended these results, in addition to the standard Google or Bing results. For example, consider a small group of college students planning a vacation together. One of the students might create a *travel* stak and share it with the others. Over time their vacation-related searches will add valuable queries and results to this stak and other members will benefit from these experiences by getting recommendations from this stak during their searches. We will review the operation of HeyStaks in more detail in Section 3.

We will focus on a specific problem faced by HeyStaks users, namely the selection of an appropriate context (stak) for their target search. Currently, HeyStaks users need to select one of their search staks at search time to ensure that any queries and selections are stored in the correct context. If users forget to select a stak, which they frequently do, then search experiences can be misrecorded, compromising stak quality and leading to poor recommendations in the future. Recently [19] proposed an alternative to manual stak selection by using textual CBR methods to recommend staks at search time. While results were promising they were not at the level necessary to use in practice. For example, to be practical it is necessary to be able to recommend an appropriate stak 80-90% of the time. However, the work of [19] achieved recommendation success rates of less than 60%. This initial solution based its recommendations on a straightforward term-overlap between the terms associated with the searcher's current query (e.g. query terms and/or terms from the results retrieved from the underlying mainstream search engine) and terms that reflect stak topics (e.g., terms associated with queries and or pages that make up the stak). The main contribution of this work is a two-part extension of the work of [19]. First we look at the potential to profile a searcher's query across multiple query instances (a so-called *search session*), since searchers often submit a series of related queries (and receive different result lists) before they find what they are looking for. Second, we describe a technique for weighting the relative importance of search session terms during recommendation. We go on to present a set of results based on live-user usage of HeyStaks to demonstrate the potential of these new techniques relative to the benchmark described by [19].

2 Related Work

There is a history of using case-based methods in information retrieval and web search. For example, the work of [18] looks at the application of CBR to legal information retrieval (see also [3]), and [6] describe a case-based approach to question-answering tasks. Similarly, in recent years there has been considerable research looking at how CBR techniques can deal with less structured textual cases. This has led to a range of so-called *textual CBR* techniques [13][23].

In the context of Web search, one particularly relevant piece of work concerns the *Broadway* recommender system [12] and a novel query refinement technique that uses case-based techniques to reuse past query refinements in order to recommend new refinements. Broadway's cases reference a precise experience within a search session and include a problem description (made up of a sequence of behavioural elements including a sequence of recent queries), a solution (a new query refinement configuration), and an evaluation (based on historical explicit user satisfaction ratings when this case was previously recommended). The work of [8] apply CBR techniques to Web search in a different way. Very briefly, their *PersonalSearcher* agent combines user profiling and textual case-based reasoning to dynamically filter Web documents according to a user's learned preferences.

More recently researchers have applied CBR concepts to web search. For example, [4] introduced *collaborative web search* (CWS) and the idea of reusing the search experiences of communities of like-minded searchers as implemented in the form of a system called *I-SPY*. In short, each community is associated with a case base of past *search cases*, with each case taking the form of a set of query terms (the problem specification) and a selected result (the problem solution). When presented with a new target query the I-SPY system retrieves a set of cases with similar queries and rank orders a set of corresponding results, recommending the top ranking results which have been frequently selected for similar queries by the community during past searches. This approach was recently expanded on by the work of [21] in the HeyStaks system, which allowed users to create and share their own case bases of search experiences. HeyStaks also extended search case representations to include snippet information, tags, and sharing and voting signals, in addition to simple query terms, in order to facilitate a more flexible approach to case similarity and retrieval. We will review HeyStaks in the following sections as it forms the basis of the work presented in this paper and then proceed to describe the stak recommendation task and evaluate our extended solution.

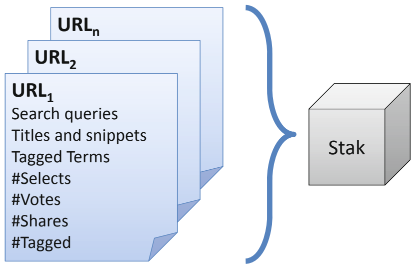


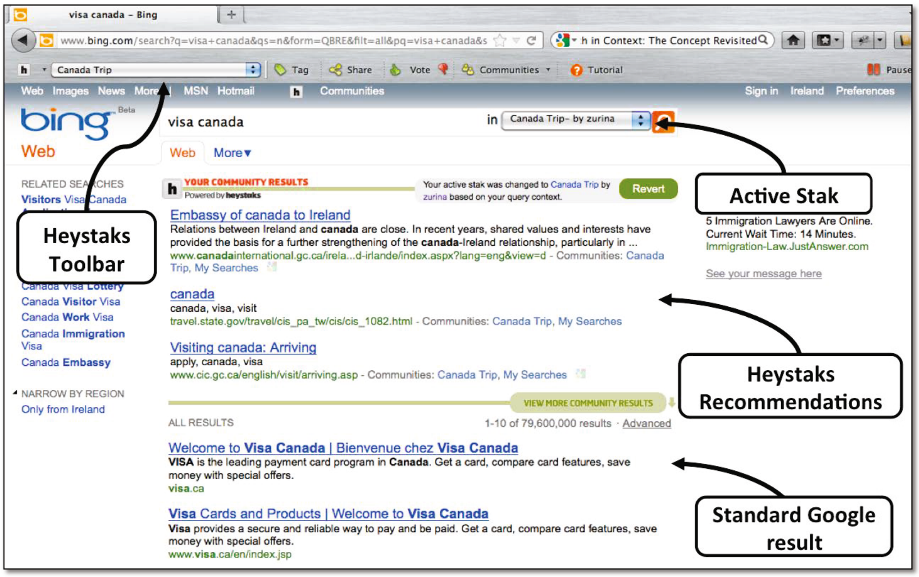
Fig. 1. Search Experiences in HeyStaks

3 HeyStaks: A Case-Based Approach to Social Search

HeyStaks combines a number of ideas to deliver an improved search experience for users. First and foremost it is based around the notion of *collaboration*, namely that it is useful for users to be able to collaborate as they search. Second it emphasizes the importance of *curation* and the willingness of interested users to create and maintain collections of topical content. Thirdly, it stresses the importance of *integration* by delivering social search within the context of an existing search service such as Google, by integrating with Google via a browser plugin. Bringing all of these ideas together HeyStaks allows communities of like-minded users to create and share curated repositories of search experiences, which deliver targeted recommendations to community members as they search, in addition to the organic results of Google, Bing or Yahoo.

The central idea in HeyStaks is the notion of a *search stak*. A stak is a named collection of search experiences. It is represented as a case base of search cases. Users can create and share their own search staks or they can join those created by others. Staks will typically be created around a topic that matters to a group of users, perhaps an upcoming vacation. At their core staks contain URLs for web pages that have been found during search sessions. Each URL is essentially the solution of a search case and is associated with a set of *specification* features that capture the different ways in which this case has been located in the past. For example, these features will typically include the terms of any queries that led to this URL being selected. Similarly, any snippet terms associated with the URL by some underlying search engine can also be used as part of the case specification. HeyStaks users can tag, rate, and share URLs too and this information will also be captured as part of a given URL's specification; see Figure 1. In this way each URL is associated with a rich set of *search experiences* that have led to its selection and these features can be used during future searches as a means to decide whether or not the URL should be recommended to the future searcher. The full details of staks and their search cases have been covered elsewhere (see [21]) and are beyond the scope of this paper.

Fig. 2 shows HeyStaks in operation. It shows a searcher, looking for Canadian visa information, benefiting from recommendations made by a group of friends


visa canada - Bing
www.bing.com/search?q=visa+canada&qs=n&form=QBRE&filt=all&pq=visa+canada&s
h in Context: The Concept Revisited
Canada Trip
Tag
Share
Vote
Communities
Tutorial
Pause
Web Images News More MSN Hotmail Communities
Sign in Ireland Preferences
bing Beta
Web
visa canada
in Canada Trip– by zurina
Web More▾
Heystaks Toolbar
Active Stak
Heystaks Recommendations
Standard Google result
Revert
VIEW MORE COMMUNITY RESULTS

RELATED SEARCHES
Visitors Visa Canada
Canada Visa Lottery
Canada Visitor Visa
Canada Work Visa
Canada Immigration Visa
Canada Embassy

NARROW BY REGION
Only from Ireland

YOUR COMMUNITY RESULTS
Powered by heystaks

Your active stak was changed to Canada Trip by zurina based on your query context.

Embassy of canada to Ireland
Relations between Ireland and **canada** are close. In recent years, shared values and interests have provided the basis for a further strengthening of the **canada**-Ireland relationship, particularly in ...
www.**canada**international.gc.ca/irela...d-irlande/index.aspx?lang=eng&view=d - Communities: Canada Trip, My Searches

canada
canada, visa, visit
travel.state.gov/travel/cis_pa_tw/cis/cis_1082.html - Communities: Canada Trip, My Searches

Visiting canada: Arriving
apply, canada, visa
www.cic.gc.ca/english/visit/arriving.asp - Communities: Canada Trip, My Searches

ALL RESULTS 1-10 of 79,600,000 results · Advanced

Welcome to **Visa Canada** | Bienvenue chez **Visa Canada**
VISA is the leading payment card program in **Canada**. Get a card, compare card features, save money with special offers.
visa.ca

Visa Cards and Products | Welcome to **Visa Canada**
Visa provides a secure and reliable way to pay and be paid. Get a card, compare card features, save money with special offers.
www.**visa**.ca/en/index.jsp

5 Immigration Lawyers Are Online.
Current Wait Time: 14 Minutes.
Immigration-Law.JustAnswer.com

See your message here

Fig. 2. The searcher is looking for visa information to enter Canada but Bing responds with results related VISA payment system. HeyStaks recognises the query as relevant to the searcher's *Canada Trip* stak and presents a set of more relevant results drawn from this stak.

who share a search stak called *Canada Trip*. The HeyStaks browser toolbar is shown, which provides access to key functionality such as tagging, rating, sharing etc. In this example, the user has selected the *Canada Trip* and a set of recommendations are shown alongside the standard Bing result list. These recommendations have been inserted into the standard Bing results page via the HeyStaks toolbar and the recommendations have been selected based on the past search experiences of other stak members. In this way our searcher benefits from the wisdom of people they trust on a topic that matters to them while they search. Once again the full details of this recommendation process are beyond the scope of this paper but they have been discussed in detail in [21].

One notable feature of this example is that the screenshot shows the *Canada Trip* stak being suggested to the user at search time; notice the message *"Your active stak was changed to Canada Trip"*. This is important for this paper because we are focused on automating this type of recommendation to the user, as opposed to recommending search results as per [21] currently these stak recommendations are very limited in HeyStaks. It is important for the user to be using the right stak for a given search if they are to benefit fully from HeyStaks' recommendations. In the past this has been a largely manual process, meaning the user was expected to select the stak at search time, which is far from ideal since it introduces an additional point of friction into the overall proposition.

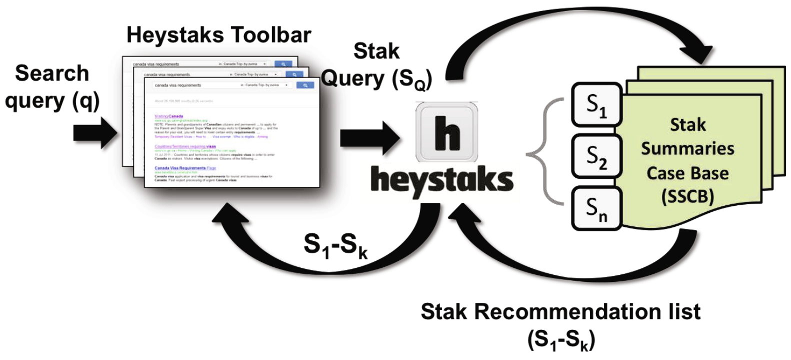


Fig. 3. An overview of the stak recommendation process

4 Recommending Search Case Bases

The recommendation of prior search results depends crucially on the recommendation of a suitable stak to the searcher at search time. The importance of selecting the correct stak is twofold. On the one hand, the detection of staks related to the current query provides the basis for result recommendations for the user and if this detection does not work properly then in all likelihood any recommendations made by HeyStaks to the user will be unlikely to be effective. But much more importantly, the current active stak provides a context for any search experiences such as selections, tagging, rating that the users make during the search session, etc. If the wrong stak is chosen, search experiences will be stored in an inappropriate stak, thus polluting this incorrect stak with irrelevant results, and preventing the case-base to gain maturity through learning based on the user's current session.

How then can we select and recommend a stak at search time? Prior work of [19] has looked at how to profile a stak and, given a current search query or partial search experience, how to identify those staks that are most likely to correspond to the current search needs. A summary of this stak recommendation process is presented in Figure 3. Briefly, for each user a new case base called the *stak summaries case base* (SSCB) is produced. In this case base each *stak case* corresponds to a single stak. In other words it is a case that is produced from a combination of the individual *search cases* in the corresponding stak case base. In effect, the specification part of each stak case is the combination of the specifications of the corresponding stak's search cases and the solution of the stak case is the corresponding stak id. In this way, stak cases are associated with the queries, snippets, URLs etc that are the basis of the search experiences within individual staks.

At search time we can use information about the searcher's current search context in order to retrieve a set of similar stak cases from the SSCB, or preferably automatically switch the user into a stak based on the most similar stak case. What information can be used to do this? In [19] a number of sources of information were considered including the searcher's current query, the snippet text for any results returned by the underlying search engine, the URLs of these results, and popularity information for the user's staks. This information can be used to build a *stak query* S_Q for the user and then we can score each stak case in the SSCB using a scoring metric such as that shown in Equation 1 to produce a ranked list of stak recommendations as per Equation 2.

$$Score(S_Q, S_C, SSCB) = \sum_{t \epsilon S_Q} tf(t, S_C) \times idf(t, SSCB) \tag{1}$$

$$RecList(S_Q, S_U, SSCB) = \begin{array}{l} SortDesc(Score(S_Q, S_C, SSCB)) \\ \forall S_C \epsilon S_U \end{array} \tag{2}$$

Thus the user's stak query S_Q is made up of a set of terms (that may include query and snippet terms, URls etc) and we can use TF-IDF [9] to calculate the relevance of a candidate stak S_C (from those staks the user is a member of, S_U) to S_Q; see Equation 1, which gives a higher weighting to S_Q terms that are common in S_C but uncommon across SSCB as a whole. By using different combinations of features for the stak query S_Q we can implement and test a number of different stak recommendation strategies as discussed by [19]. In what follows in this work we will focus on the combinations shown in Table 1.

Table 1. Strategies for Staks Recommendation

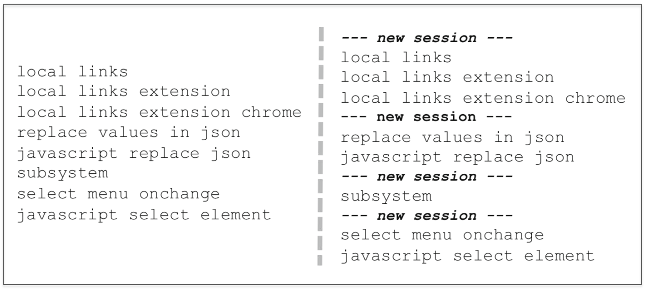
Strategy	Description
URL	URLs from the result-list
Snippet	Page titles and snippets from the result-list
Query	User's search query
URLSnippetQuery	Combination of URLs, search query, page titles and snippets
Popular	Most frequent stak for the user
URLSnippetQueryPopular	Combination of all strategies

5 Extending Stak Recommendation

The approach of [19] is limited in a number of important respects. Firstly each search is considered to be an atomic (singleton) session, which is not the way that people search in practice. Very often a searcher will require a few (related) queries to find what they are looking for (see [1]) and this means that we are not limited to using a single query (and its attendant data) for stak recommendation. In principle it is possible to assemble a stak query from the information contained in extended search sessions that span multiple queries, and in so doing provide a richer S_Q as the basis for recommendation. Moreover, the work of [19] did not consider the weighting of S_Q features/terms. But if we are constructing stak queries across multiple sessions then frequently recurring terms can be considered more important within S_Q, for example, and this information can be used to further enhance stak recommendation. We will develop and evaluate both of these ideas in the remainder of this paper.

5.1 Harnessing Extended Sessions

Given that many search sessions will span multiple queries it is natural to consider the possibility of using additional information from an evolving search session as the basis for stak recommendation. For example, while it might not be possible to reliably recommend the correct stak on the first query, the availability of an additional query (and its associated URLs and snippets) may improve recommendation quality. Thus, an alternative stak recommendation approach should build each stak query S_Q by aggregating the information that is available across related searches. Of course to do this it is necessary to be able to identify an extended search session. For the purpose of this work we use the method introduced by [11] (see Fig. 4), which effectively groups queries from a search log based on a simple term-overlap threshold.



```
local links
local links extension
local links extension chrome
replace values in json
javascript replace json
subsystem
select menu onchange
javascript select element
```

```
--- new session ---
local links
local links extension
local links extension chrome
--- new session ---
replace values in json
javascript replace json
--- new session ---
subsystem
--- new session ---
select menu onchange
javascript select element
```

Fig. 4. (Left) A sequence of queries submitted by one user. (Right) Sessions obtained based on shared query terms as per [11].

Thus, the stak query, as defined in the previous section, can now be adapted to cover sessions with multiple queries as $S_Q^* = \{S_Q^1, ..S_Q^n\}$. And in turn we can apply the stak recommendation techniques to these extended stak queries to investigate their impact on overall recommendation accuracy. In this way, as a search session evolves, the stak recommendation system has access to an increasing volume of relevant information as the basis for recommendation. The intuition is that this will improve recommendation quality, which we will come to test in due course.

5.2 Session-Based Term Weighting

The second limitation of the work presented in [19] is that the elements of a stak query are all assumed to be equally important. In other words there is no relative weighting of stak query terms even though, as we accumulate information across extended search sessions, there is an opportunity to introduce a weighting model into the stak recommendation process. Simply put, we can use term frequency information (calculated across a search session) as term weights, $W(t, S_Q^*)$, during recommendation. In this way the terms that frequently recur in the searchers queries (or in the snippet texts of the organic search results) are deemed to play a more important role during stak selection according to the new scoring function presented in Equation 3.

$$Score(S_Q^*, S_C, SSCB) = \sum_{t \epsilon S_Q^*} tf(t, S_C) \times idf(t, SSCB) \times W(t, S_Q^*) \qquad (3)$$

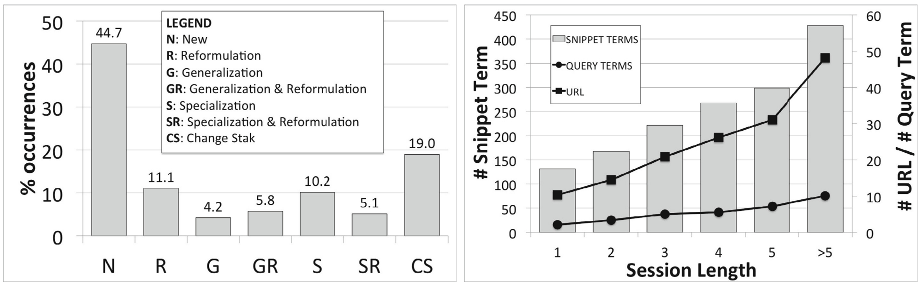
Once again the intuition is that the combination of extended search sessions as a richer source of query information combined with the availability of term weights should help to improve overall recommendation effectiveness. The next section tests this hypothesis in detail by comparing a variety of different recommendation strategies, using different sources of query information, and combining extended sessions and term weighting.

6 Evaluation

In the previous work of [19] a stak recommendation strategy was tested using only the information from singleton sessions. In what follows we will adopt a similar methodology but look at the effectiveness of recommendation when using the extended session and term-weighting techniques described above.

6.1 Dataset and Approach

The dataset comes from the HeyStaks anonymous usage logs based on a group of 28 active users, who are members of approximately 20 staks each, and who have each submitted at least 100 queries. For the purpose of this evaluation we limit our interest to only those sessions that are associated with at least one non-default search stak. This is important because it means that we can focus on search scenarios where the user has actively selected a specific stak during their search session. This selected stak is used as the ground-truth against which to judge our recommendation techniques; in other words, we are using information from the user's search session to make a stak recommendation and we compare this to the actual stak that they chose at search time. If their chosen stak matches the recommendation then the recommendation is deemed to be correct or successful. Arguably, this is quite a strict measure of success. After all, users sometimes join a number of related staks and even if the correct stak is not recommended a related one might be, which would probably still be useful to the searcher. However, we ignore these *near-miss* recommendations and treat them as failures for the purpose of this strict evaluation. According to the above criteria our test dataset covers 10,177 individual searches which have been grouped into 4,545 sessions, and the average each user has submitted 364 queries in 162 sessions.



(a) (b)

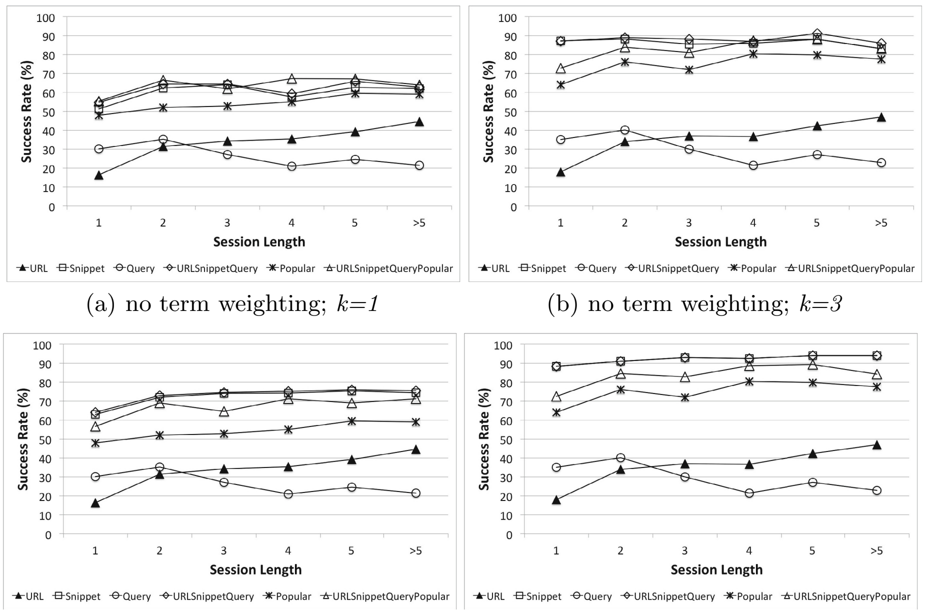
Fig. 5. Summary of session data; (a) query modification classes and (b) average unique URLs, query terms and snippet terms across different session length

6.2 Session Data

Before describing the results of the session-based evaluation it is useful to look at the relationship between consecutive queries, q_i and q_{i+1}, from the same user and session, to get a sense of the type of modifications and refinements that searchers tend to make within extended sessions. The following modification classes are based on those presented in [10]:

- *New* – initial query in a session.
- *Reformulation* – the current query is on the same topic as the previous query and both queries contain common terms. (add some terms and removed others from the previous query and both queries still have some common terms)
- *Generalization* – the current query is in the same topic as previous query, but the searcher seeking more general information (remove terms from previous query)
- *Specialization* – the current query is on the same topic as the previous query, but the search is now seeking more specific information. (adding new terms to the query)
- *Change Stak* –the current query identical with the previous query, but the stak has been changed

Fig. 5(a) shows the frequency of these query modification classes. We see that users frequently change their staks during a session (19% of the time) and that about 36% of the modifications involve changes to the terms in the query, which will ultimately lead to changes in the result-list returned to users, and so provides a strong indication that leveraging these extended sessions will deliver a richer source of information for stak recommendation. In Fig. 5(b) we present the unique number of query and snippet terms, and URLs, for different session lengths and we can see that there is a steady increase in the quantity of this information as session length grows. However, it is worth highlighting that, for example, doubling the session length, does not deliver twice the number of



(a) no term weighting; *k=1*

(b) no term weighting; *k=3*

(c) with term weighting; *k=1*

(d) with term weighting; *k=3*

Fig. 6. Success Rate for both approaches when $k = 1$ and $k = 3$

unique query or snippet terms or unique URLs; the reason being, of course, that minor modifications to the query will not radically change the new result-list.

6.3 Experimental Setup

We are primarily concerned with the accuracy of our three basic recommendation strategies *Query, Snippet, URL*, which differ based on the type of basic information used for a stak query. As per [19] we also consider the combination of these techniques and further combine them with the baseline stak *popularity* strategy, which recommends the stak most frequently used by the user. In total we look at 6 different strategies; see Table 1 earlier.

To evaluate these alternatives, we generate a recommendation list for each of the 4,545 sessions and compute the percentage of times (*success rate*) that the known active stak (ground-truth) is recommended in the top k stak recommendations (here we look at $k = 1$ and $k = 3$). We calculate this success rate across sessions of different lengths, both with and without term weighting.

6.4 Success Rate vs. Session Length

The results are presented in Fig. 6(a-d). Similar to the findings of [19], techniques such as *URL* and *Query* perform poorly, while *Popularity* and *Snippet*, and combinations thereof, perform well. This is true across all techniques and session lengths. However, there is a wide variety in absolute success rates.

As we can see from the graphs, increased session length generally implies improved success rates, especially when comparing sessions of length 1 (singleton sessions as per [19]) with sessions of length 2. It is particularly important to pay special attention to the $k = 1$ results because the ideal strategy for HeyStaks is to automatically switch the user into a correct stak, rather than present a set of (> 1) stak recommendations for the searcher to choose from. For $k = 1$, we can see for example that the best performing singleton strategies deliver success rates in the region of 51-56%; see, for example, Figure 6(a). By comparison, when we consider sessions of length 2 this tends to increase to 61-66%, a relative improvement of just under 20%. However, this rate of improvement is not sustained over longer search sessions and by and large the success rates for longer sessions are no higher than those for sessions of length 2. In other words, despite the availability of additional query, URL, and snippet data in longer sessions of length 3, 4, 5 etc., this extra data does not seem to help from a stak recommendation viewpoint. Another influencing factor is probably that searchers that *require* long sessions (> 2 queries) are probably fundamentally harder to satisfy than those that require just 2 queries and so success is likely to be more elusive. At the moment the test data contains a mixture of these data and so it is a matter for future work to further consider this explanation.

6.5 Term Weighting

The results of adding term weighting are presented in Figure 6(c, d) and further analysed in Figure 7. One of the best performing strategies when using term weighting, *URLSnippetQuery*, achieves a success rate of more than 70% for session lengths > 2, at $k = 1$); see 6(c) for example. This is an improvement of up to 17% in comparison to the results without term weighting.

Figure 7 looks at the relative performance of the three best performing techniques using term weighting and compares them to their corresponding results without term weight, across different session lengths; we focus on $k = 1$ recommendations only in this instance but comparable results are found for $k = 3$. The results show that the relative improvement due to term weighting falls off sharply with increasing session lengths. For example,*Snippet* achieves a success rate of about 52% for sessions of length 1, rising to about 62% for sessions of length 2, with term weighting is not used; see Figure 6(a). When term weighting is used these success rates are 63% and 72%, respectively, leading to relative improvements of about 23% and 11% as shown in Figure 7. These relative improvements continue to decline for *Snippet* as session lengths increase, as can be seen in Figure 7, and similar relative improvements, and subsequent declines, are also observed for *URLSnippetQuery* and *URLSnippetQueryPopular* as indicated.

These results help to clarity that, in combination, term weighting and extended sessions do have a positive impact on overall success rates. In absolute terms this combination is beneficial but scale of the relative benefit decreases with session length.

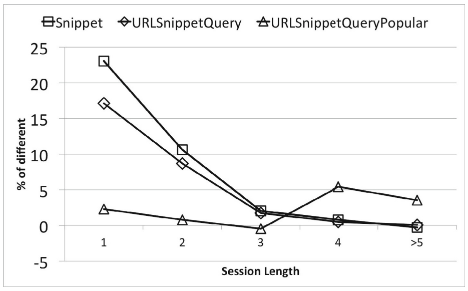
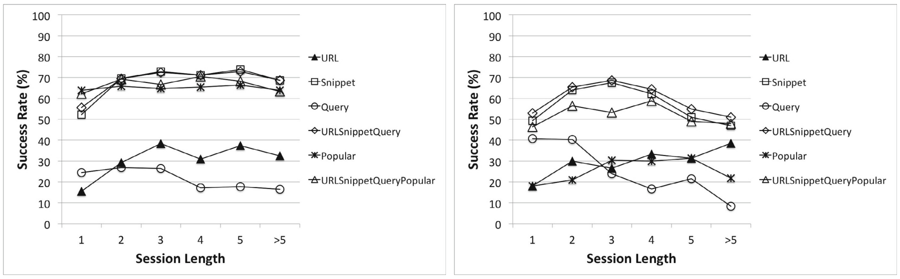


Fig. 7. Difference of success rate between standard session-based and term weighting



(a) Large staks with term weighting, k=1 (b) Small staks with term weighting, k=1

Fig. 8. Large vs. Small Search Staks

6.6 Success Rate of Stak Size

It is interesting to consider the influence of stak size on recommendation success rates. The majority of staks in this study (94%) contain a relatively few pages (1-500 URLs) which we expect to provide a relatively weak recommendation signal. As HeyStaks matures we can expect users to develop more mature staks and so it is appropriate to evaluate the relationship between recommendation success and stak size. To test this, we juxtapose the recommendation success rates by dividing the data according to the stak size into *small* (< 500 URLs) and *large* ($>= 500$ URLs), for $k = 1$ and with term weighting; see Figure 8.

Clearly there are differences in accuracy between small and large staks. Larger, more mature staks enjoy higher success rates across the various recommendation techniques and session length settings. For example, looking at *URLSnippetQueryPopularity*, we see a success rate of 63-70% for large staks (Figure 8(a)) compared to only 47-60% when used with small staks (Figure 8(b)). This is encouraging because, from a engineering standpoint, these higher success rates suggest it may be practical to implement a reliable automatic stak switching policy, for larger staks which contain more than 500 URLs.

7 Conclusions

In this paper we have reconsidered the stak recommendation challenge faced by HeyStaks, a case-based social search solution. Our main contribution has been to extend the work of [19] by exploring a number of novel stak recommendation strategies that take advantage of the additional information that is available as a source of context across extended search sessions. The results, based on live-user search logs, suggest that recommendation success can be improved by using extended search session data, albeit with certain caveats. For example, the benefit of using extended search sessions is maximized as when we consider the difference between singleton sessions and sessions of length 2. That being said, it is to the advantage of HeyStaks that these benefits are maximized for shorter sessions because these sessions are more frequent than longer sessions. Moreover, the relative improvements that we have found in stak recommendation accuracy, compared to the past work of [19], suggest that this new approach may have practical merit in a deployment setting, certainly for large stak sizes.

Acknowledgments. This work is supported by Science Foundation Ireland under grant 07/CE/I1147, HeyStaks Technologies Ltd, Ministry of Higher Education Malaysia and Universiti Teknikal Malaysia Melaka.

References

1. Spink, A., Tom Wilson, D.E., Ford, N.: Modeling Users' Successive Searches in Digital Environments. D-Lib Magazine (1998)
2. Amershi, S., Morris, M.R.: CoSearch: A System for Co-located Collaborative Web Search. In: Proceeding of the 26th Annual SIGCHI Conference on Human Factors in Computing Systems, CHI 2008, pp. 1647–1656. ACM, New York (2008)
3. Ashley, K.D.: Modeling Legal Arguments: Reasoning with Cases and Hypotheticals. MIT Press, Cambridge (1991)
4. Balfe, E., Smyth, B.: Case-Based Collaborative Web Search. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 489–503. Springer, Heidelberg (2004)
5. Bharat, K.: SearchPad: Explicit Capture of Search Context to Support Web Search. Computer Networks 33(1-6), 493–501 (2000)
6. Burke, R., Hammond, K., Kulyukin, V., Tomuro, S.: Question Answering from Frequently Asked Question Files. AI Magazine 18(2), 57–66 (1997)
7. Dou, Z., Song, R., Wen, J.R.: A Large-scale Evaluation and Analysis of Personalized Search Strategies. In: WWW 2007: Proceedings of the 16th International Conference on World Wide Web, pp. 581–590. ACM Press, New York (2007)
8. Godoy, D., Amandí, A.: PersonalSearcher: An Intelligent Agent for Searching Web Pages. In: Monard, M.C., Sichman, J.S. (eds.) SBIA 2000 and IBERAMIA 2000. LNCS (LNAI), vol. 1952, pp. 43–52. Springer, Heidelberg (2000)
9. Hatcher, E., Gospodnetic, O.: Lucene in Action. Manning Publications (2004)
10. He, D., Göker, A., Harper, D.: Combining Evidence for Automatic Web Session Identification. Information Processing & Management 38(5), 727–742 (2002)
11. Jansen, B.J., Spink, A., Blakely, C., Koshman, S.: Defining a Session on Web Search Engines. Journal of the American Society for Information Science and Technology 58(6), 862–871 (2007)

12. Kanawati, R., Jaczynski, M., Trousse, B., Andreoli, J.-M.: Applying the Broadway Recommendation Computation Approach for Implementing a Query Refinement Service in the CBKB Meta-search Engine. In: Conférence Française sur le Raisonnement á Partir de Cas, RáPC 1999 (1999)
13. Lenz, M., Ashley, K.: AAAI Workshop on Textual Case-Based Reasoning. AAAI Technical Report WS-98-12 (1999)
14. Liu, S.B.: Trends in Distributed Curatorial Technology to Manage Data Deluge in a Networked World. The European Journal for the Informatics Professional 11(4), 18–24 (2010)
15. Morris, M.R., Teevan, J.: Collaborative Search: Who, What, Where, When, Why, and How (Synthesis Lectures on Information Concepts, Retrieval, and Services). Morgan and Claypool Publishers (2010)
16. Plaza, E.: Semantics and Experience in the Future Web. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 44–58. Springer, Heidelberg (2008)
17. Qiu, F., Cho, J.: Automatic Identification of User Interest for Personalized Search. In: WWW 2006: Proceedings of the 15th International Conference on the World Wide Web, pp. 727–736. ACM Press, New York (2006)
18. Rissland, E.L., Daniels, J.J.: A Hybrid CBR-IR Approach to Legal Information Retrieval. In: Proceedings of the 5th International Conference on Artificial Intelligence and Law, pp. 52–61. ACM Press (1995)
19. Saaya, Z., Smyth, B., Coyle, M., Briggs, P.: Recommending Case Bases: Applications in Social Web Search. In: Ram, A., Wiratunga, N. (eds.) ICCBR 2011. LNCS, vol. 6880, pp. 274–288. Springer, Heidelberg (2011)
20. Smyth, B., Balfe, E., Freyne, J., Briggs, P., Coyle, M., Boydell, O.: Exploiting Query Repetition and Regularity in an Adaptive Community-Based Web Search Engine. User Model. User-Adapt. Interact. 14(5), 383–423 (2004)
21. Smyth, B., Briggs, P., Coyle, M., O'Mahony, M.: A Case-Based Perspective on Social Web Search. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 494–508. Springer, Heidelberg (2009)
22. Smyth, B., Champin, P.A.: The Experience Web: A Case-based Reasoning Perspective. In: Grand Challenges for Reasoning from Experiences, Workshop at IJCAI 2009, pp. 566–573 (2009)
23. Weber, R.O., Ashley, K.D., Bruninghaus, S.: Textual Case-based Reasoning. Knowledge Engineering Review 20(3), 255–260 (2005)

Event Extraction for Reasoning with Text

Sadiq Sani, Nirmalie Wiratunga, Stewart Massie, and Robert Lothian

School of Computing,
Robert Gordon University,
Aberdeen AB25 1HG, Scotland, UK
{s.a.sani,n.wiratunga,s.massie,r.m.lothian}@rgu.ac.uk

Abstract. Textual Case-Based Reasoning (TCBR) aims at effective reuse of past problem-solving experiences that are predominantly captured in unstructured form. The absence of structure and a well-defined feature space makes comparison of these experiential cases difficult. Since reasoning is primarily dependent on retrieval of similar cases, the acquisition of a suitable indexing vocabulary is crucial for case representation. The challenge is to ensure that this vocabulary is selective and is representative enough to be able to capture the intended meaning in text, beyond simply the surface meaning. Indexing strategies that rely on bag of words (BOW) have the advantage of low knowledge acquisition costs, but only facilitate case comparison at a superficial level. In this paper we study the influence of semantic and lexical indexing constructs on a retrieve-only TCBR system applied to incident reporting. We introduce, RUBEE (RUle-Based Event Extraction), an unsupervised approach for automatically extracting events from incident reports. A novel aspect of RUBEE is its use of polarity information to distinguish between events that occurred and any non-event occurrences. Our results show that whilst semantic indexing is important, there is evidence that case representation benefits from a combined vocabulary (both semantic and lexical). A comparative study involving a popular event extraction system, EVITA, and several baseline algorithms also indicate that events extracted by RUBEE lead to significantly better retrieval performance.

1 Introduction

The case-based reasoning (CBR) approach to problem solving assumes that similar problems have similar solutions. Accordingly relevant experiences of problem-solving are maintained as cases for future reuse. A case in CBR comprises a problem description usually represented as a set of feature values to enable case comparison, and similarity-based ranking. Thus solving a new problem involves reusing the solution of cases whose problem descriptions are most similar to that of the new problem. Textual CBR (TCBR) applies the CBR methodology to situations that are predominantly captured in text form (typically documents) e.g., reports, technical manuals and Frequently Asked Questions files [22]. Reasoning with textual content requires that a suitable indexing vocabulary is in place to represent cases and that a similarity metric is defined for case comparison. The challenge here is to automate or semi-automate the process of acquiring knowledge to facilitate representation and comparison.

An important application area for TCBR is incident reporting, where the retrieval of similar past incidents can help safety personnel to identify factors and patterns that contribute to accidents and also assist with the authoring of new reports [15,24]. Incident reports contain valuable experience in the form of descriptions of incidents that have occurred, circumstances that led to the incident, and any injuries and damage that may have resulted from the incident. These descriptions are typically centered around events. For example a fire related incident report is likely to contain snippets such as:

"Gas was **leaking** from the pipe"
"This resulted in a **fire**"
"The operator was severely **burned**"

In a retrieve-only system, given a new (query) incident report, we want to compare and retrieve past incident reports on the basis of incident cause, the type of incident or the type of injury, for example. To address this requirement it is necessary to ensure that the indexing vocabulary encompasses both lexical and semantic features - lexical to capture general context and semantic to capture relevant events and relationships.

In this paper we present an unsupervised heuristic approach to event extraction from incident reports called RUBEE (RUle-Based Event Extraction). We further present a model for using events and event polarity (whether the occurrence of the event is negated or affirmed) for text representation with a view to improving a retrieve-only CBR system, where a case represents a single incident report. Specifically, we study the effectiveness of an event-based representation in differentiating between cases that have very similar context (i.e. describe similar situations) but contain different events. For this purpose we present results from an experiment designed to study the categorisation of reports, on the basis of, whether or not injuries were sustained in similar incident scenarios. A comparative study is used to analyse classification performance of the retrieve-only TCBR system with events extracted by RUBEE versus those extracted by the popular EVITA system [19].

This paper is structured as follows: Section 2 presents background information on event extraction. Section 3 presents RUBEE, our event extraction algorithm, while the EVITA system is described in Section 4. Our proposed representation framework for using both semantic and lexical information is presented in Section 5. Evaluations are presented in Section 6, related work in Section 7 and conclusions in Section 8.

2 Event Extraction

The term 'event' has been used to mean different things in different projects. Early work in Information Extraction (IE) within the Message Understanding Conference (MUC) describes its scenario extraction task as "extract pre-specified event information and relate the event information to a particular organisation, person or artifact [14]". More recent work in information retrieval (IR) within the Topic Detection and Tracking (TDT) task define events to be "some unique thing that happens at some point in time" [1]. The Automatic Content Extraction (ACE) task describe events as "a specific occurrence involving participants, something that happens or a change of state"[4]. Despite the lack of consensus on the definition of events or the types of expressions that constitute them, there is however a division of events into atomic and complex types [10].

Atomic events are expressed in text using single words (e.g., "fall" and "break"), or multi-word expressions (e.g "take off") [10,4,18]. Such event expressions are typically verbs, but not exclusively so. This is in contrast with the view of events popularised by MUC in its scenario extraction task where events are represented by complex template structures involving participants called *roles*. Another complex view of events is presented in Roger Schank's work on Scripts where events (also called *scenes*) are described as sequences of primitive *actions* e.g., the event of ordering food at a restaurant consists of the actions of **receiving** a menu, **reading** the menu, **deciding** on the meal and giving the **order** to the waitress [20].

Effective case representation increases with exhaustivity of the indexing vocabulary [15]. Essentially this is the ability of the indexing vocabulary to represent inherent concepts and notions in text leading to better recall at retrieval time. Because atomic events are represented at the word or predicate level, this makes them better suited for exhaustively capturing all events in text compared to higher level complex events which have a more selective structure. In other words, atomic events are better suited as an indexing vocabulary for case representation. In addition, whilst complex events are highly domain dependent, atomic events are not. For example, the bombing and kidnap MUC templates are defined for a specific domain i.e. terrorism. Similarly, TDT event extraction is focused on specific real-world events e.g., "NYC subway bombing" and not the generic type of event information needed for general text representation. Thus, atomic events are applicable across multiple, different domains. In the rest of this paper any mention of event refers to atomic events.

3 RUBEE- RUle-Based Event Extraction

RUBEE is an unsupervised rule-based event extraction algorithm which exploits knowledge from linguistic analysis and a lexical database. A source document is read, tokenized, tagged with part-of-speech information and sentences are parsed into syntactic and dependency structures using the Stanford Parser [7]. This allows us to identify the grammatical roles of tokens in the sentence e.g., whether a verb is a main verb or an auxiliary. This information is used by RUBEE to decide whether candidate tokens should be accepted or rejected as valid events. The event extraction process is shown in Figure 1. Here, WordNet [16] is used to provide background knowledge for identifying event candidates. For example hypernymy information is used to identify candidate nouns for event extraction. We note that a glossary or ontology of events could also be utilised here instead of Wordnet.

RUBEE's event extraction algorithm appears in Figure 2. Given a sentence $\mathcal{S}$ a regular expression is matched in order to identify candidate token sequences based on part-of-speech information. Candidate events are then filtered using a sequence of conditional statements to identify the final set of valid events. Finally, the polarity (negative or

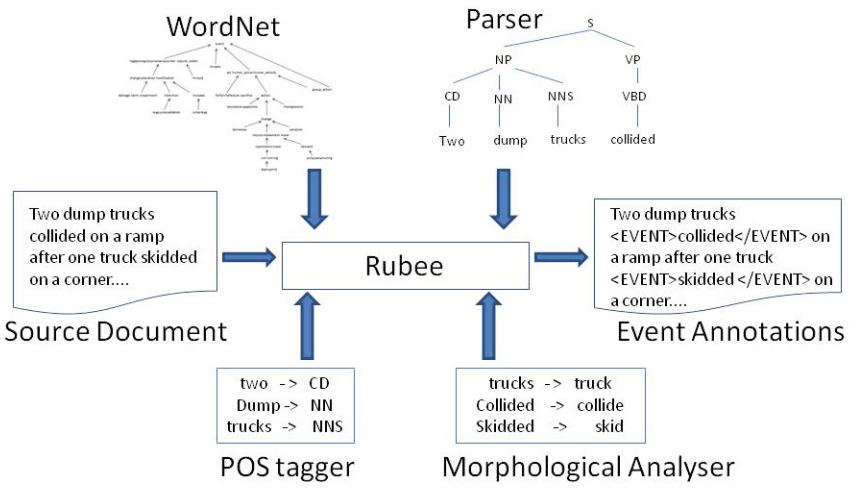


Fig. 1. Event extraction process

positive) of each event is identified. We consider event candidates from three parts-of-speech categories: verbs, nouns and adjectives. Corresponding extraction heuristics for each of these part-of-speech categories are explained with examples below followed by a discussion of how polarity information is identified during event extraction.

3.1 Verbs

Verbs typically express actions or happenings and as such are good candidates for events. However, we use the following rules to filter out unlikely verb candidates:

– Auxiliaries: Auxiliary verbs are non-main verbs in a clause and typically serve to only support the main verb. For example:

  "Closing the lid would have **prevented** the hot material from falling".

  In the preceding example (and all subsequent examples) the event is shown in bold and the non-event verbs are underlined. The verbs "would" and "have" are auxiliary verbs that modify the main event verb. Thus, only "prevented" is extracted as an event.
– Modifiers: Verbs often appear as modifiers of nouns and noun phrases e.g., "drilling team" and "cutting equipment" Such verbs are not extracted as events.
– Verb+Particle and Verb+Preposition: These types of constructs have a different meaning from their verb component e.g.,

  "The regulator was **turned off** and the fire self extinguished"
  "The fire was **put out** with a hand held extinguisher"

  Such constructs are identified and extracted as events. We validate all extracted verb+particle and verb+prep sequences by looking them up in WordNet. Thereafter, for any sequence of words not known to WordNet (e.g., spray over) we extract only the main verb (spray).

```
Let:
    $\mathcal{S} = \{t_1, ..., t_n\}$, a sentence which is a sequence of tokens $t_i$
    $s \subseteq \mathcal{S}$, a subsequence of tokens in $\mathcal{S}$
    $p_i$, a part-of-speech (pos) tag for token $t_i$
    $p = \{p_1, ..., p_n\}$, the sequence of all pos tags $p_i$ of tokens $t_i$ in $s$
    $pos(s) \rightarrow p$, a function:
    $\simeq$, a regular expression matching operator
    $\mathcal{C}$, the set of all candidate event tokens
    $\mathcal{E}$, the set of all selected events
    $\mathcal{V}$, the set of all verbs in WordNet
    $\mathcal{N}$, the set of all identified WordNet noun event Synsets
For each $s \in \mathcal{S}$
    If $pos(s) \simeq$ VB.*RB|VB.*IN|VB.*RP
        If s $\in \mathcal{V}$
            $\mathcal{C} = \mathcal{C} + s$
        Else
            $\mathcal{C} = \mathcal{C} + mainverb(s)$
    Else if $pos(s) \simeq$ VB.*
        $\mathcal{C} = \mathcal{C} + s$
    Else if $pos(s) \simeq$ NN.*
        If $hypernym(s) \in \mathcal{N}$
            $\mathcal{C} = \mathcal{C} + s$
    Else if $pos(s) \simeq$ JJ.*
        If $verbDerived(s)$
            $\mathcal{C} = \mathcal{C} + s$
For each $c \in \mathcal{C}$
    If not $auxilliary(c) \wedge$ not $NN\_modifier(c)$
        $\mathcal{E} = \mathcal{E} + c$
For each $e \in \mathcal{E}$
    $extractPolarity(e)$
```

Fig. 2. RUBEE Algorithm

3.2 Nouns

Unlike verbs, most nouns are not events. Thus identification of noun events requires a more selective process. We identified a small set of WordNet synsets called event parents and maintained their hyponyms (children nodes) as relevant event expressions. These synsets were identified by manually extracting noun events from a set of training documents, mapping each one to a corresponding WordNet synset and then identifying a suitable hypernym from the root. A hypernym is suitable if it is considered to denote a type of occurrence or event. For example the noun events "extraction", "combustion" and "absorption" are manually extracted from the training documents and the synset which subsumes these events is identified in WordNet. In this case this is the synset "Physical Process" which is defined as "a sustained phenomenon or one marked by gradual changes through a series of states". The parent of "Physical Process" is the synset "Physical Entity" which does not fit the description of a type of occurrence or event. Thus we created a rule which accepts nouns that are hyponyms of "Physical

Process" as candidate events. The final set of WordNet parent nodes used for selecting nouns are:

– Event: The first sense of "event" in WordNet is defined as "something that happens at a given time and place". Hyponyms of this synset make up the largest class of event words e.g., **collision**, **movement** and **fire**.
– Physical Process: This synset is defined as "a sustained phenomenon or one marked by gradual changes through a series of states" and it includes the hyponyms **ignition**, **combustion** and **overheating**.
– Ill Health: This is defined by WordNet as "a state in which you are unable to function normally and without pain". Hyponyms of this synset include the events: **fracture**, **contusion** and **laceration**.
– Symptom (medicine): This has the definition: "Any sensation or change in bodily function that is experienced by a patient". Relevant hyponyms include: **soreness** and **pain**.
– Injury: We ignore the first sense of "injury" because it is already a hyponym of the synset "Ill Health". The second sense of "injury" has the definition"An accident that results in physical damage". Hyponyms of this synset include the event **concussion**.

3.3 Adjectives

The last class of events types are adjectives, which often occur as participles e.g.,

"A fitter suffered a **lacerated** forehead"
"A light vehicle driver received a **bruised** shoulder"

These event types are extracted with the help of WordNet which is used to identify adjectival expressions that are derived from verbs. WordNet maintains a "participle of" relation between adjectives and their corresponding root verbs. For example the adjective "elapsed" has a "participle of" relation with the verb "elapse". However, this strategy was found to have very limited coverage. Instead an alternative strategy was used whereby morphological analysis is used to derive the verb from the adjective before validating with Wordnet. Since participles typically have the same spelling as past-tense verbs, a lemmatiser is used to transform the adjective into a root verb. For example the adjective "fractured" is lemmatised to "fracture". The lemma is then looked-up in WordNet. If the lemma is a valid verb, the adjective is accepted as a valid event.

3.4 Event Polarity

The polarity of an event is negative if the occurrence of the event is explicitly negated in the text and positive otherwise. Negative polarity is often expressed using a negative word e.g.,"not" and "no". Event polarity is particularly important for retrieval because it helps to distinguish between affirmed and negated occurrences of the same event. This helps to avoid false matching of events that have opposite polarity. Take for example the following sentences:

"An operator suffered crush **injuries**"
"No contact with the electricity was made and **no injuries** were sustained"

Without identifying the polarity of injury, the two sentences can incorrectly be considered similar even though the second example clearly negates the occurrence of injuries.

```
Let:
    $N = \{n_1, ..., n_m\}$, a set of negation words
    $\mathcal{E} = \{e_1, ..., e_m\}$, a set of events
For each $e \in \mathcal{E}$
    If $hasNegDeterminer(e) \vee hasNegModifier(e)$
        $\vee\ isObjectOf(e, n \in N)$
        $e = \neg e$
```

Fig. 3. Polarity Extraction Algorithm

Event polarity is extracted using dependency parse information to check for negative modifiers and negations as shown in Figure 3. All events that have a negative determiner ("no"), a negation modifier ("not") or are objects of a word that indicates negation (e.g "avoid") are considered to have negative polarity. Consequently all events are stored together with their corresponding polarity value which is later utilised in our case representation and comparison strategy.

4 EVITA

EVITA is a system for identifying and extracting events from text using a combination of linguistic analysis, heuristic rules and lexical lookup [19]. One of the key differences between EVITA and RUBEE involves the manner in which sentences are processed. While RUBEE uses full dependency parsing, EVITA uses chunking; a form of shallow parsing that produces linguistically defined groups of adjacent words e.g., noun phrases and verb phrases, rather than full parse trees. Although chunking is a less expensive operation compared to parsing, parse trees provide richer syntactic information of sentences and so are more useful for deep linguistic analysis. Consequently, EVITA's rules are based on pattern matching on word sequences while RUBEE's rules are based on dependency-tree structures.

The event extraction process employed by EVITA is shown in Figure 4. Heads of verbal chunks are extracted as events by EVITA, provided that they do not belong to a set of non-event verbs obtained from lexical inventories. Unlike RUBEE, EVITA does not recognise verb+particle and verb+prep event types. When such constructs are encountered, EVITA extracts only the head verb. Noun events are extracted based on hypernymy information from WordNet, and is likely to be similar to RUBEE. However, details of the synsets used are not given. Extraction of adjectival events is based on lookup whereby candidate adjectival events are accepted if they occur in the list of annotated events in the TimeBank-1.2 Corpus which contrasts with the use of morphological analysis by RUBEE.

EVITA extracts a number of event attributes including polarity using pattern matching techniques. Again, details of these pattern matching techniques are not given.

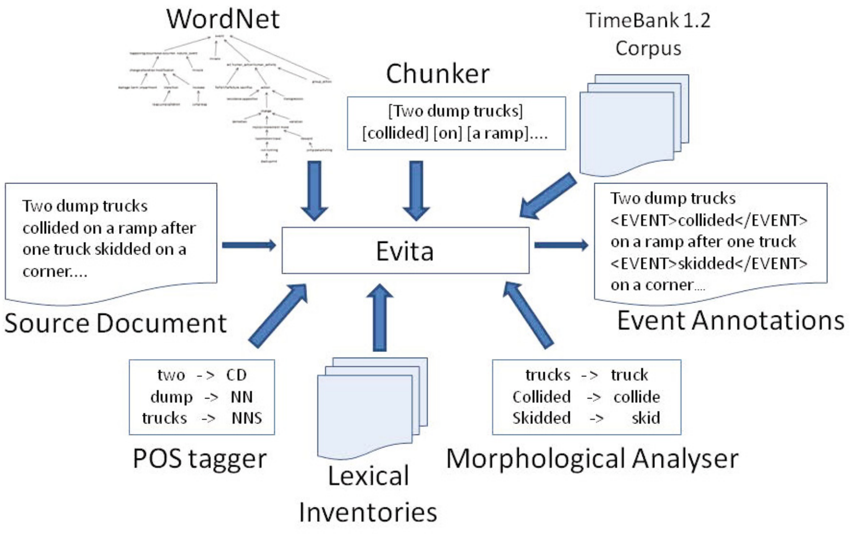

WordNet
TimeBank 1.2 Corpus
Chunker
[Two dump trucks] [collided] [on] [a ramp]....
Two dump trucks collided on a ramp after one truck skidded on a corner....
Source Document
Evita
Two dump trucks <EVENT>collided</EVENT> on a ramp after one truck <EVENT>skidded</EVENT> on a corner....
Event Annotations
two -> CD
dump -> NN
trucks -> NNS
POS tagger
Lexical Inventories
trucks -> truck
Collided -> collide
Skidded -> skid
Morphological Analyser

Fig. 4. Event extraction process

5 Case Representation

Once events have been extracted, they need to be utilised for case representation. In this section we present a model for utilising extracted events for textual case representation where cases are represented using both lexical and semantic information. Lexical information is represented using a standard Bag-of-Words (BOW) indexing vocabulary where text is represented in a vector space whose dimensions correspond to individual terms. Similarly, semantic information is represented using a Bag-of-Events (BOE) vector representation where dimensions correspond to the event vocabulary and separate dimensions are used to represent negative and positive polarity instances of the same event. Thus a case is represented as a pair:

$$c = (\boldsymbol{t}, \boldsymbol{e}) \tag{1}$$

Where $\boldsymbol{t}$ is the BOW representation and $\boldsymbol{e}$ is a BOE representation for the case c. Here any standard text representation scheme such as binary vectors or tf-idf vectors can be used for the entries of both $\boldsymbol{t}$ and $\boldsymbol{e}$. Note that while $\boldsymbol{e}$ captures event information, $\boldsymbol{t}$ includes important contextual information that may not be captured by $\boldsymbol{e}$. Document similarity is thus computed as shown in Equation 2.

$$SIM(c_q, c_i) = (1 - \alpha)Sim(\boldsymbol{t}_q, \boldsymbol{t}_i) + \alpha Sim(\boldsymbol{e}_q, \boldsymbol{e}_i) \tag{2}$$

Where $SIM(c_q, c_i)$ is the global similarity between a query case c_q and any case c_i from the casebase, $Sim(\boldsymbol{t}_q, \boldsymbol{t}_i)$ is the BOW similarity between c_q and c_i, $Sim(\boldsymbol{e}_q, \boldsymbol{e}_i)$ is the BOE similarity between c_q and c_i, and α is a mixing parameter. Thus the similarity between two cases is an aggregation of their terms and events similarities, whilst α controls the contribution of each representation's similarity to overall global similarity. Note that increasing the value of α increases the contribution of the BOE representation. Both $Sim(\boldsymbol{t}_q, \boldsymbol{t}_i)$ and $Sim(\boldsymbol{e}_q, \boldsymbol{e}_i)$ are obtained using the cosine similarity measure given in equation 3.

$$Sim(a,b) = Cos(a,b) = \frac{\sum_{i=0}^{n} a_i \times b_i}{\sqrt{\sum_{i=0}^{n} a_i^2} \times \sqrt{\sum_{i=0}^{n} b_i^2}} \qquad (3)$$

6 Evaluation

The aim of our experiments is to establish the utility of event-based semantic indexing for retrieval of incident reports. Our comparative study is applied to the following representation schemes:

1. BOW: a BOW-only representation where $\alpha = 0$
2. BOE: a BOE-only representation where $\alpha = 1$
3. $Comb$: a combined representation where $0 < \alpha < 1$

The BOE representation can be generated using three extraction strategies:

1. VERBS: a baseline approach that extracts only verbs as events according to part-of-speech information without further linguistic analysis
2. RUBEE (as in Section 3)
3. EVITA (as in Section 4)

Accordingly BOE representations obtained using the different event extraction approaches are called $\boldsymbol{BOE_{VERBS}}$, $\boldsymbol{BOE_{RUBEE}}$ $\boldsymbol{BOE_{EVITA}}$ respectively. The representation obtained by combining events from RUBEE and BOW as described in Section 5 is called $\boldsymbol{Comb_{RUBEE}}$. Currently we determine the alpha value that results in best performance empirically. Case retrieval performance using the above representations is measured based on classification accuracy with 3 nearest neighbours (3-NN) on stratified 10-fold cross-validation experiments. Significance is reported from a Wilcoxon signed-rank test with 95% confidence.

6.1 Datasets

Several benchmark datasets were created using incident reports crawled from the Government of Western Australia's Department of Mines and Petroleum website [1]. These incident reports are pre-classified into "Injury" and "NoInjury" classes. Accordingly we treat this as a classification task. Details of these datasets are given in Table 1. We also combine the TRUCKR and TRUCKC datasets to form a new dataset called ROLLCOL.

[1] http://dmp.wa.gov.au

This new dataset is used to further test if event information can help distinguish between collision and rollover incidents involving trucks. Each dataset in Table 1 contains 200 documents; 100 documents in each class. All datasets have a similar vocabulary size (with MISCI having the largest vocabulary) from which the indexing vocabulary is drawn for each algorithm.

Table 1. Datasets

Name	*Domain*	*Description*	*Voc. Size*
TRUCKC	TruckCollision	Incidents involving truck collision	1182
Fire	Fire	Incidents involving fire outbreak	1326
TRUCKR	TruckRollover	Incidents involving truck rollover	1031
LIGHTV	LightVehicle	Incidents involving light vehicle accidents	1064
MISCI	MiscIncidents	Miscellaneous incidents	1581
ROLLCOL	RolloverCollision	A combination of TRUCKR and TRUCKC incidents	1212

6.2 Results

From table 2, we observe that event-only representation with BOE_{RUBEE} was significantly better than BOW on 4 of the datasets. Performance of BOE_{RUBEE} on the RollCol dataset is not significantly better than BOW while BOW is significantly better than both BOE_{RUBEE} and BOE_{EVITA} on the MiscI dataset. The reason for this might be explained by the variety of different types of incidents and injuries in this dataset introducing a degree of sparseness into the BOE representation. BOE_{RUBEE} significantly outperforms BOE_{EVITA} on all datasets except the RollCol dataset where BOE_{EVITA} performs slightly (but not significantly) better. BOE_{VERBS}'s performance was generally poor compared to all other approaches including BOW. This shows that the linguistic analysis used by the event extraction algorithms is important for identifying event information for document indexing.

Table 2. Classification accuracies of different representation schemes. Best results on each dataset are presented in bold.

	TruckC	*Fire*	*TruckR*	*LightV*	*MiscI*	*RollCol*
BOW	80.5	84.7	78.4	81.0	84.7	83.4
BOE_{VERBS}	78.5	83.4	76.7	75.3	75.6	81.1
BOE_{EVITA}	80.8	82.7	74.4	81.3	78.6	87.1
BOE_{RUBEE}	84.5	**90.0**	85.4	85.1	81.0	85.2
$Comb_{RUBEE}$	**87.5**	**90.0**	**86.4**	**88.1**	**88.6**	**91.1**

For the combined representation ($Comb_{RUBEE}$), we observed improvements over all 4 individual indexing schemes on all 6 datasets. Specifically, $Comb_{RUBEE}$ performed significantly better than BOW, BOE_{VERBS} and BOE_{EVITA} on all datasets. Comparing with BOE_{RUBEE}, $Comb_{RUBEE}$ performed significantly better on all datasets with the exception of FIRE and TRUCKR. This confirms our hypothesis that the lexical information in the BOW representation and the semantic information in the BOE representation are complementary. Thus, a combination of both leads to even better retrieval performance.

In Table 3 we present results for BOE_{RUBEE} with and without polarity information. Improvements are realised with polarity information on all datasets except FIRE and MISCI. Improvements on the TRUCKC and TRUCKR datasets are statistically significant. Table 4 provides statistics of negations found in each dataset. Observe that in the FIRE datasets, a total of 8 events were found with negative polarity. However, none of these were negations of injury events and thus, no benefit was realised on classification accuracy. In contrast, 25 negations were extracted from the TRUCKR dataset, 14 of which were negations of injuries. This leads to significantly better classification accuracy on the TRUCKR dataset.

Table 3. Classification accuracy of RUBEE with and without event polarity

	TruckC	*Fire*	*TruckR*	*LightV*	*MiscI*	*RollCol*
BOE_{RUBEE}	**84.5**	**90.0**	**85.4**	**85.1**	81.0	**85.2**
$BOE_{RUBEE}(NoPol)$	82.7	89.9	81.7	84.2	**81.6**	84.8

For the ROLLCOL dataset in Table 4, a total of 46 negations were found, 15 of which are negations of injuries. However, recall that the task on this particular dataset is to distinguish between "Collision" and "Rollover" incidents. Thus, negations of injuries are found in both classes and are not useful for distinguishing between different classes. Also, unlike "Injury" and "NoInjury" classes, "Collision" and "Rollover" incidents are not polar opposites. Consequently out of all negations found, none were negations of "Collision" or "Rollover" events. This further suggests that polarity information is particularly useful for distinguishing between classes that are polar opposites.

Table 4. Statistics of negations extracted from all datasets

	TruckC	*Fire*	*TruckR*	*LightV*	*MiscI*	*RollCol*
Total event negations	42	8	25	27	20	46
Negations of injury events	9	0	14	9	3	15

Figure 5 shows average accuracy for increasing values of α over all runs of the RUBEE algorithm. Best results are generally obtained within the range $0.4 \leq \alpha \geq 0.7$. This indicates the BOE representation is largely responsible for the improved performance of the Combined approach. The difference between the highest and lowest accuracy obtained between $\alpha = 0.1$, and $\alpha = 0.9$ (i.e excluding BOW-only and BOE-only representations) is from 3.8% for Fire to 6.4% for the TruckR. However, note that (with the exception of the MiscI and RollCol datasets) the variation in accuracy levels-off with higher values of alpha ($\alpha >= 0.5$).

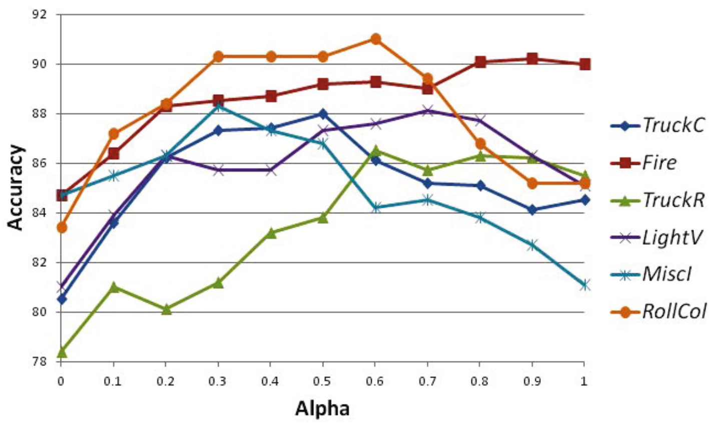


Fig. 5. RUBEE's performance as a function of α on each dataset

These results demonstrate the utility of event extraction for representing textual cases in domains characterised by eventualities. The results also confirm our proposed document representation model effectively combines contextual information from terms with semantic information from events. Lastly, the comparison between RUBEE and EVITA on these tasks points in favour of RUBEE as an effective event extraction system.

7 Related Work

The limitations of the standard BOW approach for textual case representation is well understood in TCBR and several approaches have been utilised to introduce semantic knowledge in order to improve case comparison and retrieval. Statistical approaches have been used for estimating semantic relatedness between terms in a corpus. Latent Semantic Indexing (LSI) [8] is a popular example that exploits co-occurrence patterns of terms and documents to create a semantic concept space where documents that are related are brought closer together. In this way, LSI brings out the underlying latent semantic structure in texts. In contrast our approach utilises natural language processing techniques to extract semantically rich case indexing vocabulary.

Approaches that employ shallow semantic analysis to improve retrieval are presented in the FAQ Finder [3] and FallQ [13] Question Answering systems. FAQ Finder uses WordNet to identify similarity between semantically related terms thus overcoming the problem of variation in vocabulary. While FAQ Finder uses keywords for case representation, our approach uses events as an indexing vocabulary. On the other hand, the indexing vocabulary of FallQ is based on Information Entities obtained from a domain lexicon. In contrast, RUBEE uses linguistic analysis together with a domain independent lexicon (WordNet) making it domain independent and more portable.

Other approaches propose using information extraction to map natural language text onto knowledge containers such as templates, frames or Scripts. For example, an approach that uses knowledge structures called Factors for representing legal cases is presented in [2]. A Factor-based representation allows for comparing cases using a limited set of abstractions that capture the semantics of legal arguments in the cases rather than the exact lexical expression of the arguments. A similar system that maps legal texts into a structured case representation with a fixed number of attributes is presented in [21]. On the other hand, the Fast Reading Understanding and Memory Program (FRUMP) [9] uses Scripts to achieve deep natural language understanding of text. A major disadvantage of all these systems however, is the manual effort required to create the set the knowledge structures (Scripts, Factors, Frames etc). Also, such systems rely on considerable amounts of domain knowledge which includes knowledge of the text genre, relevant conventions, as well as implicitly assumed background and world knowledge [17]. Thus, these systems are highly domain specific and require considerable knowledge engineering effort in order to port them to new domains. Furthermore, while systems like FRUMP have been applied to the tasks of Summarisation and Question Answering, it remains to be shown that deep understanding of text as proposed by these systems is necessary or useful for simple retrieval.

Event-based Text representation using Semantic Role Labeling (SRL) has been popularly used in Question Answering [23,6,12]. These approaches use event argument structures to identify the most likely answer to a question from candidate answer passages. For example annotating the question "who assassinated President Kennedy" with semantic roles, the expected answer role can be identified as the agent of the event "assassinated". Thus given candidate answer sentences that contain the phrase "President Kennedy" in the patient role of the event "assassinate", the noun or noun-phrase that is the agent of that same event would be extracted as the correct answer. Because SRL provides argument structures for sentences, it is not clear how output from SRL would scale to document comparison. In contrast our approach uses extracted events to represent text in a vector space model. Our extraction strategy closely follows the ACE [4] specification whilst further utilising relationship information from a lexical database. Also, key to our approach is the representation of events together with their polarity attribute which to the best of our knowledge has not been applied to text retrieval.

The identification of polarity is essential in Sentiment Analysis research [5]. However, event polarity (whether the occurrence of an event is explicitly negated or affirmed) differs significantly from sentiment polarity which aims to identify whether the opinion expressed about a topic is positive or negative. Thus lexicons used in sentiment analysis e.g., SentiWordnet are not suitable for extracting event polarity. For example the statement "a terrible collision happened" has negative sentiment due to the word "terrible" having a SentiWordNet score of -0.875, but has positive event polarity because the collision event did occur. Also, the word "prevent" in the example"she prevented the collision" produces negative event polarity because the collision was averted, but has positive sentiment score in SentiWordNet ($+0.25$). Lists of negation words are typically used for identifying negation [5,11]. Our approach uses dependency parsing to check for negation modifiers (rule 1) and negative determiners (rule 2) in addition to a list of negation words (rule 3).

8 Conclusions

In this paper we have demonstrated the utility of event information for representing textual cases in a retrieve-only CBR system. The contributions of this paper include a domain independent, rule-based approach for the extraction of atomic events called RUBEE. We also present a general framework for the representation of text using both lexical and event information. Furthermore we demonstrate how event polarity can be utilised within text representation to distinguish between asserted and negated occurrences of the same event. Finally, we presented a comparative analysis with the popular EVITA event extraction system and the results show better case classification accuracy with RUBEE over EVITA.

Future work will involve identifying the relative importance of different types of events and also learning relationships such as similarity and causality between events. Furthermore, we plan to extend our evaluation to other domains in order to verify the domain independence of RUBEE.

References

1. Allan, J., Carbonell, J., Doddington, G., Yamron, J., Yang, Y.: Topic detection and tracking pilot study final report. In: Proceedings of the Broadcast News Transcription and Understanding Workshop (1998)
2. Brüninghaus, S., Ashley, K.D.: The Role of Information Extraction for Textual CBR. In: Aha, D.W., Watson, I. (eds.) ICCBR 2001. LNCS (LNAI), vol. 2080, pp. 74–89. Springer, Heidelberg (2001)
3. Burke, R.D., Hammond, K.J., Kulyukin, V.A., Lytinen, S.L., Tomuro, N., Schoenberg, S.: Question answering from frequently asked question files: Experiences with the FAQ Finder system. Tech. rep. University of Chicago (1997)
4. Consortium, L.D.: ACE (Automatic Content Extraction) English Annotation Guidelines for Events (2005)
5. Councill, I.G., McDonald, R., Velikovich, L.: What's great and what's not: learning to classify the scope of negation for improved sentiment analysis. In: Proceedings of the Workshop on Negation and Speculation in Natural Language Processing. NeSp-NLP 2010, pp. 51–59. Association for Computational Linguistics (2010)
6. Dan, S., Mirella, L.: Using semantic role to improve question answering. In: Proceedings of EMNLP 2007 (2007)
7. De Marneffe, M., Maccartney, B., Manning, C.D.: Generating typed dependency parses from phrase structure parses. In: Proceedings of International Confrerence on Language Resources and Evaluation (2006)
8. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by latent semantic analysis. Journal of the American Society of Information Science 41, 391–407 (1990)
9. Dejong, G.: An overview of the FRUMP system. In: Strategies for Natural Language Processing. Lawrence Erlbaum, Hillsdale (1982)
10. Filatova, E., Hatzivassiloglou, V.: Domain-independent detection, extraction, and labeling of atomic events. In: Proceedings of the Fourth International Conference on Recent Advances in Natural Language Processing, RANLP 2003 (2003)
11. Hogenboom, A., van Iterson, P., Heerschop, B., Frasincar, F., Kaymak, U.: Determining negation scope and strength in sentiment analysis. In: International Conference on Systems, Man and Cybernetics, pp. 2589–2594. IEEE (2011)

12. Kaisser, M., Webber, B.: Question answering based on semantic roles. In: Proceedings of the Workshop on Deep Linguistic Processing, Deep LP 2007. Association for Computational Linguistics (2007)
13. Lenz, M., Burkhard, H.D.: CBR for Document Retrieval: The FALLQ Project. In: Leake, D.B., Plaza, E. (eds.) ICCBR 1997. LNCS, vol. 1266, pp. 84–93. Springer, Heidelberg (1997)
14. Marsh, E., Perzanowski, D.: Muc-7 evaluation of ie technology: Overview of results. In: Proceedings of the Seventh Message Understanding Conference (MUC-7) (1998)
15. Massie, S., Wiratunga, N., Craw, S., Donati, A., Vicari, E.: From Anomaly Reports to Cases. In: Weber, R.O., Richter, M.M. (eds.) ICCBR 2007. LNCS (LNAI), vol. 4626, pp. 359–373. Springer, Heidelberg (2007)
16. Miller, G.A.: Wordnet: A lexical database for english. Communications of the ACM 38, 39–41 (1995)
17. Moens, M.F.: Information Extraction: Algorithms and Prospects in a Retrieval Context (The Information Retrieval Series). Springer, New York, Inc. (2006)
18. Sauri, R., Goldberg, L., Verhagen, M., Pustejovsky, J.: Annotating Events in English TimeML Annotation Guidelines (2009)
19. Saurí, R., Knippen, R., Verhagen, M., Pustejovsky, J.: Evita: a robust event recognizer for QA systems. In: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT 2005, pp. 700–707. Association for Computational Linguistics (2005)
20. Schank, R.C., Abelson, R.P.: Scripts, plans, and knowledge. In: Proceedings of the 4th International Joint Conference on Artificial Intelligence, vol. 1, pp. 151–157. Morgan Kaufmann Publishers Inc. (1975)
21. Weber, R., Martins, A., Barcia, R.M.: On legal texts and cases. In: Textual Case-Based Reasoning: Papers from the AAAI 1998 Workshop (Technical Report WS-98-12), pp. 40–50. AAAI Press, Menlo Park (1998)
22. Weber, R.O., Ashley, K.D., Brüninghaus, S.: Textual case-based reasoning. Knowledge Engineering Review 20, 255–260 (2005)
23. Wiegand, M., Leidner, J.L., Klakow, D.: Combining term-based and event-based matching for question answering. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Information Retrieval, pp. 715–716. ACM (2007)
24. Wilson, D.C., Carthy, J., Abbey, K., Sheppard, J., Wang, R., Dunnion, J., Drummond, A.: Textual CBR for Incident Report Retrieval. In: Kumar, V., Gavrilova, M.L., Tan, C.J.K., L'Ecuyer, P. (eds.) ICCSA 2003. LNCS, vol. 2667, pp. 358–367. Springer, Heidelberg (2003)

Explanation-Aware Design of Mobile myCBR-Based Applications

Christian Severin Sauer[1], Alexander Hundt[2], and Thomas Roth-Berghofer[1]

[1] School of Computing and Technology, University of West London, Ealing, London, UK
[2] Department of Computer Science, University of Hildesheim, Germany
{christian.sauer,thomas.roth-berghofer}@uwl.ac.uk, alexander.hundt@uni-hildesheim.de

Abstract. This paper focuses on extending the explanation capabilities of the *myCBR* SDK as well as on the optimisation of the *myCBR* SDK in the context of Android-based mobile application development. The paper examines the available knowledge for explanation generation within context-aware CBR systems. The need for the integration of new explanation capabilities is then demonstrated by an Android-based context- and explanation-aware recommender application. Upon the experience gathered during implementation of the prototype a process for the integration of explanation capabilities into the *myCBR* SDK is introduced. Additionally, constraints and requirements for the integration of explanation capabilities into *myCBR* are introduced. Within this process we distinguish domain dependent and domain independent knowledge. We do this with regard to the different requirements for the integration of explanation capabilities into *myCBR* for the two types of knowledge. The paper further details on our on-going effort to adapt the *myCBR* SDK for use on the Android platform.

Keywords: Explanation-aware computing, Case-based reasoning.

1 Introduction

Explanations are key elements to build trust and confidence during a conversation between humans. This is also true for a conversation between a human user and an information system. Being able to provide sufficient explanations can be seen as a vital capability of information systems to enhance the trust and confidence of users into the systems and their results [6]. Based on this well established fact, the question of how to integrate explanation capabilities into the artificial reasoner, as the core component of an information system, is an important current topic of research. The need for information systems that are able to provide explanations is even more acute with regard to the current evolution of mobile applications. Mobile applications inherit restrictions regarding their GUI as well as offer new ways for user interaction. Such new ways of interaction are, for example, given by touch screens and sensors integrated into mobile devices.

With the aforementioned facts at hand, software that is aiming at the design of information systems for mobile applications must be able to provide a range of explanation capabilities. The design software must further be optimised for development targeted at the new mobile application environment.

The aim of this paper is to show the process of integrating explanation capabilities into the *myCBR* SDK as well as optimising the SDK for mobile development. *myCBR* is an open-source case-based reasoning tool and SDK.[1] *myCBR* already has a range of explanation capabilities [12]. Within this paper we examine the division of explanation generation from system knowledge from the reasoner and additional explanation knowledge needed to generate further explanations within an explainer component [13].

We assume the systems knowledge to be given by the knowledge stored in the four knowledge containers of a CBR system [9]. We further differentiate between domain independent and domain specific knowledge. For the further development of the *myCBR* SDK we thus investigate how to integrate additional capabilities into the SDK's API for generating further explanations from: a) domain independent knowledge stored in the four knowledge containers [14] and b) domain specific explanation knowledge. While doing so, we also take into account the new possibilities offered by the use of a range of sensors within mobile devices. These sensors offer a variety of possibilities to make the mobile system context aware [2]. This increased the need further to provide explanations about its inner workings, but it also provides opportunities to base explanations on the available context knowledge [1].

The rest of this paper is structured as follows: In Section 2 we examine available problem solver knowledge and possibly needed domain specific explanation knowledge. In Section 3 we provide a brief overview of previous and related approaches to the use of explanations within CBR. Section 4 introduces a case study of the development of a mobile Android-based CBR application. The following section 5 gives an overview of a basic process for integrating additional explanation capabilities into the SDK while section 6 outlines our approach at adapting the *myCBR* SDK for mobile application development. Section 7 highlights some of the key challenges we identified for integrating further explanation capabilities into the *myCBR* SDK. We close with a summary in Section 8 and an outlook on further extensions of the *myCBR* SDK.

2 Available and Additional Knowledge for Explanation Generation

In the context of generating explanations in this paper, we distinguish two perspectives. The first perspective is that of a knowledge engineer modelling the CBR knowledge for use with the *myCBR* SDK. The second perspective is that of a software engineer integrating a ready-made CBR engine in a (mobile) application. Seen from such a software engineer's perspective a CBR-system achieves its given tasks by distributing those into smaller subtasks.

[1] `http://www.mycbr-project.net`

In the context of this work domain independent knowledge within a CBR system is defined as the knowledge present within the meta model of CBR. Thus by domain independent we mean the knowledge present within the established structures and knowledge representation approaches used within the CBR system itself. An example for such domain independent knowledge is given by the use of taxonomies within the knowledge container vocabulary. From such taxonomies explanations about 'part of' relations of concepts within the vocabulary can be derived. Thus, domain independent explanations regarding the reasoning of the CBR engine can be generated from the knowledge stored in the knowledge containers. These domain independent explanations can be derived from the structure of the knowledge representation used within the knowledge containers [11,14]. We exploit these formalisation approaches to derive functionalities to provide domain independent explanations for the *myCBR* SDK. Such additional explanation capabilities can add to the quality and usability of CBR applications developed with *myCBR*.

Aside from the possibility to generate explanations from the already present domain independent knowledge formalisation employed within the problem solver component, i.e., a CBR system, we further assume that there is a need to provide further, domain specific explanation knowledge. Such explanation knowledge is needed, for example, to generate the following kinds of explanations:

– justifications for design decisions a knowledge engineer has taken, e.g., the weights of an amalgamation function,
– explanations describing the real world domain specific dependencies of attributes of a case, and
– explanations based on the context knowledge the system is operating on.

We further distinguish the possible explanations generated from domain specific knowledge into two kinds, static and dynamic explanations. Static explanations are aiming at providing the knowledge engineer with the means to explain to the user design decisions and domain specific knowledge modelling decisions of the knowledge engineer. Such static explanations, provided as canned explanations [16], could explain a variety of characteristics of a CBR system, e.g.,

– provide canned explanations of the purpose of attributes of a case,
– provide rationales for the choice of attributes,
– justify the choice/modelling of functions used as similarity measures,
– explain weights assigned to attributes within an amalgamation function, and
– provide concept explanations for concepts and attributes of the domain (already implemented in *myCBR*).

For the SDK the knowledge engineer has to be provided with additional user interfaces to be able to acquire this domain specific explanation knowledge for the CBR engine.

Dynamic explanation capabilities are aiming at enhancing the user interaction with the CBR application. An example for such an interactive explanation capability is the well established approach of critiquing [5], i.e., suggesting the next most relevant attribute to the user. To allow for such interactive explanations *myCBR* has to provide mechanisms for providing the necessary explanation- and reasoner knowledge via its API to the software engineer.

There is a third kind of knowledge available to the software engineer to possibly generate further explanations. Focusing on mobile applications we have to keep in mind that most mobile devices today are equipped with a variety of sensors. Most commonly these sensors provide knowledge about the geospatial context of the device [17]. It is further possible to use the connectivity present in most mobile devices to access the social context of a user of such a mobile device. This context knowledge can be easily used to enhance an explanation-aware mobile CBR application into an application that is also context-aware [2].

As many explanations can be derived from context knowledge [16] it is of some importance to have at least a brief look at the possibility to do so. As context knowledge is not yet integrated into *myCBR* this brief look is mainly focused on the perspective of the software engineer and how he is able to make use of the context knowledge to generate explanations. An example connecting CBR to context knowledge is given by the idea of deriving personal data from the social context of a user e.g. his account at a social network. For example, extracting the age and sex of the user from such data to be used as values for a query to a CBR system, or going even further extracting and exploiting relevant knowledge about colleagues and friends (social context).

Based on the described available knowledge to generate explanations for mobile CBR applications we established the following main questions:

– How can the present domain independent knowledge formalisation within the knowledge containers of a CBR system be used to integrate further explanation capabilities into the *myCBR* SDK?
– How can additional explanation capabilities be provided to the knowledge engineer to add domain specific knowledge to the explainer component?
– What amount of effort is to be expected for: a) The integration of additional explanation capabilities into the *myCBR* SDK itself and b) What additional computational effort is to be expected for the applications developed with the extended *myCBR* SDK?

To answer these questions we conducted a small case study consisting of the development of an explanation-aware and context-aware mobile *myCBR* application. Based on the experiences we made during the implementation of this Android-based prototype we provide answers to the three questions.

3 Previous and Related Work

Enhancing CBR with explanations and using the CBR meta model and further domain specific knowledge within a CBR system is an area of high interest. Aamodt describes an explanation-driven approach to CBR [1]. The respective implementation, CREEK, applies extensive and explicit general knowledge. It features an explanation engine which splits the actual reasoning task into the subtasks activate, explain and focus. CREEK utilises explanations directly to support its case-based problem solver.

Sørmo et al. discuss the necessity and possibilities of both, employing and generating explanations within CBR systems [18]. They further discuss the complexity as well as the benefits of the use and generation of explanations within CBR. Given the wide variety of already implemented approaches to derive explanations described it becomes clear that one has to think about how to integrate these existing and possible further approaches to employ and derive explanations within an SDK for the design of CBR systems. The use of knowledge encoded within the meta model of CBR to generate explanations is presented, for example, in the work of Plaza et al. for the knowledge container case base [7].

Besides these general approaches to explanation generation from the CBR meta model we build on previous work of extending the explanation capabilities of the *myCBR 2* GUI [12]. In contrast to earlier versions of *myCBR* (Versions 1.x and 2.x) [19], which were plug-ins for the Open Source ontology editor Protégé [3], *myCBR 3* is a completely new and OSGi-based tool. *myCBR 3* still focuses on ease-of-use regarding the creation of the case model, modelling of similarity measures and testing the similarity-based retrieval by offering an easy-to-use graphical user interface. *myCBR 3*'s distinguishes SDK and GUI. Lillehaug et al. demonstrate how a plugin for Protégé 4.x using the *myCBR 3* SDK can facilitate explanations for the retrieval phase of a CBR system [4].

In this paper we look at enhancing the SDK. Our work is embedded in the vision of an explanation-aware architecture for extracting and case-based processing of experiences from Internet communities, SEASALT with explanations or, short, SEASALT^{exp} [10]. The SEASALT approach provides a framework for sharing experience using an agent-based system architecture layout [8]. In a SEASALT^{exp} instantiation, agents work together in a complex fashion in order to provide an answer to a user query. The agents can be any kind of program, for example, *myCBR* systems. Domain-independent/canned explanations are needed to provide answers to standard questions.

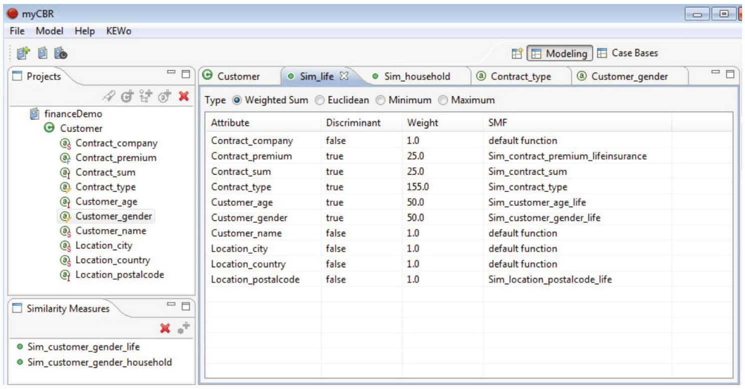
4 Case Study

In order to provide suggestions and examine the possibilities for implementing further explanation capabilities for *myCBR* we implemented a simple *myCBR*-based recommender system for the financial sector on a mobile device. In the prospected scenario of the prototype application the recommendation process leads to a very detailed contract. These contracts along with the information of the respective customers attributes, which are suited by the selected contract may be used as cases to fill a case base.

The first and most important functional requirement of the application was to implement the core library of *myCBR* in an Android environment. The application therefore had to be able to use and represent all functionalities that are provided by the *myCBR* API, thus enabling any Android device to serve as platform for further *myCBR*-based applications. The purpose of the application then was it to present a dialogue to the user in which he could chose between two types of contract: Life insurance and household insurance. After specifying the kind of contract the user could both manually enter his personal data and specify certain thresholds for the kind of contract he was looking for. Such thresholds were for example the monthly premium the user would like to pay. Additionally to the manual input of this information the application was designed to use context knowledge, given by the location of the user, his age and sex, automatically derived by the devices GPS sensor and retrieved from a social context. Whereas the location was of importance for the household insurance contract and the age and sex of the user was of importance for a possible life insurance of the user. The social context used within the prototype was a user's Google+ account[2]. When entering values for query attributes the application should only allow input that reflects the attribute's data type. If the application allows any kind of input it had to correctly convert this input so that it matched the data type of the attribute. After the completion of the query the retrieval results were presented to the user in that way that the two most similar contracts (cases) were presented to the user. The user thereafter could rate the quality of the retrieval by voting on it and assigning it with a value of either poor, fair or good.

4.1 Knowledge Domain and Modelling

Concerning the chosen knowledge domain as a financial service tool the application goal was to provide a basic recommendation system as a test bed for the integration of further explanation capabilities. Since it was assumed that a large contract data base may serve as case base for retrievals, the application's design had to reflect the typical attributes of a financial contract. Furthermore the chosen attributes had to be relevant for calculating a similarity. Figure 1 shows the case attributes and the amalgamation function chosen for a life insurance contract within the modelling view of the *myCBR* SDK.



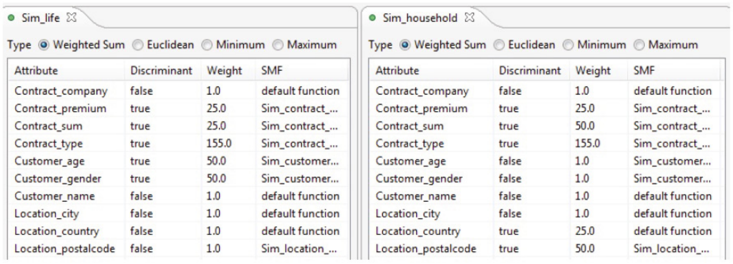
Type [✓ Weighted Sum] Euclidean Minimum Maximum

Attribute	Discriminant	Weight	SMF
Contract_company	false	1.0	default function
Contract_premium	true	25.0	Sim_contract_premium_lifeinsurance
Contract_sum	true	25.0	Sim_contract_sum
Contract_type	true	155.0	Sim_contract_type
Customer_age	true	50.0	Sim_customer_age_life
Customer_gender	true	50.0	Sim_customer_gender_life
Customer_name	false	1.0	default function
Location_city	false	1.0	default function
Location_country	false	1.0	default function
Location_postalcode	false	1.0	Sim_location_postalcode_life

Fig. 1. Modelling the life insurance amalgamation function within *myCBR 3*

[2] http://plus.google.com

To determine similarities between attribute values of different cases, local similarity measures have been created for all non-text attributes. Furthermore to reflect the differences in relevance of attributes for a business context some attributes have been provided with two local similarity measures. As it is possible to switch between amalgamation functions within *myCBR* based CBR systems during runtime we chose to integrate two amalgamation functions to test this feature with regard to its use for generating relevance explanations in the application. The weights we have chosen for the two amalgamation functions are designed to a) reflect the relevance of a given attribute for the overall retrieval as well as to specify a 'knock out' attribute. This attribute was 'contract type' which weight of 155 was exactly the sum of all other attributes and thus served as a filter attribute to suppress the retrieval of, for example, a life insurance case if the user chose household insurance before and vise versa. Depending on the selected contract type the appropriate similarity measures will be selected via a corresponding amalgamation function. Providing different similarity measures for the attributes is mandatory to take into account the varying relevancies for deviating attribute values depending on the business context. Figure 2 shows that the attribute with the highest ranking is the contract type for both functions. This decision was made to ensure that a wrong value for this attribute would always outweigh the remaining attributes regardless their similarity (155 is the sum of all the other weights). The remaining attribute weights were chosen to reflect the attributes relevance regarding their business context.As we were aiming on a basic prototype implementation we haven't considered other weighting strategies for the global similarity measure.


Sim_life

Type [✓ Weighted Sum] Euclidean Minimum Maximum

Attribute	Discriminant	Weight	SMF
Contract_company	false	1.0	default function
Contract_premium	true	25.0	Sim_contract_...
Contract_sum	true	25.0	Sim_contract_...
Contract_type	true	155.0	Sim_contract_...
Customer_age	true	50.0	Sim_customer...
Customer_gender	true	50.0	Sim_customer...
Customer_name	false	1.0	default function
Location_city	false	1.0	default function
Location_country	false	1.0	default function
Location_postalcode	false	1.0	Sim_location_...

Sim_household

Type [✓ Weighted Sum] Euclidean Minimum Maximum

Attribute	Discriminant	Weight	SMF
Contract_company	false	1.0	default function
Contract_premium	true	25.0	Sim_contract_...
Contract_sum	true	50.0	Sim_contract_...
Contract_type	true	155.0	Sim_contract_...
Customer_age	false	1.0	Sim_customer...
Customer_gender	false	1.0	Sim_customer...
Customer_name	false	1.0	default function
Location_city	false	1.0	default function
Location_country	true	25.0	default function
Location_postalcode	true	50.0	Sim_location_...

Fig. 2. The amalgamation functions for both business contexts

4.2 Necessary Explanations

In order to provide evidence for the hypothesis of this paper the application has to implement explanations that exploit the information provided by the underlying reasoning system. To satisfy the user's questions towards the outcome of the reasoning process appropriately it has to be portrayed first which kinds of explanations are suitable for certain components of a CBR-system [11,14].

Picturing the reasoning process it is clear that the similarity measures play an important role as they ultimately determine the result of the retrieval and therefore have to be the first choice in this case study to provide explanations. Similarity measures are divided into local measures and amalgamation functions. While local measures give information about the relation between different values of an attribute the amalgamation function indicates the relevance of each attribute. Because similarities are expressed in normalised numeric values they can be interpreted dynamically without having to store a pre-formulated explanation for each resulting characteristic.

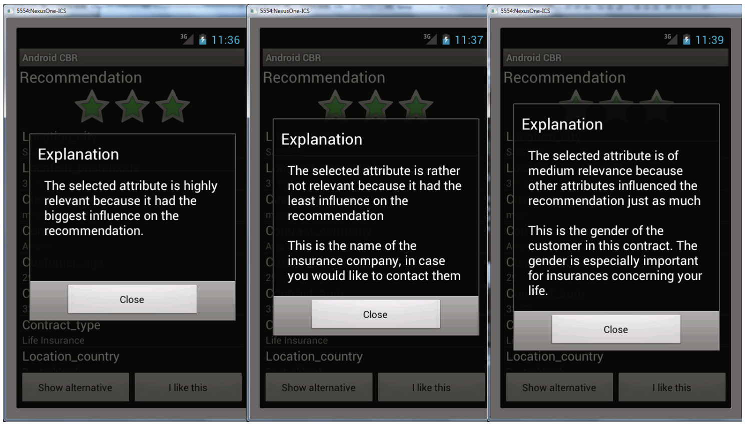
The second knowledge container that can improve the transparency of the reasoning process is the vocabulary. In the vocabulary the domain knowledge is modelled in inheritance or decomposition structures, meaning the entities in this model are in a 'is-a' respectively 'part-of' relation to their superior entities. These relations allow, e.g., for purpose explanations of attributes. Furthermore the vocabulary offers insight into the actual data types thus providing transparency for allowed values. Unsatisfying retrieval results from the case base may also be adapted to fit the user's query even better. The adaptation knowledge that is necessary for this conversion therefore would automatically provide the ability to explain these adaptations. However representing this knowledge requires the greatest effort while creating a CBR-system. For this case study the implementation of means to explain why a recommendation was offered at the end of the retrieval process has been chosen to be the main focus. The application will therefore focus on explaining similarities and the composition of amalgamation functions.

4.3 Explaining Attribute Relevance

To examine the possibility of integrating new explanation capabilities into first the prototype application and to later on abstract them for integration into the *myCBR* SDK we implemented an additional explanation capability using domain specific knowledge. The aim of this additional explanation capability was to combine already available canned explanations about the nature of attributes with a new form of canned explanations, explaining their relevance within a query. As these relevance explanations are based on domain specific knowledge they provide as a new explanation capability not yet present in *myCBR* (except canned conceptualisation explanations for concepts and attributes).

To demonstrate the approach of relevance explanations a simple check for the highest respectively lowest weight of an attribute was conducted. After the retrieval activity has been loaded within the application the user may tab on any attribute, thus expressing his request for further explanations for that attribute. The relevance of that attribute was then calculated dynamically by comparing the weight of all attributes against each other, thus searching for the attribute with the highest weight and the attribute with the lowest weight. If the current attribute had either of those values a predefined text was added to the returned explanation text, stating that the attribute is 'highly relevant', 'not relevant' or, if neither of those values matched the attributes weight, 'of medium relevance'.

We choose to separate the actual computation of relevance from matching the returning value with a predefined text. This modularisation emphasises that either method, relevance determination and interpretation of the result, is exchangeable by more sophisticated future functions.



Explanation

The selected attribute is highly relevant because it had the biggest influence on the recommendation.

Explanation

The selected attribute is rather not relevant because it had the least influence on the recommendation

This is the name of the insurance company, in case you would like to contact them

Explanation

The selected attribute is of medium relevance because other attributes influenced the recommendation just as much

This is the gender of the customer in this contract. The gender is especially important for insurances concerning your life.

Fig. 3. Attribute explanations/justifications

We tried to establish the level of satisfaction and its accompanying additional computational effort for the new explanations. Therefore we provided the user with a possibility to rate the retrieval results. The outcome of this implementation provides the explanation capabilities of the application towards the user that are illustrated in Figure 3. The illustration shows the composite explanation that is displayed if an attribute is selected on the screen. In the centre and right screenshot each first paragraph represents the natural explanation of the determined relevance, while the second paragraph represents the concept explanation that has been defined in the *myCBR* desktop application.

Despite the convenient result for the user, the chosen flat hierarchy of attributes does not yet offer the full range of possibilities to generate explanations. We have chosen the simple flat representation of attributes for simplicity reasons aiming at not obfuscating the process of explanation generation by an over complex first approach to it. Given the already satisfying results we achieved with the described basic representation of attributes it is obvious that exploiting more complex modelling approaches for the attributes, which are to be expected in complex recommender systems, promise more versatile possibilities for the generation of explanations. As we are not trying to introduce better solutions to the use of explanations within recommender systems, we chose however to implement the basic approach to highlight the possibility to communicate/ making available design choices made by the knowledge engineer who designed the CBR system, enriched with additional explanation knowledge, to the software engineer who implements a mobile CBR application.

In order to provide a more valid assessment of an attribute's relevance more complex calculations need to be implemented that take into account not only the single weight of an attribute within the amalgamation function but also assess the range of weights as well as possible clusters of attributes that have only slightly deviating weights. Statistical methods might be suitable and should be further investigated. Furthermore in order to improve the additional load that comes with the explanation abilities it should be attempted to calculate the relevance once for every attribute after the retrieval and store them, rather than starting calculation every time a user requests an explanation for a single attribute. Overall it can be said that the ability to justify the result through similarity measure explanations can be implemented easily and that it provides a basis on which building blocks of naturally expressed texts can be used to compose comprehensible and thus satisfying explanations for the user.

5 Integrating New Explanation Capabilities into the SDK

In this section we propose the abstraction of the prototypical new explanation capability to a level that allows for its integration into the *myCBR* SDK. Further we derive a process for such an integration task on a generic level to give an insight in the important aspects of the integration process we discovered during our research and development. We deem these findings and process description useful for any open-source developer interested in extending the explanatory capabilities of the *myCBR* SDK.

The overall findings for the whole process of integrating new explanation capabilities into *myCBR* are that the coupling between the calculation of values for explanations and the mapping of canned texts to these should be minimal to allow for maximum modularisation and thus wide reuse of the respective calculation and assignment classes for explanation generation. Furthermore before integrating a new explanation capability for domain independent knowledge into the SDK it is advisable to carefully examine the code structure of the SDK to integrate the necessary knowledge representation with minimal effort into the existing classes of the SDK. The same approach of considering the 'best' spot where to integrate new functionalities is advisable for additions to the GUI of the SDK necessary to provide the interactions needed by the knowledge engineer to model the domain specific explanation knowledge into the system.

The overall process comprises these high level steps: First, possible constraints regarding the explanation capability to be added have to be taken into account such as the availability of the knowledge to generate the explanations. Thus an SDK developer aiming to integrate a new explanation capability into the SDK has to check if that knowledge is either already available from the SDK for domain independent explanations or how it can be added for domain specific explanations. The domain specific explanations further need to be checked what kind of formalisation approach is to be chosen for them. A further constraint is the computational cost. It has to be established beforehand if the generation of the explanation is sufficiently adding to the usability of SDK/applications to justify its additional computational effort. This is especially important with regard to the mobile context the SDK is to be used in and its limitations with regard to computational resources. The last constraint is the actual usefulness of a new type of explanation capability on a generic level. To establish this one has to think about, e.g., how often a user might want to get explanations of the kind to be provided by the new planned explanation capability and if these frequencies justifies the development of the capability.

If a new explanation capability is conforming to all the above mentioned constraints, a *myCBR* SDK developer should first implement it in a prototypical way. This way the new capabilities internal use of explanatory knowledge as well as, for example, its functions employed to generate the actual explanation can be 'test driven' by the SDK developer on an actual example and the process of the explanation capability can be evaluated and refined before its integration into the *myCBR* SDK.

Once the prototypical implementation of the new explanation capability is established and refined a SDK developer has to consider how the knowledge used within it might be abstracted to allow for frequent reuse of the new capability by decoupling it as most from the domain specific knowledge as possible. Of course for new domain independent explanation capabilities, relying on the knowledge formalisation within the knowledge containers of the CBR system, such an abstraction is not necessary. If the developer has created an abstracted form of the tested new explanation capability she has to provide and thus add (at least for domain specific knowledge based explanations) new means of storing and accessing the domain specific explanatory knowledge within the SDK. For this purpose it is advisable to reuse and adapt the most similar or suitable existing approaches to knowledge storage already existent within the SDK before implementing an entirely new form of knowledge storage.

For domain independent explanation capabilities the SDK developer has to establish if the CBR system provides the necessary knowledge for the new explanation capability. Further he has to make sure she integrates the necessary explanation generation and access methods to generate explanations from the problem solver knowledge and provide the means to access these via the API for the software engineer who wants to make use of the explanations.

For domain specific explanation capabilities the developer hast to provide additionally the storage and access functionalities needed to be added to the Explainer component to extent its explanatory knowledge. Additionally to the similarly necessary integration of the necessary explanation generation and access methods to generate explanations from the explainer's knowledge and the need to provide the means to access these via the API fort the software engineer, the developer in this case has also to provide the necessary additional GUI functionalities for the knowledge engineer so that he can incorporate the domain specific knowledge into the CBR system he designs.

6 Implementation Details of Mobile Functionalities

In this section we provide an outline of the adaptation of the *myCBR* SDK and API towards a more versatile version with regard to the development of mobile applications employing CBR as their reasoning component.

For the implementation of mobile *myCBR* applications we have for now branched off a new version of the *myCBR* SDK. This new version is aiming at the necessary decoupling of the functionalities needed for the CBR engine from the necessary GUI elements in a mobile application. We further use this branch of the mobile *myCBR* SDK as a test bed for enhancements with regard to the usability as well as the new explanation capabilities described so far. For this purpose we are currently revising the core elements used in the described prototype application decoupling functional code from GUI related code. This revision aims also at abstracting the core elements necessary up to the API level to make them available as interfaces for mobile application developers.

In this new revision we are following the basic layout of the existent *my-CBR* with the following changes already implemented: We added a package 'mobileCbr' which contains the example prototype application as an example for developers as to how to integrate *myCBR* in their application. We further added a 'core' package that provides all classes that are not related to an Android-typical layout manager, thus aiming at decoupling functionality from GUI and layout related classes. Within the core package we implemented the a Project-Manager which is used for the handling of *myCBR* project data. It provides now an interface to application developers to serve as a 'one stop' mean to access, import and handle *myCBR* projects regardless of their storage format.

Within the ProjectManager we are currently implementing the ConceptManager which is aiming at providing all necessary functionalities to handle the concepts and attributes present within a *myCBR* project and additionally offer further functionalities. These further functionalities will be given by a unique referencing approach for concepts allowing for a new PathManager which will manage nested concepts from a *myCBR* project within a mobile application. This managing of nested concepts aims at allowing for an approach to centrally handle the available information about concepts and their visualisation with regard to the GUI techniques offered by the Android platform.

We also added a package 'attributes' which abstracts the handling of the attributes to the API level of *myCBR*Ȧs with the ConceptManager, this package also aims at centrally provide access to all attributes for application developers with regard to the GUI techniques available within Android. For example, for a numerical attributes the application developer can specify that only a numerical keyboard should be displayed to the user in the GUI.

By integrating the 'query' package into the core package we introduce a new approach of query composition. The idea behind this new query composition is to collect a series of attribute-value pairs via the setter-methods invoked during a number of activities, whereas each activity is the presentation of a specialised GUI element to ask a value for a given attribute from the user, and then assemble the query upon request, e.g. if the user finished specifying attributes for his query.

As a retrieval performed by the *myCBR* engine can become a computational intensive operation we are currently planning to provide an asynchronous thread that handles the retrieval. Being asynchronous the retrieval task can be handled as a separate thread allowing for the remaining parts of a mobile application to continue being accessible while the, eventually time consuming, retrieval task is processed by the mobile device. Another approach to handle the possibly time consuming retrieval task is given by a possible client server approach where the *myCBR* engine would be running on a server PC and only be accessed by the client mobile application to post the query and receive the retrieval results generated on the *myCBR* running server. Such an alternative client server approach has also currently been tested successfully by Satzky [15].

7 Challenges

Abstracting the additional explanation capability used within the prototype to the level of the *myCBR* API in accordance with the already existing structures and mechanisms to provide concept explanations proved to be more complex than expected. Nevertheless we are expecting the abstraction of the domain specific knowledge based explanations to be not an easy task as every such addition requires the provision of the necessary storage structures also to be added to the *myCBR* SDK. To amend these difficulties we think that the explanation manager could serve as an explanation agent in the SEASALT^{exp} architecture, which can dynamically provide separate statements about a relevance rating (e.g., a calculated level of significance) and a rating interpretation (instead of creating another explanation class for relevance explanations).

Providing access to the data of a *myCBR* project within an Android application was another challenge. Due to the encapsulation of activities within the Android platform it was initially difficult to transfer information from the *myCBR* project between said activities. By providing new approaches to the handling of the *myCBR* project data described in Section 6, namely the Project-, Concept-, and AttributeManager we were able to provide access to this data within an Android application.

We are additionally testing the computational effort of retrieval operations within *myCBR* projects at the current time to establish the necessity of an asynchronous retrieval task for an Android-based mobile *myCBR* application.

8 Summary and Outlook

In this paper we showed that a *myCBR* application can be run on an Android mobile device without any restrictions to the functionality of *myCBR* components. We described a mobile, context-sensitive and explanation-aware recommender system. With this prototype a simple approach on how to implement basic *myCBR* functionalities on an Android device has been illustrated that can used as a reference for further mobile applications. We further proposed an approach that enables developers to add explanations of the relevance of weighted attributes to an application. We then examined the possibility to abstract this prototypical new explanation capability for integration into the *myCBR* SDK. We additionally provided a high level overview of the process associated with such a task. We further detailed on our current effort to adapt *myCBR* to better suit the needs of Android-based mobile development.

Besides the above mentioned results the paper did not detail on the notion of context awareness of the prototype application due to space limitations. We were nevertheless able to integrate, based on theoretical definitions of context, four pre-defined categories to abstract different contexts into the application. These categories could also be integrated in the *myCBR* SDK to provide a context information container for future applications and enable future research work on context knowledge based explanations.

During the implementation of explanations of the similarity weights we considered to use descriptive languages to differentiate explanations for varying relevance's of an attribute. However as we were aiming for a simple prototypical implementation the idea was substituted with predefined String attributes. Nevertheless this approach should be further investigated in future works on this subject as employing descriptive languages offer some great opportunities to a) enhance the explanation capabilities of *myCBR* and b) exploit the use of descriptive languages within the semantic web as a way of knowledge acquisition for possible future explanation capabilities.

As we have advanced the mobile *myCBR* branch, as described earlier, we are considering integrating critiquing explanations as well as critiquing based suggestions of attributes into the SDK. This is assumed to be easily accomplished due to the already implemented separated handling of attributes within mobile *myCBR* applications and the planned implementation of a 'Query factory' allowing for a fine granular composition of a query. Additionally the prototype applications explanation capability of an amalgamation function can easily be adapted to provide the necessary information about the importance of attributes to queue them for the planned suggestion scheme.

Enabling a flexible and light-weight reasoning tool to operate on a smartphone device offers a variety of new opportunities of which only a few have been addressed in the scope of this work.

Acknowledgement. This work was supported by the Research Excellence Framework (REF) of the UK (Explanation-aware myCBR Development project). We would like to thank Klaus-Dieter Althoff and Kerstin Bach at the German Research Centre for Artificial Intelligence DFKI GmbH for their valuable input and feedback.

References

1. Aamodt, A.: Explanation-driven case-based reasoning. Topics in Case-Based Reasoning, 274–288 (1994)
2. Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P.: Towards a Better Understanding of Context and Context-Awareness. In: Gellersen, H.-W. (ed.) HUC 1999. LNCS, vol. 1707, pp. 304–307. Springer, Heidelberg (1999)

3. Gennari, J.H., Musen, M.A., Fergerson, R.W., Grosso, W.E., Crubézy, M., Eriksson, H., Noy, N.F., Tu, S.W.: The evolution of Protégé an environment for knowledge-based systems development. Int. J. Hum.-Comput. Stud. 58(1), 89–123 (2003)
4. Lillehaug, M.B., Roth-Berghofer, T., Kofod-Petersen, A.: Myeacbr – myCBR as explanation-aware Protégé plugin. In: Petridis, M., Roth-Berghofer, T., Wiratunga, N. (eds.) Sixteenth UK Workshop on Case-Based Reasoning (2011), http://ceur-ws.org/Vol-829/ (last access: April 13, 2012)
5. McCarthy, K., Reilly, J., McGinty, L., Smyth, B.: Experiments in dynamic critiquing. In: Proceedings of the 10th International Conference on Intelligent User Interfaces, pp. 175–182. ACM (2005)
6. Moore, J.D., Swartout, W.R.: Explanation in expert systems: A survey. Research Report RR-88-228. University of Southern California, Marina Del Rey, CA (1988)
7. Plaza, E., Armengol, E., Ontañón, S.: The explanatory power of symbolic similarity in case-based reasoning. Artificial Intelligence Review 24(2), 145–161 (2005)
8. Reichle, M., Bach, K., Althoff, K.D.: Knowledge engineering within the application independent architecture seasalt. In: Baumeister, J., Nalepa, G.J. (eds.) Int. J. Knowledge Engineering and Data Mining, pp. 202–215. Inderscience Publishers (2010)
9. Richter, M.M.: 1. Introduction. In: Lenz, M., Bartsch-Spörl, B., Burkhard, H.-D., Wess, S. (eds.) Case-Based Reasoning Technology. LNCS (LNAI), vol. 1400, pp. 1–15. Springer, Heidelberg (1998)
10. Roth-Berghofer, T., Sauer, C.S., Althoff, K.D., Bach, K., Newo, R.: Seasaltexp - an explanation-aware architecture for extracting and case-based processing of experiences from internet communities. In: Proceedings of the LWA 2011 - Learning, Knowledge, Adaptation (2011)
11. Roth-Berghofer, T.R.: Explanations and Case-Based Reasoning: Foundational Issues. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 389–403. Springer, Heidelberg (2004)
12. Roth-Berghofer, T.R., Bahls, D.: Explanation capabilities of the open source case-based reasoning tool mycbr. In: Petridis, M., Wiratunga, N. (eds.) Proceedings of the Thirteenth UK Workshop on Case-Based Reasoning UKCBR 2008, pp. 23–34. University of Greenwich, London (2008)
13. Roth-Berghofer, T.R., Richter, M.M.: On explanation. Künstliche Intelligenz 22(2), 5–7 (May 2008)
14. Roth-Berghofer, T., Cassens, J., Sørmo, F.: Goals and kinds of explanations in case-based reasoning. In: WM, pp. 264–268 (2005)
15. Satzky, J.: Implementierung und Erweiterung der fallbasierten Rezeptsuchmaschine CookIIS für mobile Endgeräte, bachelor thesis (2011)
16. Schank, R.: Explanation patterns: Understanding mechanically and creatively. Lawrence Erlbaum (1986)
17. Segev, A.: Identifying the Multiple Contexts of a Situation. In: Roth-Berghofer, T.R., Schulz, S., Leake, D.B. (eds.) MRC 2005. LNCS (LNAI), vol. 3946, pp. 118–133. Springer, Heidelberg (2006)
18. Sormo, F., Cassens, J., Aamodt, A.: Explanation in case-based reasoning–perspectives and goals. Artificial Intelligence Review 24(2), 109–143 (2005)
19. Stahl, A., Roth-Berghofer, T.R.: Rapid Prototyping of CBR Applications with the Open Source Tool myCBR. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 615–629. Springer, Heidelberg (2008)

A Competitive Measure to Assess the Similarity between Two Time Series

Joan Serrà and Josep Lluís Arcos

IIIA-CSIC, Artificial Intelligence Research Institute, Spanish National Research Council, Bellaterra, Barcelona, Spain
{jserra,arcos}@iiia.csic.es

Abstract. Time series are ubiquitous, and a measure to assess their similarity is a core part of many systems, including case-based reasoning systems. Although several proposals have been made, still the more robust and reliable time series similarity measures are the classical ones, introduced long time ago. In this paper we propose a new approach to time series similarity based on the costs of iteratively jumping (or moving) between the sample values of two time series. We show that this approach can be very competitive when compared against the aforementioned classical measures. In fact, extensive experiments show that it can be statistically significantly superior for a number of data sources. Since the approach is also computationally simple, we foresee its application as an alternative off-the-shelf tool to be used in many case-based reasoning systems dealing with time series.

1 Introduction

Data in the form of time series pervades almost any scientific domain [9,11]. Observations that unfold over time usually represent valuable information subject to be analyzed, classified, predicted, or interpreted [4,8,10]. Real-world examples include financial data (e.g. stock market fluctuations), medical data (e.g. electrocardiograms), computer data (e.g. log sequences), or motion data (e.g. geolocation of moving objects). Dealing with time series represents a challenge for these and many other scientific domains.

Dealing with time series has also been a challenge for the case-based reasoning (CBR) community. Apart from the two workshops on time series prediction held in the 2003 and 2004 International Conferences on CBR [6,7], several CBR systems have coped with cases involving time series or sequential information. Xiong and Funk [22] presented a CBR system managing symbolic time series from a medical domain. Their approach was based on the identification of key sub-sequences and on the transformation of the original time series into a more compact representation. In a preliminary work [3], the authors demonstrated the value of incorporating knowledge discovery techniques to CBR and, in particular, the value of the technique they used to extract significant sub-sequences, which allowed them to automatically discover non-trivial regularities. Montani et al. [13] used the discrete Fourier transform (DFT) in their CBR system to reduce the comparison of (entire) time series to just the first Fourier coefficients, and also to implement indexing structures for the time series. Other CBR systems reduce the dimensionality by transforming the time series using temporal abstractions [1] or hierarchical symbol abstractions [21]. A further interesting approach dealing with time series is the CASEP2 system [23], which was proposed as a hybrid system that, combining CBR and artificial neural networks, performed time series classification in an efficient way. Also related to time series is the Ceaseless CBR model introduced by Martín and Plaza [12], which processes a continuous data stream holding several problem descriptions.

A core issue when dealing with time series is determining their pairwise similarity, i.e. the degree to which a given time series resembles another one. In fact, a time series dissimilarity (or similarity) measure is central to many mining, retrieval, classification, and clustering tasks [4,10]. However, deriving a measure that correctly reflects time series dissimilarities is not straightforward. Apart from dealing with a high dimensionality (time series can be roughly considered as multi-dimensional data), the calculation of such measures needs to be fast, robust, and efficient. Moreover, there is the need for generic dissimilarity measures, so that they can be readily applied to any data set, being this application the final goal or just an initial approach to a given task.

With years, several time series dissimilarity measures have been proposed. However, it seems that the most common measures, proposed long time ago, turn out to be the most competitive ones [10,20]. Wang et al. [20] perform an extensive comparison of classification accuracies for 13 different time series dissimilarity measures across 38 contrasting data sources (we also refer the interested reader to [20] for pointers to the original references proposing or using such measures in the context of mining time series data). After reporting the results, one of the main conclusions of the study is that, despite of the new proposals, the Euclidean and dynamic time warping (DTW) [14,15] dissimilarity measures are extremely difficult to beat, remaining two of the most robust, simple, generic, and efficient measures.

In this paper we propose a new time series dissimilarity measure based on minimum jump costs (MJCs). The main idea behind this measure is that it reflects the cumulative cost of iteratively 'jumping' from one time series to the other, starting at the beginning of a time series until the end of any of them is reached, and without going backwards. As it will be shown by extensive and rigorous experiments, MJC clearly outperforms the Euclidean distance. Moreover, we will see that MJC can statistically significantly outperform DTW for a number of data sets. This, jointly with the computationally simple operations behind MJC, makes it a good candidate measure to be incorporated to any standard toolkit for time series similarity, retrieval, or classification and, by extension, to any case-based reasoning system dealing with time series.

The remaining of the paper is organized as follows. We first present some scientific background by outlining the calculation of the Euclidean and DTW dissimilarity measures (Sec. 2). The description of the MJC dissimilarity measure comes next (Sec. 3). We then explain our evaluation methodology and present the obtained results (Secs. 4 and 5, respectively). A conclusion section ends the paper (Sec. 6).

2 Scientific Background

Across years, several dissimilarity measures have been proposed, the most simple ones being variants of the $\mathrm{L_p}$ norm,

$$d_{\mathrm{L_p}}(\mathbf{x},\mathbf{y}) = \sqrt[p]{\sum_{i=1}^{M}(x_i - y_i)^p}, \tag{1}$$

where p is a positive integer, M is the length of the time series, and x_i and y_i are the i-th element of time series $\mathbf{x}$ and $\mathbf{y}$, respectively. Usually $p = 2$, yielding the Euclidean distance, one of the first generic dissimilarity measures proposed for time series [2]. In case $\mathbf{x}$ and $\mathbf{y}$ were not of the same length, one can always re-sample one to the length of the other, an approach that works well for many data sources [10]. The Euclidean distance is one of the most used and efficient time series dissimilarity measures. Indeed, its accuracy may be very difficult to beat in some scenarios, specially when the length of the time series increases [20]. Nonetheless, we believe that such affirmation needs to be carefully assessed with extensive experiments and under broader conditions, considering different distance-exploiting algorithms.

Another classical option for computing the dissimilarity between two time series is dynamic time warping (DTW) [14,15]. DTW belongs to the group of so-called *elastic* dissimilarity measures [10,20], and works by optimally aligning (or 'warping') the time series in the temporal dimension so that the accumulated cost of this alignment is minimal. In its most basic form, this cost can be obtained by dynamic programming, recursively applying

$$D_{i,j} = \delta(x_i, y_j) + \min\{D_{i,j-1}, D_{i-1,j}, D_{i-1,j-1}\} \tag{2}$$

for $i = 1, \ldots, M$ and $j = 1, \ldots, N$, being M and N the lengths of time series $\mathbf{x}$ and $\mathbf{y}$, respectively. Except for the first cell, which is initialized to $D_{0,0} = 0$, the matrix D is initialized to $D_{i,j} = \infty$ for $i = 0, 1, \ldots, M$ and $j = 0, 1, \ldots, N$. In the case one deals with uni-dimensional time series, the sample (or local) dissimilarity function $\delta()$ is typically taken to be the square of the difference between x_i and y_j, i.e. $\delta(x_i, y_j) = (x_i - y_j)^2$. In the case we deal with multidimensional time series or we have some domain-specific knowledge, the sample dissimilarity function $\delta()$ must be chosen appropriately, although many times the Euclidean distance is used.

The final dissimilarity measure between time series $\mathbf{x}$ and $\mathbf{y}$ typically corresponds to the total accumulated cost $d_{\mathrm{DTW}}(\mathbf{x},\mathbf{y}) = D_{M,N}$. A normalization of $d_{\mathrm{DTW}}(\mathbf{x},\mathbf{y})$ can be performed on the basis of the alignment of the two time series, which is found by backtracking from $D_{M,N}$ to $D_{0,0}$ [14]. However, in preliminary analysis we found the normalized variant to be equivalent, or sensibly less accurate, than the unnormalized one.

Several constrains can be applied in the computation of D. A common operation is to introduce a window parameter w [15], such that the recursive formula of Eq. 2 is only applied for $i = 1, \ldots, M$ and

$$j = \max\{1, i' - w\}, \ldots, \min\{N, i' + w\}, \tag{3}$$

where i' is progressively adjusted for dealing with different time series lengths, i.e. $i' = \lfloor iN/M \rceil$, using $\lfloor\ \rceil$ as the round-to-the-nearest-integer operator. Notice that if $w = 0$ and $N = M$, $D_{M,N}$ will correspond to the squared Euclidean distance. Notice furthermore that when $w = N$ we are using the unconstrained version of DTW.

The introduction of constrains, and specially of the window parameter w, generally carries some advantages [10,14,20]. For instance, they prevent from 'pathological alignments' (which typically go beyond the main diagonal of D) and, therefore, they usually provide better dissimilarity estimates. In addition, DTW constrains allow for reduced computational costs, since only a percentage of the cells in D needs to be examined.

DTW stands as the main benchmark against to which new dissimilarity measures need to be compared with [20]. Very few measures have been proposed that systematically outperform DTW for a number of different data sources. However, these measures are usually more complex than DTW, sometimes requiring extensive parameter tuning of one or more parameters. Additionally, no careful, rigorous, and extensive evaluation of the accuracy of these measures had been initially done, and further studies fail to assess the statistical significance of their improvement [20]. In this paper we pay special attention to all these aspects in order to formally assess the benefits of the measure we propose.

3 Minimum Jump Costs Dissimilarity

We now detail the calculation of the minimum jump costs (MJC) dissimilarity measure. The main idea behind MJC is that, if a given time series $\mathbf{x}$ resembles $\mathbf{y}$, the cost of iteratively 'jumping' between their samples should be small. In other words, if $\mathbf{x}$ and $\mathbf{y}$ are similar, we could only draw short lines between them when placed on the same time axis. Intuitively, these jumps (or lines) should be iteratively done from the beginning of the time series until we reach an end, otherwise we would be discarding some possibly relevant parts of the time series. Similarly, if we kept jumping (or drawing lines) both forward and backwards, we could be iterating an infinite number of times. Thus we force to jump (or draw) in the forward direction only. Finally, since we want a single number reflecting the global dissimilarity between $\mathbf{x}$ and $\mathbf{y}$, the most straightforward solution is to add the costs of performing a jump (or the lengths of the lines). A more formal definition follows.

Let $\mathbf{x} = x_1, \ldots x_M$ and $\mathbf{y} = y_1, \ldots y_N$ be two time series of potentially different lengths M and N, respectively. We define the minimum jump costs (MJC) dissimilarity measure d_{XY} as the cumulative minimal cost for iteratively jumping from one time series to the other, i.e.

$$d_{\mathrm{XY}} = \sum_i c_{\min}^{(i)}, \tag{4}$$

where $c_{\min}^{(i)}$ is the cost of the i-th jump, which should be minimal. Supposing that for the i-th jump we are at time step t_x of time series $\mathbf{x}$ and that we previously visited time step $t_y - 1$ of $\mathbf{y}$,

$$c_{\min}^{(i)} = \min\left\{ c_{t_x}^{t_y}, c_{t_x}^{t_y+1}, c_{t_x}^{t_y+2}, \dots \right\}, \tag{5}$$

where $c_{t_x}^{t_y+\Delta}$ is the cost of jumping from x_{t_x} to $y_{t_{y+\Delta}}$ and $\Delta = 0, 1, 2, \dots$ is an integer time step increment such that $t_y + \Delta \leq N$. Notice that we can only go forward, i.e. we cannot visit time series samples before t_x or t_y. After a jump is made, t_x and t_y are updated accordingly, i.e. t_x becomes $t_x + 1$ and t_y becomes $t_y + \Delta + 1$. This way we enforce that no time step position is repeated and that the iterative algorithm does not go backwards. As mentioned, the formulation of Eq. 5 corresponds to a jump from time series $\mathbf{x}$ to $\mathbf{y}$. In case we want to jump from $\mathbf{y}$ to $\mathbf{x}$, only t_x and t_y need to be swapped in Eq. 5. We start the iterations at $t_x = 1$, considering $t_y = 1$, and jump between $\mathbf{x}$ and $\mathbf{y}$ until an end of a time series is reached, i.e. until $t_x = M$ or $t_y = N$.

To define a jump cost $c_{t_x}^{t_y+\Delta}$ we consider the temporal and the magnitude dimensions of the time series. Therefore we define

$$c_{t_x}^{t_y+\Delta} = (\phi\Delta)^2 + \delta(x_{t_x}, y_{t_y+\Delta}), \tag{6}$$

where ϕ represents the cost of advancing in time and $\delta()$ is the magnitude dissimilarity function, which we take to be $\delta(x_{t_x}, y_{t_y+\Delta}) = (x_{t_x} - y_{t_y+\Delta})^2$, equivalently to what we do with DTW (Eq. 2). We set ϕ proportional to the standard deviation σ expected for the time series,

$$\phi = \beta \frac{4\sigma}{\min\{M, N\}}, \tag{7}$$

and introduce the parameter $\beta \in [0, \infty)$, $\beta \in \mathbb{R}$, which controls how difficult is to advance in time. A value of $\beta = 0$ implies no cost ($\phi = 0$), whereas values of $\beta \to \infty$ imply that only samples at time stamp t_y will be considered ($\Delta = 0$, see Eq. 6). This latter case makes d_{XY} equal to the squared Euclidean distance between $\mathbf{x}$ and $\mathbf{y}$.

Finally, notice that d_{XY} is asymmetric. Depending whether we start at x_1 or y_1 we will obtain different values. To obtain a symmetrized dissimilarity measure we use

$$d_{\mathrm{MJC}}(\mathbf{x}, \mathbf{y}) = \min\left\{ d_{\mathrm{XY}}, d_{\mathrm{YX}} \right\}, \tag{8}$$

where d_{XY} and d_{YX} are the cumulative MJCs obtained by starting at x_1 and y_1, respectively. Measures d_{XY}, d_{YX}, and by extension $d_{\mathrm{MJC}}(\mathbf{x}, \mathbf{y})$ can be considered as elastic measures [20].

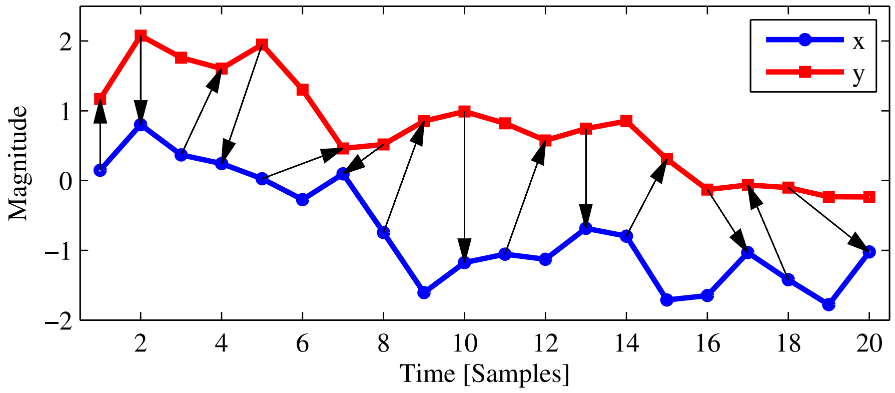


Fig. 1. Example of the recursive jumps performed between time series $\mathbf{x}$ and $\mathbf{y}$. The algorithm starts with time series $\mathbf{x}$ at $t_x = 1$ (x_1) and ends when $t_x = M$ or $t_y = N$ (x_{20} in the example).

Fig. 1 helps explaining the calculation of d_{XY}. Suppose that we are at sample x_5 and that we previously jumped from y_5 to x_4 (hence the values of $t_x = 5$ and $t_y = 6$). We now want to jump to time series $\mathbf{y}$ again. In addition, we want the cost of the jump to be minimal. Therefore, we evaluate Eq. 6 for all possible $t_y + \Delta$, i.e. for $\Delta = 0, 1, \ldots, 14$ (from time steps 6 to 20). With that we obtain that the best jump option is y_7 ($\Delta = 1$). After the jump we update t_x to 6 and t_y to 8, the next time steps that will be considered in the following iteration.

Algorithms 1 and 2 provide the implementation details for the whole dissimilarity calculation. Notice that we do not need to compute all possible costs, thanks to the introduction of the monotonically increasing term $\phi\Delta$ which, furthermore, can be precomputed. Notice also that since d_{XY} is cumulative, an early abandoning strategy can be additionally implemented to speed-up computations [10]. This way, if only the first nearest neighbor of a time series was required, we would only accumulate costs until we reached the smallest $d_{\mathrm{XY}}^{\mathrm{best}}$ found so far, exiting the process before its end since the current cumulative cost d_{XY} could not be smaller than $d_{\mathrm{XY}}^{\mathrm{best}}$. See [10] for more details about this procedure.

4 Evaluation Methodology

The efficacy of a time series dissimilarity measure is commonly evaluated by the classification accuracy it achieves [10,20]. For that, the error ratio of a distance-based classifier is calculated for a given labeled data set, understanding the error ratio as the number of wrongly classified items divided by the total number of tested items. The standard choice for the classifier is the one nearest neighbor (1NN) classifier. Following [20], we can enumerate several advantages of using

Algorithm 1. dXY($\mathbf{x}$,$\mathbf{y}$)

```
Input: Time series $\mathbf{x} = x_1, \ldots x_M$ and $\mathbf{y} = y_1, \ldots y_N$
Output: Cumulative MJC dissimilarity measure $d_{\text{XY}}$
 1: $t_x, t_y \leftarrow 1$
 2: $d_{\text{XY}} \leftarrow 0$
 3: while $t_x \leq M$ and $t_y \leq N$ do
 4:    $d_{\text{XY}} \leftarrow d_{\text{XY}}+\text{cmin}(\mathbf{x},t_x,\mathbf{y},t_y)$
 5:    if $t_x > M$ or $t_y > N$ then
 6:       break
 7:    end if
 8:    $d_{\text{XY}} \leftarrow d_{\text{XY}}+\text{cmin}(\mathbf{y},t_y,\mathbf{x},t_x)$
 9: end while
10: return $d_{\text{XY}}$
```

Algorithm 2. cmin($\mathbf{x}$,t_x,$\mathbf{y}$,t_y)

```
Input: Time series $\mathbf{x} = x_1, \ldots x_M$ and $\mathbf{y} = y_1, \ldots y_N$; time indices $t_x$ and $t_y$
Output: Minimum jump cost $c_{\text{min}}$; updated $t_x$ and $t_y$
 1: $c_{\text{min}} \leftarrow \infty$
 2: $\Delta, \Delta_{\text{min}} \leftarrow 0$
 3: while $t_y + \Delta \leq N$ do
 4:       $c \leftarrow (\phi\Delta)^2$
 5:       if $c \geq c_{\text{min}}$ then
 6:             if $t_y + \Delta > t_x$ then
 7:                   break
 8:             end if
 9:       else
10:             $c \leftarrow c + (x_{t_x} - y_{t_y+\Delta})^2$
11:             if $c < c_{\text{min}}$ then
12:                   $c_{\text{min}} \leftarrow c$
13:                   $\Delta_{\text{min}} \leftarrow \Delta$
14:             end if
15:       end if
16:       $\Delta \leftarrow \Delta + 1$
17: end while
18: $t_x \leftarrow t_x + 1$
19: $t_y \leftarrow t_y + \Delta_{\text{min}} + 1$
20: return $c_{\text{min}}$
```

this approach. First, the error of the 1NN classifier critically depends on the dissimilarity measure used. Second, the 1NN classifier is parameter-free and easy to implement. Third, there are theoretical results relating the error of a 1NN classifier to errors obtained with other classification schemes. Fourth, some works suggest that the best results for time series classification come from simple nearest neighbor methods. We refer to [20] and references therein for more details about these aspects.

To asses a classifier's error, out-of-sample validation needs to be done. In our experiments we follow a two-fold cross-validation scheme [16] with balanced data sets (same number of items per class). We repeat the validation 10 times and report average error ratios. To assess the statistical significance of the difference between two error ratios we employ the Friedman's test [5], a non-parametric two-way analysis of variance that deals with dependent samples. We use $p < 0.05$ and apply the Bonferroni adjustment to compensate for multiple experiments [16]. Therefore, using k folds, r repetitions, and s data sets, the actual p^*-value corresponds to $p^* < 1 - \sqrt[krs]{1-p}$. Hence, with our setting, $p^* < 7.124 \cdot 10^{-5}$.

We perform experiments with 36 different time series data sets from the UCR time series repository [11]. This is the world's biggest time series repository, and some authors estimate that it makes up to more than 90% of all publicly-available, labeled data sets [20]. It comprises synthetic, as well as real-world data sets, and also includes one-dimensional time series extracted from two-dimensional shapes [11]. The 36 data sets considered here practically correspond to the totality of the UCR repository. Only 4 data sets were discarded prior to and independently from the present work. Within the 36 data sets, the number of classes ranges from 2 to 50, the number of time series per data set ranges from 56 to 9,236, and time series lengths go from 24 to 1,882 samples (a total of 728,611,296 samples from 51,888 time series have been processed). For further details on these data sets we refer to the cited references.

Before performing the experiments, all time series from all data sets were Z-normalized so that they had zero mean and unit variance. Furthermore, in the training phase of our cross-validation we performed an in-sample optimization of the measures' parameters. This optimization step consisted of a grid search within a suitable range of parameter values. For DTW we used 30 linearly-spaced integer values of $w \in [0, 0.25N]$ plus $w = N$ (the unconstrained DTW variant). For MJC we used 30 linearly-spaced real values of $\beta \in [0, 25]$ plus $\beta = 10^{10}$ (in practice corresponding to the squared Euclidean distance variant of $\beta \rightarrow \infty$). After the grid search, the parameter value yielding to the best in-sample error ratio was kept for out-of-sample testing.

5 Results

A full account of the error ratios obtained with the Euclidean distance, DTW, and MJC for the 36 data sets is provided in Table 1. A baseline consisting of using a random dissimilarity measure is also reported. For that we draw a random number from the uniform distribution between 0 and 1 and return this number as the actual dissimilarity between two time series. The rest of the procedure is the same as for the other dissimilarity measures tested.

First we compare the error ratios of the Euclidean and DTW dissimilarity measures (Fig. 2). We observe that, with the considered data, DTW is usually superior to the Euclidean distance. In 15 of the 36 data sets the error ratios obtained for DTW are statistically significantly below the ones obtained for the

Table 1. Average error ratios for the 36 data sets used in the paper. The * symbol indicates that the dissimilarity measure is statistically significantly superior to all others (see text). Best results are highlighted in bold, independently of their statistical significance.

Data set	Random	Euclidean	DTW	MJC
50words	0.980	0.528	**0.361**	**0.361**
Adiac	0.972	0.374	0.378	**0.365**
Beef	0.802	0.487	0.495	**0.458**
CBF	0.670	0.018	**0.001**	**0.001**
ChlorineConcentration	0.667	0.116	**0.111**	0.115
CincECGTorso	0.750	0.003	**0.000***	0.003
Coffee	0.498	**0.017**	0.036	0.054
DiatomSizeReduction	0.749	0.014	**0.009**	0.010
ECG200	0.496	**0.126**	0.133	0.138
ECGFiveDays	0.501	0.012	0.012	**0.001***
FaceFour	0.748	0.155	0.085	**0.041**
FacesUCR	0.929	0.171	0.069	**0.044***
Fish	0.851	0.194	0.196	**0.119***
GunPoint	0.505	0.079	0.027	**0.013**
Haptics	0.796	0.615	0.584	**0.569**
InlineSkate	0.856	0.558	0.521	**0.432***
ItalyPowerDemand	0.498	**0.035**	**0.035**	0.039
Lighting2	0.508	0.322	**0.189***	0.284
Lighting7	0.850	0.428	**0.260**	0.345
MALLAT	0.873	0.021	**0.017**	0.018
MedicalImages	0.902	0.350	**0.276**	0.325
Motes	0.499	0.090	0.068	**0.039***
OliveOil	0.744	0.129	**0.125**	0.147
OSULeaf	0.840	0.434	0.395	**0.296***
SonyAIBORobotSurface	0.501	0.024	0.023	**0.019**
SonyAIBORobotSurfaceII	0.507	0.027	0.026	**0.019**
StarLightCurves	0.666	0.126	**0.117**	0.120
SwedishLeaf	0.933	0.220	0.157	**0.124***
Symbols	0.833	0.038	**0.020**	0.021
SyntheticControl	0.834	0.099	**0.010***	0.035
Trace	0.748	0.210	**0.001***	0.055
Two-Patterns	0.749	0.030	**0.000***	0.001
TwoLeadECG	0.498	0.007	**0.001**	0.003
Wafer	0.504	**0.004**	**0.004**	0.005
WordsSynonyms	0.960	0.535	0.379	**0.355**
Yoga	0.496	0.082	0.071	**0.061***

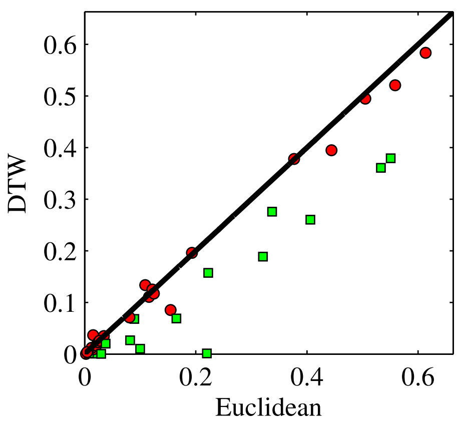


Fig. 2. Pairwise comparison between Euclidean and DTW dissimilarity measures. Values in the lower-right triangular part indicate better results for DTW (better results for Euclidean distance would be scattered in the upper-left triangular part). Green squares indicate statistically significant differences in error ratios (non-significant ratios are denoted with red dots).

Euclidean distance. Notice though that for a few data sets the Euclidean distance is slightly but not statistically significantly superior to DTW. This is due to the fact that the optimization step fails to learn a better w parameter value, which for these specific cases would have been $w = 0$.

We now turn our attention to the proposed measure based on MJCs (Fig. 3). When comparing it with the Euclidean distance (Fig. 3 left) we find that MJC is usually superior. In fact, we have an equivalent situation as we had when comparing DTW and the Euclidean distance. In 17 of the 36 data sets the error ratios obtained for MJC are statistically significantly below the ones obtained for the Euclidean distance. Again, for the very same reason outlined before, the Euclidean distance is slightly but not statistically significantly superior to MJC in a few data sets.

The interesting comparison though is between DTW and MJC (Table 1 and Fig. 3 right). At a first sight, their error ratios look very similar. DTW's error ratio is lower than MJC's in 16 of the 36 data sets and MJC's is lower than DTW's in also 16 of the 36 data sets. However, if we just focus on statistically significant results, MJC outperforms DTW in 8 of the 36 cases while DTW only outperforms MJC in 5 of the 36 cases. This points towards a slight superiority of MJC with respect to DTW.

From the error ratios reported with the considered data sets we see that the Euclidean distance is never statistically significantly superior to DTW nor to MJC (which is a clear consequence of the fact that both DTW and MJC incorporate the Euclidean distance as a special case of their parameters value). This reduces the comparison to DTW and MJC. Therefore, summarizing, we see that MJC outperforms DTW in 8 of the 36 data sets (≈22%), that DTW

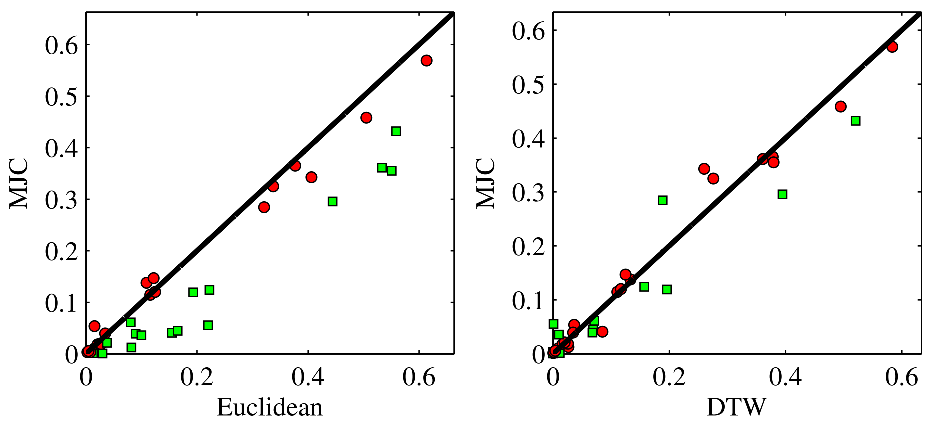


Fig. 3. Pairwise comparison between Euclidean and MJC dissimilarities (left) and between DTW and MJC (right)

outperforms MJC in 5 of the 36 data sets (≈14%), and that for the remaining 23 data sets (≈64%) the error ratios are comparable within statistical significance. The fact that MJC outperforms DTW for roughly 22% of the considered data sets highlights the potential of the former and has clear implications for researchers and practitioners dealing with new data, as such new data set could potentially be one of the data sets where MJC statistically significantly outperforms the classical DTW.

6 Conclusion and Discussion

We have presented a new approach to assess dissimilarities between time series based on minimum jump costs (MJC). Beyond the novelty of the concept, we have shown that it is computationally easy to implement (just a few lines of code) and that further efficiency issues can be deployed. More importantly, we have shown that the MJC dissimilarity measure is very competitive. Under rigorous and extensive experiments we find that, in many situations, it can statistically significantly outperform dynamic time warping, a dissimilarity measure which is regarded as very difficult to beat. All these facts encourage the incorporation of the MJC dissimilarity measure to the standard off-the-shelf toolkit for retrieving and classifying time series data.

Intuitively, it seems clear that by deriving a data-specific dissimilarity measure targeted to a particular problem one would always outperform generic measures such as the Euclidean, DTW, or MJC. However, this does not preclude considering these generic measures as part of an initial approach or assessment. Furthermore, it could also well be the case that, for such a specific data set, the derived, data-specific measure was comparable to or even less competitive than the three measures considered here. In these cases, the usage of a potentially complex data-specific dissimilarity measure could be difficult to justify.

We also notice that all considered time series dissimilarity measures are 'global' dissimilarity measures, i.e. they match whole time series. These are the big majority of time series dissimilarity measures. However, measures considering 'local' or subsequence matches and their variants do also exist (see e.g. [18,19]). Given a data set that needs of such local matches, a common operation to still use global dissimilarity measures is to partition the whole time series into multiple subsequences, either by exploiting some previous knowledge of the data or simply by a brute-force moving window strategy. This partitioning increases the number of comparisons between (sub)series and sometimes implies a further operation to merge the result of such comparisons (e.g. by taking the mean or the maximum similarity found [18,19]).

In this contribution we do not specifically treat the case of multidimensional time series. Indeed, all the considered data sets from the UCR time series repository are uni-dimensional. Nonetheless, one should notice that the multidimensional case can be easily handled. One option could be to consider each component or dimension as a single time series, calculate its dissimilarity, and finally aggregate all such dissimilarities to form a global measure (potentially weighting individual dissimilarities). However, one should notice that the formulation of both DTW and MJC naturally incorporates the possibility to deal with multidimensional time series, since they both use a sample (local) dissimilarity function $\delta()$ (Eqs. 2 and 6), which may be problem-specific and adapted to the particular nature of the considered time series.

The number of CBR systems that deal with time series data is increasing in domains such as health care or industrial monitoring. Although an important issue is the selection of the appropriate sample dissimilarity function $\delta()$, the availability of powerful, general-purpose measures for comparing time series is required to speed-up the development of these CBR systems. In this research, MJC has been evaluated in 36 data sets with time series of lengths ranging from 24 to 1,882 samples. Although for some of the considered data sets the time series are relatively long, they generally model a unique complex pattern. Thus, we believe that the identification of key sub-sequences such as in [17,22] or the dimensionality reduction applied in [13] would not improve classification performance in these data sets. Contrastingly, generic dissimilarity measures such as the ones considered here can be very useful in data sets where, after recurrent patterns have been identified, the resulting sub-sequences still need to be compared.

Acknowledgments. This research has been partially funded by 2009-SGR-1434 from Generalitat de Catalunya, TIN2009-13692-C03-01 from the Spanish Government, and EU Feder funds. JS also acknowledges JAEDOC069/2010 from Consejo Superior de Investigaciones Científicas.

References

1. Bottrighi, A., Leonardi, G., Montani, S., Portinale, L., Terenziani, P.: Intelligent Data Interpretation and Case Base Exploration through Temporal Abstractions. In: Bichindaritz, I., Montani, S. (eds.) ICCBR 2010. LNCS, vol. 6176, pp. 36–50. Springer, Heidelberg (2010)
2. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases. In: Proc. of the ACM SIGMOD Int. Conf. on Management of Data, pp. 419–429 (1994)
3. Funk, P., Xiong, N.: Case-based reasoning and knowledge discovery in medical applications with time series. Computational Intelligence 22(3/4), 238–253 (2006)
4. Han, J., Kamber, M.: Data mining: concepts and techniques. Morgan Kaufmann, Waltham (2005)
5. Hollander, M., Wolfe, D.A.: Nonparametric statistical methods, 2nd edn. Wiley, New York (1999)
6. Kanawati, R., Malek, M., Salotti, S. (eds.): Proc. of the 1st Workshop on Applying CBR to Time Series Prediction (ICCBR-2003). Dept. of Computer and Information Science, Northwestern University of Science and Technology (2003)
7. Kanawati, R., Salotti, S. (eds.): Proc. of the 2nd Workshop on Applying CBR to Time Series Prediction (ECCBR 2004). Dept. of Sistemas Informaticos y Programación. Universidad Complutense de Madrid (2004)
8. Kantz, H., Schreiber, T.: Nonlinear time series analysis. Cambridge University Press, Cambridge (2004)
9. Keogh, E.: Machine learning in time series databases (and everything is a time series!). Tutorial at the AAAI Int. Conf. on Artificial Intelligence (2011)
10. Keogh, E., Kasetty, S.: On the need for time series data mining benchmarks: a survey and empirical demonstration. Data Mining and Knowledge Discovery 7(4), 349–371 (2003)
11. Keogh, E., Zhu, Q., Hu, B., Hao, Y., Xi, X., Wei, L., Ratanamahatana, C.A.: The UCR time series classification/clustering homepage (2011), `http://www.cs.ucr.edu/%7eeamonn/time_series_data`
12. Martin, F.J., Plaza, E.: Ceaseless Case-Based Reasoning. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 287–301. Springer, Heidelberg (2004)
13. Montani, S., Portinale, L., Leonardi, G., Bellazzi, R., Bellazzi, R.: Case-based retrieval to support the treatment of end stage renal failure patients. Artificial Intelligence in Medicine 37(1), 31–42 (2006)
14. Rabiner, L.R., Juang, B.: Fundamentals of speech recognition. Prentice-Hall, Upper Saddle River (1993)
15. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. on Acoustics, Speech, and Language Processing 26(1), 43–50 (1978)
16. Salzberg, S.L.: On comparing classifiers: pitfalls to avoid and a recommended approach. Data Mining and Knowledge Discovery 1(3), 317–328 (1997)
17. Serrà, J., Müller, M., Grosche, P., Arcos, J.L.: Unsupervised detection of music boundaries by time series structure features. In: Proc. of the AAAI Int. Conf. on Artificial Intelligence (in press, 2012)
18. Serrà, J., Serra, X., Andrzejak, R.G.: Cross recurrence quantification for cover song identification. New Journal of Physics 11(9), 093017 (2009)

19. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. Journal of Molecular Biology 147, 195–197 (1981)
20. Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., Keogh, E.: Experimental comparison of representation methods and distance measures for time series data. Data Mining and Knowledge Discovery (in press, 2012), `http://dx.doi.org/10.1007/s10618-012-0250-5`
21. Xia, B.B.: Similarity search in time series data sets. MSc thesis, Simon Fraser University, Burnaby, Canada (1997)
22. Xiong, N., Funk, P.: Concise case indexing of time series in health care by means of key sequence discovery. Applied Intelligence 28(3), 247–260 (2008)
23. Zehraoui, F., Kanawati, R., Salotti, S.: CASEP2: Hybrid Case-Based Reasoning System for Sequence Processing. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 449–463. Springer, Heidelberg (2004)

Case-Based Reasoning Applied to Textile Industry Processes

Beatriz Sevilla Villanueva and Miquel Sànchez-Marrè

Knowledge Engineering Machine Learning Group, Universitat Politècnica de Catalunya-BarcelonaTech, Barcelona, Spain
{bsevilla,miquel}@lsi.upc.edu

Abstract. This paper describes an innovative usage of Case-Based Reasoning to reduce the high cost derived from correctly setting the textile machinery within the framework of European MODSIMTex project.

Furthermore, this system is capable of dealing with flexible queries, allowing to relax or restrict the searches in the case base. The paper discusses the ideas, design, implementation and an experimental evaluation of the CBR applied to the spinning scenario included in the project framework.

Keywords: Case-Based Reasoning, Textile Industry, Filter-based Retrieval, Similarity-Based Retrieval.

1 Introduction

The textile industry faces important challenges regarding the design of multifunctional textile products, because of the enormous difficulty to relate the design/processing parameters of the component materials with the quality parameters of the resulting textile structure. It is often impossible to define the characteristics and attributes of a given textile structure due to the difficulty (and sometimes impossibility) of measuring these parameters (parameters like flexibility and compressibility of some kind of fabrics). Furthermore, product performance failures are unacceptable because there is a clear risk of functionality loss, and in consequence, sometimes there is danger for human life. Up to now, the typical development practice for multifunctional textiles is to adjust the processing parameters and modifying preliminary products by trial and error, producing samples until the desired quality, design and functionality parameters can be safely achieved during production. This procedure accumulates problems along the manufacturing value chain. For instance, a deviation in the specifications of a yarn can produce a fabric that does not match the required performance or functionality. Moreover, it proves very difficult to match the designers's idea with the final product. Furthermore, this procedure costs a lot of time and effort and therefore makes prototype development a very expensive process with high energy and material wasting. This is especially critical when a company develops new multifunctional technical textiles. In order to overcome this obsolete paradigm, there is a need for the development of new tools for rapid prototyping and production set-up of these multifunctional textile products.

The best way to optimize the configuration of textile structures is to apply the mathematical analytical simulation on the mechanisms that actuate on the internal components of these textiles structures. However, in this case, taking the results of the analytical calculation as the only answer for the problem would be very risky, due to the complexity of the structures to be simulated, and the difficulty of matching the real structure with the virtual mathematical model. There are certain parameters of influence and response that do not have any kind of direct mathematical relationship. On the other hand, although some parameters are only evaluated subjectively, they are very important to quantify the quality of a given textile structure. Therefore, it is important to apply the mathematical analysis in those specific parts of the virtual textile model where it can be applied.

In the textile industry high costs are achieved to produce products. Increased cost of time and resources is due to the incorrect set up of the machinery. These errors derived from adjusting the machinery are quite common. So, the problem to solve it is to find a possible configuration to guide the textile machinery settings. Thus, we are facing to different scenarios. Each new product corresponding to a new process from a concrete scenario does not contain the same parameters neither has the same needs of other products in different scenarios.

1.1 MODSIMTex Project

The MODSIMTex project[1] objective is to build up a system which dramatically reduces the cost of developing new technical textile products by reducing the time, energy and raw material waste during the production machinery setup process.

Figure 1 depicts the general idea of the MODSIMTex system comparing the current process of creating a new product and the advantages of using the MODSIMTex system. The current setup process is a trial and error process. Therefore, a reduction of number of trials would highly improve the general performance.The system will support the product development and production for all products in textile value-added chain that conforms four scenarios: Spinning, Weaving, Knitting and Non woven.

The solution proposed by MODSIMTex is to develop a composed simulation system, where the result is given through 3 methods, which jointly can accomplish precise and suitable results:

– **Mathematical models**: Mathematical models are being developed to simulate the behaviour of the 4 textile structures studied in this project (yarns, woven fabrics, knit fabrics and non-woven fabrics).
– **Finite Elements**: Development of a finite elements simulation system to simulate the physical properties of the textile structures, based on the mathematical models developed for these textile structures.

[1] The MODSIMTex project (*development of a rapid configuration system for textile production machinery based on the physical behaviour simulation of precision textile structures*) is a FP7 EU Research project. Grant Agreement NMP2-SL-2008-214181.

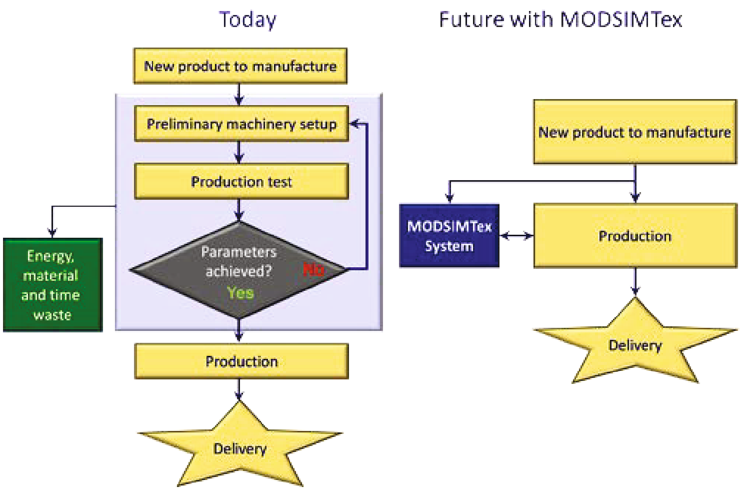


Fig. 1. MODSIMTex problem model

– **Case-Based Reasoning**: Development of an artificial-intelligence based simulation system for the physical properties of textile structures. This CBR system will complement the results of the other simulation systems filling the gaps where the finite elements and mathematical models cannot be applied.

1.2 The Problem Description - Why Applying CBR?

The MODSIMTex system is a tool for helping to adjust the parameters of a textile process. This system starts when a user wants to create a new textile product and he/she asks for a suitable solution satisfying as much as possible his/her requirements on the process parameters. Hence, the idea is to use some AI method to approximate these process parameters, learning from the past executions of this kind of processes.

Since MODSIMTex pretends to be a system implanted in the real textile industry, the database will grow over time as more products are manufactured. Each industry produces different products and each product can have different processes associated. In addition, these processes can change. For instance, when a machine is replaced by a new one, the machine settings are completely different.

All in all, an A.I. method which supports the following properties is needed:

– Incremental: The algorithm should adapt itself along time with the new data.
– Non-dependent of any parameter, since each query can contain different parameters and the relevance of these parameters can change depending on the user requirements.
– Being able to predict more than one parameter.

These characteristics point to a flexible methodology like CBR. The fact that CBR is a lazy learning approach allows that the algorithm is not influenced by any parameter, and can be adapted to any new situation giving more importance to some parameters than others. Besides, the implementation of CBR can be enough open to allow the prediction of more than one parameter. Moreover, textile end users are not interested in a general model which generalize all the data, but a particular solution proposed for the particular query that they are requesting.

1.3 Related Work

The vast majority of the currently available textile design software applications, are nevertheless limited to the visual representation, without any kind of mechanical or physical evaluation of the properties of the textile structures. Hence, these systems lack the ability to assist in the rapid manufacturing process configuration.

In the literature, some publications that deal with the prediction of some specific properties using artificial intelligence methods are found. Notwithstanding, almost all of these approaches seem to be application-specific. In the next list, they are classified by the involved scenario:

- *Spinning:* In Ruzhong et al [18] there is a combination of artificial neural networks with support vector machines (SVM). Lü et al [15] concentrates on SVM, and only on worsted yarn properties. For cotton yarn properties prediction, Van Langenhove et al [14] use learning classifier sets as well as [10]. In [3], a specific application of case-based reasoning for the yarn tenacity prediction has been presented.
- *Knitting:* In Ertugrul et al [7] there is a prediction of the bursting strength of plain knitted cotton fabrics with feed-forward neural networks, and with an adaptive-network-based fuzzy inference system. A prediction of the specific influence of yarn type and fabric structure on the knitted garmented is found in[20].
- *Weaving:* Klöppels and Wulfhorst describe the auto-warp system [12]. The auto-warp system is a simulation-based feedback control approach for the warp tension. ITEMA [2] offers a system with a failure feedback control system that varies the loom speed until a given failure rate is reached. There are as well approaches of loom manufacturers to store the processing settings in central databases.
- *Non-Woven:* There are few publications related to the non-woven and artificial intelligence. Most of them deal with functional property estimation. Vroman et al [21] deals with the estimation of functional properties like: water permeability, filtration level, breaking resistance, elongation at peak and bursting strength. Also, [5] describes a fuzzy model for estimating functional properties as well as structural parameters in function of machine and material parameters.

[2] ITEMA: http://www.itemagroup.com

Another point to take in account is that our system must *support more than one attribute belonging to the Case Solution*, and this classification between description and solution can vary on demand. So, in CBR literature, some hybrid solutions have been found to face up with the reuse step, but most of them are oriented to solve only one solution attribute. Most of the proposals use techniques from supervised learning and have only one attribute to predict [8]. For example, a system to predict oceanographic temperatures [4]. This system retrieves the most similar cases and retrains a radial basis network with them to create a new solution. Another CBR approach that handles more than one attribute as solution is the case completion CBR [2], but the attributes are predefined to belong to the description or the solution part. It handles more than one attribute to solve, but in this case, they solve them step by step. As they are dealing with domains that have predefined and well-known attribute dependencies, the order in the steps is derived from those dependencies. In CBR Recommender literature, the compromise-driven retrieval [16][17] can be found. This concept allows to extend the query specification including rules for numerical attributes: *Less-is-Better* and *More-is-Better*.

2 The CBR Approach in MODSIMTex

In general, for all the project scenarios, the problem can be represented as in figure 2 which is the CBR role: given a new query that has some attribute values defined and other not, the CBR returns an estimation for all the attribute values of the query.

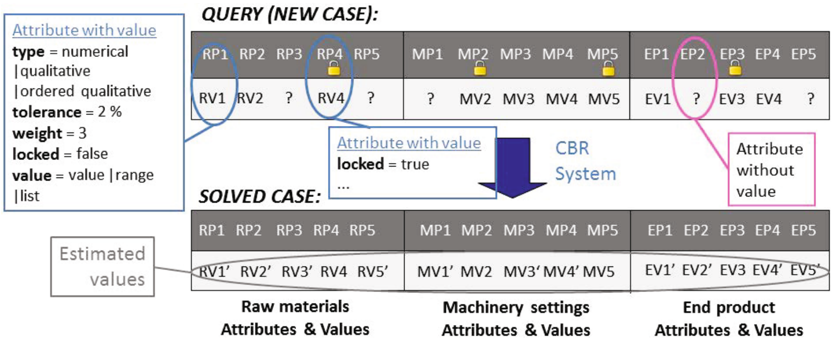


Fig. 2. MODSIMTex problem model

Among these defined values, there are some of them which must be satisfied by the solution case. In order to implement that, their attributes could be marked as locked or not (see the locked symbol in Fig. 2). As it will be explained later, this feature allows to implement a Filter-based retrieval, as a first step in the CBR system. After the cases are filtered by these locked attribute values, then the similarity-based retrieval is applied.

The CBR approach is a module of the whole MODSIMTex system. This system launches the CBR module when it is necessary and decides what to do with the results, and which results will be selected from all the modules. The MODSIMTex schema is described in section 3. The CBR architecture is detailed in the figure 3. In this figure, there is represented how the *Calculation Manager* interacts with the CBR system. The communication between both systems is by means of an *XML* file, that it will be named from now on *communication file*. This communication file contains all the information concerning the new query and the parameters to tune the CBR. Besides, both systems share the same database.

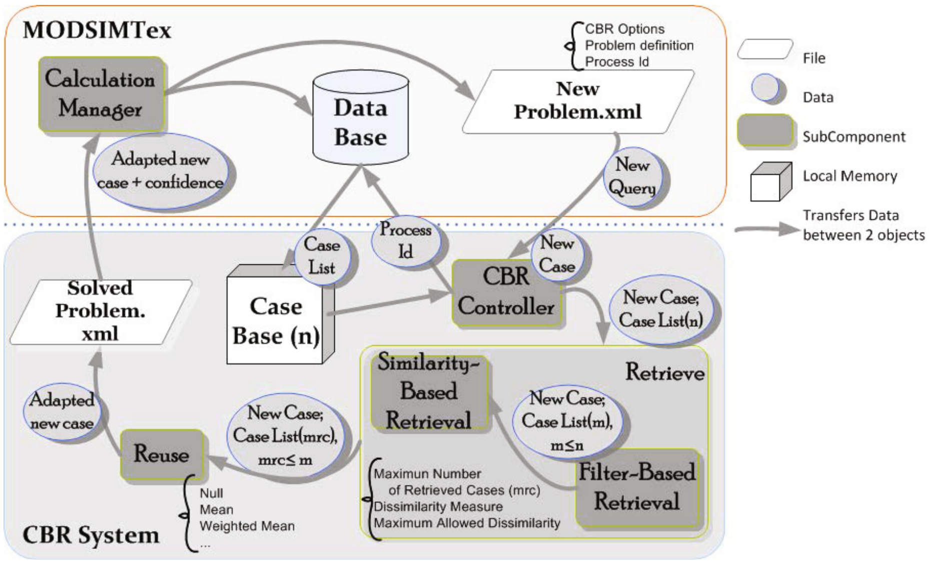


Fig. 3. CBR framework architecture schema

The next subsections start defining how is modeled the MODSIMTex data: case, query and case base structure. Afterwards, there are the descriptions of the four phases: Retrieve, Reuse, Revise and Retain.

2.1 Case Structure

A case is the representation of the MODSIMTex process. A textile process involves machinery and materials: an input material *raw material* and output *end product* (see Fig.2). These elements can be described by parameters or attributes. Consequently, the process can be described as list of attribute-value list.

By default, it is not defined which attributes are part of the description or the solution. This classification is made when a new case or query is defined along with the rest of information. Therefore, the solution for the textile product system cannot be a static structure. This fact makes difficult to use a CBR approach. Moreover, these attributes or parameters are heterogeneous in the sense that not all are of the same type (nature). The system supports numerical, non-ordered qualitative and ordered qualitative attributes.

The knowledge that can be added is limited by this flat case structure. However, for each attribute some context information can be defined to characterize it, as for example, to know its nature. One of the simplest ways to add extra knowledge is by including weights to the attributes (see next section). Numerical attributes are the most common ones in MODSIMTex domain. When dealing with them, we realized that there were two different behaviours. The first behaviour is sensitive to small changes in a close range of values. The second one considers that if two values are close enough can be treated as equals. For the attributes with this last behaviour, a *tolerance* or *goodness* can be defined, delimiting how close two values should be for being considered as equals. Thus, these behaviours affect the local similarity assessment.

2.2 Query Structure

The query is described into the communication file as it is shown in figure 3. For each attribute, an extra information is defined. This information affects only the current new query. In every new query, some attributes have values and others do not (see the figure 2). Actually, the fact that some attributes have a value determines whether the attribute belongs to the solution part or the description part. As it was mentioned before, the importance of an attribute is defined at this moment, because depending on the user preferences this importance can vary from one query to another even when both belong to the same process.

Our approach introduces some options to guide the search of the closest - most similar cases. These options give the user the possibility to restrict or relax the search and to change what is the description and solution of a case.

Usually, CBR systems attempts to define a new query as a case with the same structure defined in the case base but without value/s in the solution part. To relax this concept, the first what it is introduced is the possibility to define which are the attributes belonging to the description or solution in the same query. Second, the concept of value is extended depending on the attribute type. For numerical attributes is possible to specify a single value or a range, and for qualitative attributes, a single modality or a list of them.

Furthermore, each attribute in the description part could be *locked* indicating that the cases to be retrieved must have "equal" values as the ones in the query. In that case, the system filters the cases with these values. Hence, this *locked* parameter defines which are the attributes used in the filter-based retrieval and those that will be used for similarity-based retrieval (see section 2.4). Therefore, there are two classifications of the attributes: depending on its *nature* and on its *behaviour*. The next descriptive attribute classification is given from the behaviour's point of view:

– locked: the retrieved cases must show an *equal* value to the query value in this attribute . Regarding that the *equality* concept depends on whether it

has a defined tolerance (see previous section 2.1) or the query value is single value or a range or a list.
– non-locked: the values of the retrieved cases might be equal or not. Depending on the attribute weight, the search will take more into account that the values of the retrieved cases are closer to the query value.

2.3 Case Base

The Case Base is composed by data coming from the MODSIMTex Database. The case base is dynamically generated from the MODSIMTex database given a concrete process (specified in the communication file). Data consists on past executions of this process. Consequently, for each new query of a different process, the case base has to be reloaded with different cases (see Fig.3).

The case base is flat. Non indexation is suitable because depending on each process and query, some attributes could be in the solution or the description part, and the relevance of the attributes could change as well. Moreover, the case base is a selection of the cases belonging to the same process. So, it is not feasible in terms of time and memory to create one structure for each process of each scenario (spinning, knitting, weaving and non-woven).

2.4 Retrieve

The retrieval task is split into two steps as it is depicted in figure 3. First, the cases are filtered based on the locked attributes and secondly, the most similar cases, based on the distance measure and the attribute relevance, are selected.

The idea is to find a compromise between filter-based retrieval [1] and similarity-based retrieval. A filter-base retrieval is a restrictive search because everything which is defined as mandatory has to be satisfied. So, this kind of search is not flexible and usually ends without any result. This compromise is defined by the feature *locked* determining which are the attributes for using filter-based retrieval and those for using similarity-based retrieval.

The main tunning parameters related to the retrieval task which can be set by the end user are:

– *The Maximum Number of Retrieved Cases.* The list of the most similar cases is tailed by this number.
– *The Maximum Allowed Dissimilarity between the query case and the retrieved cases.* All the cases that are further from this threshold are automatically dismissed, even though there were not enough cases to return.
– *The dissimilarity measure.* The selection of the distance metric with its own parameters.

Therefore, a distance measure is used to calculate the dissimilarity. At this point, it is important to remark that the results will vary depending on the chosen distance and also, on the weights defined for each attribute. The offered distance measures are global and these globals functions use a local distance that copes with the different attribute types, ranges, list and the tolerance concept. The equation 1 computes the local distance for numerical attributes. The local distance of qualitative attributes are simpler because it only has to check if the values are equals or if the case value belongs to query list.

$$localDistance(q_i, c_i) = \begin{cases} 0 \quad \text{if } q_i \text{ is a value and } c_i \in [q_i - t, q_i + t] \\ 0 \quad \text{if } q_i \text{ is a range and } c_i \in [q_i.lower - t, q_i.upper + t] \\ min(|c_i - (q_i - t)|, |c_i - (q_i + t)|) \ \text{ if } q_i \text{ is value with tolerance } t \\ min(|c_i - (q_i.lower - t)|, |c_i - (q_i.upper + t)|) \text{ if } q_i \text{ is a range with tolerance } t \end{cases} \tag{1}$$

Most of these global measures have parameters to be adjusted and also, support attribute weights. The end user can select the global distance functions among the following:

- *L'Eixample* [19] is a heterogeneous distance sensitive to weights. Its most important parameter is a threshold, *alpha*, that defines the boundary for using quantitative or qualitative values for numerical attributes.
- The Minkowski metric is probably one of the most commonly used. Its most important parameter is the *p-norm* that defines the $L_p$space (1 = Manhattan; 2 = Euclidean). This approach allows to use attribute weights. In fact, with r=2 becomes an approximation of HOEM (heterogeneous overlap euclidean metric)[9].
- Canberra [13] and Clark distances [6], are normalized distances sensitive to small changes close to the origin. In [11] is shown that they both have a good performance on ranked data.
- Weighted Geometric Difference Mean is the weighted geometric mean of the local distance among all the attributes. This average is not constant and lower than the arithmetical one and penalize more those attributes that are further from the mean local distance.

Finally, this phase returns a list of the most similar cases that are not further than the *maximum allowed dissimilarity* and the list is not longer than the *maximum number of retrieved cases.*

2.5 Reuse

Reuse is, inherently, one of the most complex tasks in the CBR cycle, since it requires a deep understanding of the represented situation in the retrieved case/s and of the domain knowledge. When there is not sufficient information about one attribute, it seems a good solution to use a predicting algorithm, such as a neural network, support vector machine, or a decision tree. Notwithstanding, the problem is that the solution should have one or more models for every attribute that has not a value, besides that the solution attributes are changing in each query. Thus, it is not viable to pre-calculate all the possible models. And to train these models has an extra temporal cost. On the other hand, normally these methods are trained and thereafter used for predicting, but in this case, the data base is small, so the training part could be not enough to train these algorithms. In addition, the system copes with all the processes from all the scenarios in MODSIMTex. These are the reasons to face up the reuse step with generic methods.

The included reuse strategies are the following:

– **Null**: The case with the lowest distance is selected, then the small differences are abstracted away and they are considered as non-relevant. So, the solution of the retrieved cases is directly transferred to the new case as its solution.
– **Mean**: When having more than one retrieved case, then the mean (for numerical and ordered qualitative attributes, which have been transformed into a numerical scale) or the mode (for non-ordered qualitative attributes) is computed.
– **Optimum selection**: The list of the retrieved cases is ranked and then the solution of the first case is transferred. The idea is to obtain the case that optimize this attribute which could represent a quality or cost function. In order to rank the cases, it is also possible to combine the distance with the given attribute.
– **Weighted Mean**: A weighted mean or mode is computed. In this case the weights could be the own distance (smaller distance implies bigger weight), or a user-given attribute or a combination of both. It is the same idea of the *Optimum Selection* but using more than one retrieved case.

Like the rest of CBR configuration, the method to be selected and the optional attribute to weight the cases are included in the *communication file*.

2.6 Revise and Retain

The end user takes into account the proposed solution offered by the MODSIMTex system and he/she makes his/her own decision on how setup the machinery for making the new product. Thus, the real settings used by the end user are the experience which is stored in the MODSIMTex database.

The MODSIMTex system choose the proposed solution for each attribute among the module's solutions. To support this decision, the CBR module provides a *confidence level* for each attribute solution. This confidence is based on the variability of the solutions of the cases that have been used for the reuse step, and it is penalized by the missing values that they contains. Besides this evaluation, the MODSIMTex system shows the proposed solution to the end user but do not offer any feedback mechanism.

The CBR system has only access to the data for reading, reminding that the database is the compilation of the past real execution records. Therefore, in the retain phase, CBR system only has to write the results in the *communication file*. These results include: the predicted solution with the respective confidence levels and the list of the cases used in the reuse step.

3 Design and Implementation of the CBR Module in MODSIMTex

The MODSIMTex architecture is depicted in figure 4. The MODSIMTex system is a plugin of an existing software called *Retrieval System*. The CBR module should attend all requests from the MODSIMTex system, as a part of it. The communication among the modules is done through the *XML* file (*communication file*) as it was mentioned before. This file is responsible of transferring all the information between the different modules and the *calculation manager*. The calculation manager is in charge to decide the order of the module executions and to select the best results to be shown in the interface.

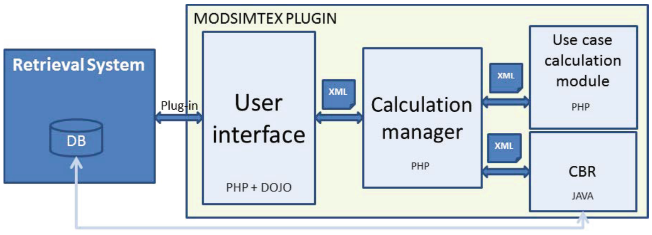


Fig. 4. MODSIMTex integration: Modules which compose the MODSIMTex system

The CBR module is required to be an independent implementation. This module is developed in Java 6 and the libraries used for the MODSIMTex data are: SAX for parsing the XML files and JDBC for the connection with the database.

4 An Application: Spinning Scenario

The CBR module has been developed and successfully tested with real data coming from different scenarios and partners following the workplan of the project. Theses tests has been carry out by the textile experts of every industry. They are asking the system for different queries and evaluate whether there are good results or not.

For testing which is the most suitable configuration and having a general idea of how is the CBR performance, a battery of CBR executions has been carried out in the spinning scenario. For this testing, the spinning experts suggested to try two *template cases*. In both cases, the process involves a raw material (fiber), machinery and an end product (yarn). The first template case1 is focused on knowing which are the machine setting for the given raw material and end product properties and the second (case2), it is focused on predicting which are the properties of the end product for the given machine settings and raw material properties.

Table 1. Template Cases Suggested by the Spinning Expert with Statistical Results

Query Specification					Statistical Results (%)		
Case	Input Attributes	Type	Tolerance(%)	Weight	NMSE	CV	Success
Case1	Fiber Fineness	Num.	3	5	0,92	0,0063	-
	Staple Fiber Length	Num.	3	5	1,2	0,005	-
	Yarn Fineness	Num.	3	4	2,98	0,035	-
	Yarn Strength	Num.	8	4	3,27	0,056	-
	Break Elongation	Num.	5	4	1,91	0,040	-
	CVm Uster	Num.	3	4	1,36	0,028	-
	Hairness Uster	Num.	3	4	3	0,040	-
Case2	Fiber Fineness	Num.	3	5	7,66	0,04	-
	Staple Fiber Length	Num.	3	5	12,29	0,04	-
	Twist Coefficient	Num.	3	5	8,78	0,05	-
	Rotor Speed	Num.	3	4	8,75	0,06	-
	Disgregator Speed	Num.	-	3	21,87	0,12	-
	Rotor Type	Qualit.	-	4	-	-	63
	Nozzle Type	Qualit.	-	3	-	-	59,3
	T.S. Type	Qualit.	-	2	-	-	65,5

In table 1, for each template case, the attributes shown are those filled with values (case description). Each case belongs to a different process as a percentage. Thus, the case bases are different. In these templates, the spinning experts setup the weights (ranging from 0 to 5) and the tolerance values. The first case base contains 264 non repeated cases and one repeated case and 49 attributes. The data contains around 30% of missing values. The second case base contains 264 non repeated cases, 253 attribute and 61% of missing values.

The evaluation of the CBR results for both spinning template cases has been carried out through a similar process to the leave-one-out validation algorithm. The procedure starts deleting all the cases that are repeated. Then for each case$_i$ (i = 1..264) in the case base until all are solved:

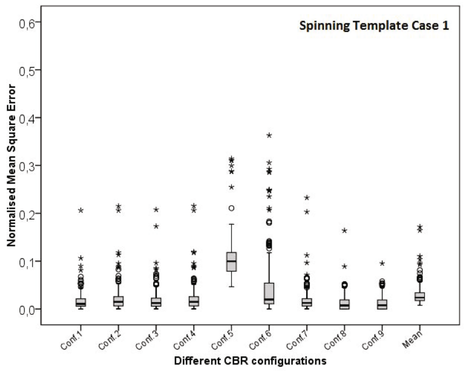
1. The case$_i$ is removed from the case base.
2. The query input attribute values are replaced by the values in the case$_i$.
3. This new query is solved and the proposed solution is compared with the real solution (case$_i$).
4. The case$_i$ is introduced again in the case base.

This experimental evaluation has been performed using several combinations of the CBR configurations. Most of them led to similar error results. In table 1 are shown the normalized mean square error (NMSE) and the coefficient of variance (CV) for each numerical attribute. The percentage of success (Success) for qualitative attributes. These measures are the average of all NMSE, CV and Success obtained in all the tested configurations. From both experiments (*case1* and *case2*) can be concluded that the error ratio is small in average. The different behaviour is due to the fact that in the template case1 there was 30% of missing values against the 61% in case2. Even though, the high percentage of missing values, the performance of the CBR system is good.

In figure 5 all the different configurations tested are shown. These configurations are the following:

Configuration	Distance	Reuse Algorithm	N.Ret.Cases
Conf.1	Weighted Euclidean	Null	1
Conf.2	Weighted Euclidean	Weighted Mean	5
Conf.3	Weighted Euclidean	Weighted Mean	3
Conf.4	Weighted Euclidean	Mean	1
Conf.5	Weighted Geometric	Weighted Mean	5
Conf.6	Weighted Geometric	Null	1
Conf.7	Weighted Manhattan	Weighted Mean	3
Conf.8	Canberra	Weighted Mean	3
Conf.9	Canberra	Null	1
Mean	Mean of all the configurations		

The graphics in Fig.5 depict the boxplot distribution of the normalized mean square error (NMSE) for each query case against the spinning Case Base and for each of the tested configurations plus the mean of all them. This NMSE has been calculated by comparing the proposed case against the original one. In the template case1, the average of NMSE for all the query cases and all the configurations tested is 2,9% with a standard deviation of 2%. This menas an average accuracy of 97,1%, which is very high. In the template case2, the average of NMSE is 12% with a standard deviation of 7%. This means a average of 88%, which is really good.



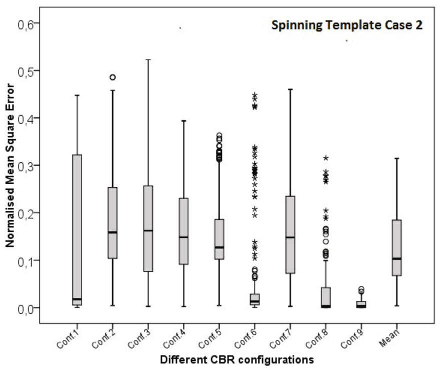


Fig. 5. Boxplot of the different CBR configurations. The y-axis is the distance between the proposed cases and the original case.

From the different configurations results in the template case1, it can be concluded that the worst configurations are using the *weighted geometric* distance (Conf.5 and Conf.6). Also, the *null* reuse strategy seems to be the best approach. In template case2, it seems that again the *null* strategy is a good one (Conf.1, Conf.6, Conf.9). Also the *Canberra* distance has good performance (Conf.8 and conf.9).

From a textile point of view, the textile experts, after analysing the results of several tests (they are testing on their own as well) declared that the obtained results were good approximations to the real values. The textile experts stated that to them was shown the viability of the CBR system to estimate the machine settings and yarn properties by introducing different conditions of the spinning processes.

5 Conclusions and Future Work

The CBR module described in this paper has been experimentally evaluated. From a qualitative point of view and from the textile industry experts who are in charge of validating the system, have concluded that the system is suitable for approximating the machinery settings, as a preliminary evaluation. From the testing it can be concluded that the accuracy of the proposed CBR system is high, and therefore, suitable for practical applications in textile industry.

In the near future a more statistical and numerical evaluation will be done as the other scenarios and components of the MODSIMTex project will be successfully integrated. Then, a deeper evaluation will be possible including all four scenarios: Spinning, Weaving, Knitting and Non-Woven. Moreover, as the design of the core of the CBR module is a very general tool, it can also be applied to other domains where a case can be characterized as an attribute-value list.

The first version of the system showed us the need of a more flexible way to formulate queries. The textile industry experts had the need to define the new queries more vaguely or more restrictive depending on the case. The MODSIMTex interface has been adapted to offer a more flexible way to make the new queries, so the CBR module has been also adapted in order to understand these new queries.

When we were testing, authors realized that besides to the special features used such as ranges, tolerance or locked attributes, there are attributes which the experts want to *minimize* or *maximize*. Up to now, it is possible to rank cases only with one attribute with the *Optimum Selection* or the *Weighted Means* reuse algorithms. Hence, in a new release, it will be possible to set up this behaviour, to *minimize* or *maximize* some conditions. Thus, new features to manage them will be included. These conditions are specially useful to handle attributes representing quality or cost properties.

Acknowledgments. The authors acknowledge the support of the MODSIMTex project. Specially, as spinning experts, Jose Antonio Tornero and Francesc Cano from the Institute of Textile Research and Industrial Cooperation of Terrassa (INTEXTER-UPC).

References

1. Bridge, D.: Product recommendation systems: A new direction. In: Weber, R., von Wangenheim, C.G. (eds.) Proceedings of the Workshop Programme at the Fourth International Conference on Case-Based Reasoning, pp. 79–86 (2001)

2. Burkhard, H.D.: Case completion and similarity in case-based reasoning. Computer Science and Information Systems 1(2), 27–55 (2004)
3. Cheng, Y., Cheng, K.: Case-based reasoning system for predicting yarn tenacity. Textile Research Journal 74, 718–722 (2004)
4. Corchado, J., Lees, B.: A hybrid case-based model for forecasting. Applied Artificial Intelligence 15(2), 105–127 (2001)
5. Deng, X., Vroman, P., Zeng, X., Koehl, L.: A fuzzy criterion for selecting relevant process parameters for the development of nonwoven products. Journal of Information and Computing Science 2(2), 93–102 (2007)
6. Eidenberger, H.: Distance measures for mpeg-7-based retrieval. In: Proceedings of the 5th ACM SIGMM International Workshop on Multimedia Information Retrieval, MIR 2003, pp. 130–137. ACM, New York (2003)
7. Ertugrul, S., Ucar, N.: Predicting bursting strength of cotton plain knitted fabrics using intelligent techniques. Textile Research Journal 70, 845–851 (2000)
8. Fernández-Riverola, F., Corchado, J.M.: Sistemas híbridos neuro-simbólicos: Una revisión. Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial 4(11), 12–26 (2000)
9. Giraud-Carrier, C., Martínez, T.: An efficient metric for heterogeneous inductive learning applications in the attribute-value language. Intelligent Systems, 341–350 (1995)
10. Hunter, L., Fan, J., Spies, A.: Latest developments in measuring cotton properties and relating them to yarn properties. In: 25th International Cotton Conference. Faserinstitut Bremen, Bremen (2000)
11. Jurman, G., Riccadonna, S., Visintainer, R., Furlanello, C.: Canberra distance on ranked lists. In: Proceedings of Advances in Ranking NIPS 2009 Workshop (2009)
12. Klöppels, M., Wulfhorst, B.: Auto-warp - new components of the automation concept for the loom. Fibres and Textiles in Eastern Europe 5, 100–102 (1997)
13. Lance, G.N., Williams, W.T.: Mixed-data classificatory programs i - agglomerative systems. Australian Computer Journal 1(1), 15–20 (1967)
14. Langenhove, L., Sette, S.: Learning classifier systems for modeling the spinning process. In: 24th International Cotton Conference. Faserinstitut Bremen, Bremen (1998)
15. Lü, Z., Yang, J., Xiang, Q., Wang, X.: Support vector machines for predicting worsted yam properties. Indian Journal of Fibre and Textile Research 32, 173–178 (2007)
16. McSherry, D.: Similarity and Compromise. In: Ashley, K.D., Bridge, D.G. (eds.) ICCBR 2003. LNCS, vol. 2689, pp. 1067–1067. Springer, Heidelberg (2003)
17. McSherry, D.: Completeness Criteria for Retrieval in Recommender Systems. In: Roth-Berghofer, T.R., Göker, M.H., Güvenir, H.A. (eds.) ECCBR 2006. LNCS (LNAI), vol. 4106, pp. 9–29. Springer, Heidelberg (2006)
18. Ruzhong, J., Zhijun, L., Jianguo, Y.: Estimating a product quality by support vector machines method. Mechatronics and Automation. In: ICMA 2007, Harbin, pp. 3907–3912 (2007)
19. Sànchez-Marrè, M., Cortés, U., Roda, I.R., Poch, M.: Sustainable case learning for continuous domains. Environmental Modelling and Software 14(5), 349–357 (1999)
20. Semnani, D., Latifi, M., Tehran, M., Pourdeyhimi, B., Merati, A.: Effect of yarn appearance on apparent quality of weft knitted fabric. Journal of the Textile Institute 96, 295–301 (2005)
21. Vroman, P., Koehl, L., Zeng, X., Chen, T.: Designing structural parameters of nonwovens using fuzzy logic and neural networks. International Journal of Computational Intelligence Systems 1(4), 329–339 (2008)

Natural Language Generation through Case-Based Text Modification

Josep Valls and Santiago Ontañón

Computer Science Department
Drexel University
Philadelphia, PA, USA 19104
josep@valls.name, santi@cs.drexel.edu

Abstract. Natural Language Generation (NLG) is one of the longstanding problems in Artificial Intelligence. In this paper, we focus on a subproblem in NLG, namely *surface realization through text modification*: given a source sentence and a desired change, produce a grammatically correct and semantically coherent sentence that implements the desired change. Text modification has many applications within text generation like interactive narrative systems, where stories tailored to specific users are generated by adapting or instantiating a pre-authored story. We present a case-based approach where cases correspond to pairs of sentences implementing specific modifications. We describe our retrieval, adaptation and revise procedures. The main contribution of this paper is an approach to perform case-adaptation in textual domains.

1 Introduction

Natural Language Generation (NLG) is one of the longstanding problems in Artificial Intelligence. In this paper, we focus on a specific subarea in NLG, namely *surface realization*. Surface realization is the final stage in the NLG pipeline [11], and is in charge of generating actual natural language text given a sentence specification, embodying all decisions related to the grammar and morphology of a particular language [6]. We present a case-based approach to the problem of *surface realization through text modification*: given a source sentence and a desired change (e.g. replace a phrase from a sentence by another phrase, change the subject, change the tense of a verb, etc.), produce a grammatically correct sentence that implements the desired change. In the remainder of this paper we will refer to this problem simply as "text modification".

The research reported in this paper was motivated by work in the Riu system [9], an interactive fiction system that generates stories in natural language by combining and modifying pre-authored story snippets. One of the major drawbacks in Riu is the quality of the text presented to the user, due to the difficulty of generating grammatically correct and semantically coherent natural language. The approach presented in this paper can greatly increase the quality of systems like Riu, but it can also be applied to any other system that generates text by instantiating and modifying predefined text or templates.

In order to address the problem of text modification, we present a CBR approach called CeBeTA. With a few notable exceptions such as CR2N [2] and some recent work in jCOLIBRI [10], the vast majority of work in CBR for natural language focuses on retrieval and spell checking. The main contribution of CeBeTA is that we propose not only retrieval, but also adaptation and revision methods for natural language domains. In particular, cases in CeBeTA contain pairs of sentences, one being a modification of the other. When a new problem arrives to CeBeTA requesting to modify a sentence, the system infers which are the modifications that need to be done to the sentence based on retrieved cases, and they are applied to obtain a collection of candidate solutions. CeBeTA then uses an automatic revision module that selects the solution most likely to be grammatically correct and semantically coherent.

The remainder of this paper is structured as follows. Section 2 formally presents the problem of surface realization through text modification. Section 3 presents the CeBeTA system to address this problem. After that, Section 4 describes our empirical evaluation, and Section 5 compares CeBeTA with existing related work. The paper closes with conclusions and future work.

2 Text Modification through CBR

In this article we present a CBR approach to the surface realization phase of the NLG pipeline. Surface realization has been traditionally dealt with the use of annotated templates. In this paper, we present an alternative solution, based on case-based text modification, which doesn't require annotated templates.

In this section we will look at the specific problem we are trying to address, our proposed solution and a system that implements a subset of this proposal.

2.1 Problem Statement

The problem we address in this paper is the natural language generation by modification of given sentences, specifically:

Given:
1. A correct sentence s (for example: "Alice loves pizza").
2. A desired modification, represented as a replacement $p_1 \rightarrow p_2$ (for example "Alice" $\rightarrow$ "The girls').

Find: A new grammatically correct sentence s' where p_1 has been replaced by p_2 (for example "The girls love pizza").

Finding tokens in a string of text and performing replacements is a trivial task for computers. For example, given the sentence "Alice loves Bob", a simple replacement instruction could be stated as: "Bob" $\rightarrow$ "pizza". The result of the desired replacement would be: "Alice loves pizza". However, simple token replacement doesn't always result in grammatically correct sentences. For example if we would like to execute the replacement instruction "Alice" $\rightarrow$ "I', then the

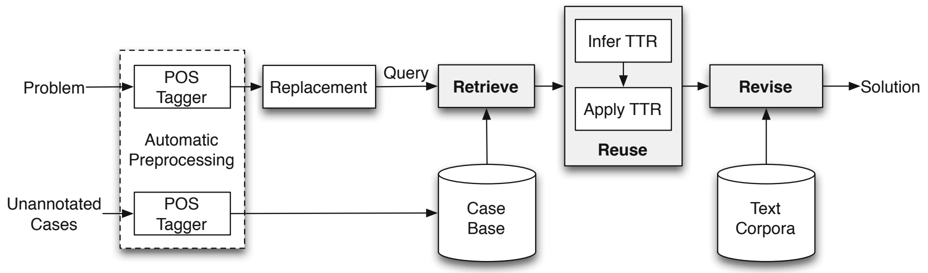


Fig. 1. Overview of the CeBeTA architecture for case-based text modification

output of the replacement would be: "I loves Bob", which is incorrect due to the verb inflexion rules of the English language.

This problem has been addressed in previous work with the use of annotated templates that contain *part-of-speech* (POS) information and some sort of dependency tree or typed dependency list [4] that reflects the relationships between components in phrases and phrase relationships within sentences (i.e. the system needs to know that "Alice" is the subject, singular, female, etc.).

In order to achieve this solution, the surface realizer still needs to understand and process all the annotations properly. Annotated templates have been in use since the early days of NLG and are still in use by modern systems. Hand annotation of templates is a significant bottleneck in NLG systems, for which some automated solutions have been proposed. However, automated annotation is unreliable, due to the ambiguous nature of natural language.

In this paper, we present a CBR approach that can be used to address the problem of surface realization through text modification. Furthermore we propose a model that is domain and language independent and that can be easily extended by feeding domain and language specific corpora along with a very simple grammar modification module.

This problem was motivated by the Riu system [9], a computational analogy-based story generation system, that, as part of its story generation process, produces text by modifying existing natural language sentences. Currently, Riu modifies existing sentences by performing simple replacements, often yielding incorrect sentences. The proposed solution can be applied to range of problems related to text generation by either enriching replacement-based systems like Riu or replacing annotated templates with sets of unannotated documents.

2.2 Conceptual Model

From a CBR point of view, NLG poses major problems in all the stages of the CBR cycle [1]. This section provides an overview of our approach, that we call *Surface Realization through Case-Based Text Modification* (CeBeTA), to deal with the problem of case-based text adaptation. Figure 1 shows all the elements of the CeBeTA architecture, which we describe below.

CeBeTA takes as its input a problem consisting of two parts: a sentence s, and a desired change $p_1 \rightarrow p_2$, where p_1 is a phrase appearing in s. Moreover, we require the problem specification to also include a second sentence s_2 containing p_2; this second sentence is used during the automatic preprocessing to obtain information about p_2 and is then discarded. An example problem is:

– s :“Alice wants to have a pet dog”
– $p_1 \rightarrow p_2$:“Alice” $\rightarrow$ “The girls”
– s_2 :“The girls sing many happy songs”

The previous problem amounts to asking the system to produce a valid sentence after replacing “Alice” by “The girls” from the sentence “Alice wants to have a pet dog”. The expected solution of the system to this problem is “The girls want to have a pet dog”. Moreover, the implemented version of our system allows for the problem to contain more than one desired replacement.

Cases are defined as pairs of grammatically correct sentences (s_s, s_t) where s_t is the result of performing some sort of replacement to s_s and later modifying the result to make sure it is grammatically correct. In the rest of this paper we will call s_s the *source* sentence, and s_t the *target* sentence.

As shown in Figure 1, before the CBR system can actually process either the problem or the cases, they are analyzed using a POS tagger. The tagging process determines the grammatical category of each word (determinant, adjective, etc.) in the context it is found. The second sentence, s_2, included in the problem is used only to obtain the POS tags for p_2, being subsequently discarded.

As shown in Figure 1, retrieval is the first process executed when a problem is received. During case retrieval, a case from the case base is selected by using a similarity measure between problem and cases, as described in Section 3.1. The similarity measure utilises the POS information to find cases which contain similar sentences and similar replacements.

As stated above, one of the main aims of our system is to be able to work from unannotated text, thus, cases in the case base are just pairs of sentences in natural language. Although the cases are automatically POS tagged, there are no additional annotations describing which modifications were performed to the source sentence to produce the target sentence. For that reason, reuse in our system works in two stages: in a first stage, the system tries to infer the transformations that led from the source sentence in the retrieved case to the target sentence in the case (“Infer TTR” in Figure 1). Then, in a second stage the system applies the inferred transformations to the sentence in the problem at hand, in order to produce candidate solutions (“Apply TTR” in Figure 1). These transformations consist of small grammatic operators, that we call *Text Transformation Routines* (TTR) (described in Section 3.2).

Finally, our model includes an automatic revision phase using a *Probabilistic Evaluator* (PE). The PE is a module implementing a language model capable of predicting the likelihood that a sentence is grammatically correct and semantically coherent. The PE ranks the solution candidates and selects the highest ranking candidate as the solution.

3 Technical Approach

This section describes our implementation of the CeBeTA architecture presented in the previous section, detailing how retrieval, reuse and revision work.

3.1 Retrieval

As described above, each case $c = (s_s, s_t)$ in the CeBeTA system consists of a pair of natural language sentences in plain text. During the retrieval process, CeBeTA aims at finding one case that contains a pair of sentences similar to the problem at hand. Moreover, as described above, both cases and problems are automatically analyzed using a POS tagger before being used by the system, so, in the rest of this paper, we will assume that sentences and replacements have been already POS tagged.

To accomplish that task the system performs two steps:

1. Replacement: given the problem at hand, composed of a sentence s_1, and a given replacement $p_1 \rightarrow p_2$, the system generates a new sentence s_r, by directly replacing p_1 by p_2 in s_1, without performing any further changes. The result of preprocessing is the pair (s_1, s_r), which we call the *query*.
2. Nearest neighbor retrieval: after preprocessing, the query now consists of a pair of sentences, like the cases in the case base. Therefore, it is now possible to use a distance metric between pairs of sentences to retrieve the case that is most similar to the query.

The distance metric used in the CeBeTA system between a query (s_1, s_r) and a case (s_s, s_t) is defined as:

$$d((s_1, s_r), (s_s, s_t)) = SED(s_1, s_s) + SED(s_r, s_t)$$

where $SED(s, s')$ is defined as a custom *Sentence Edit Distance* (SED)[12] function. In other words, the distance between the query and a given case is the sum of the distances between the source sentence of the query to the source sentence of the case, and the target sentence of the query to the target sentence of the case. Other forms of aggregating the two distances could be employed by using other t-norms, the choice of the optimal aggregation operator is part of our future work. However, in our experiments, we observed that a simple addition was enough to obtain acceptable results.

The SED distance uses the POS-tagged words as tokens, and, as other edit distance algorithms, computes the shortest possible way in which one sentence could be edited in order to be identical to the other. Each of these edit operations has a different cost. In order to determine the cost of each operation, an undirected weighted graph was built with the different linguistic components of a phrase. The arcs between a particle a and a particle b are estimated according to the impact of replacing a by b, or vice-versa. A subset of the graph used for the experiments reported in this paper is shown in Figure 2.

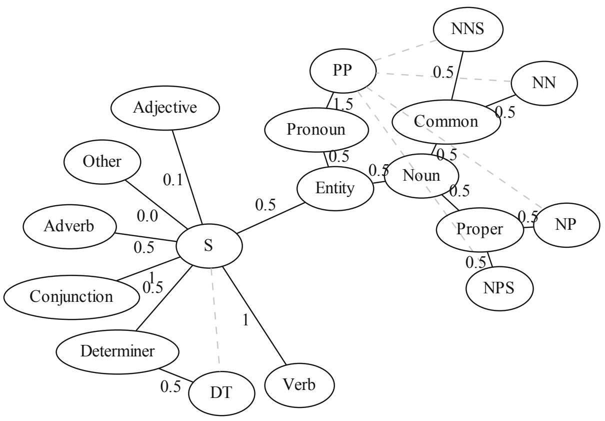


Fig. 2. Subset of the cost graph used in the sentence edit distance used by CeBeTA

The full graph features 54 nodes and 53 arcs plus 5 path overrides (shown in Figure 2 as dashed lines). We estimated the cost of inserting and deleting a word by adding the cost from the POS-class of the word to the root element of the graph (*S*). For example, the cost of deleting a singular proper noun (*NP*) would be $0.5 + 0.5 + 0.5 + 0.5$. And the cost of inserting a plural common noun (*NNS*) would be $0.5 + 0.5 + 0.5 + 0.5$.

For the edit operation, the current system computes an additional cost in order to account for lemma or morphological differences. During POS tagging, words are also *stemmed* (i.e. the stem or lemma of each word is computed).

The edit operation between two words has an additional cost of 0.1 when the lemma does not match. For example, the cost of replacing "dogs" by "dog" is $0.5+0.5+0.0$ but "dogs" by "bird" is then $0.5+0.5+0.1$. For pronoun and noun-pronoun changes, we add 0.05 when the gender does not match and 0.05 when the number does not match; the cost of replacing "girl" by "her" is $1.0+0.1+0.0+0.0$ but "girls" by "him" is then $1.0 + 0.1 + 0.05 + 0.05$. Lemma information could be used to further enhance the distance metric using semantic and conceptual information. Part of our future work would be to enhance the graph, fine-tune arc costs and account for lemma compatibility, for example, using the Regressive Imagery Dictionary (RID) or the Wordnet lexical database.

3.2 Reuse

There are two main ideas behind the reuse process in the CeBeTA system:

- Cases do not contain the adaptation routines used to produce the solutions. CeBeTA infers the adaptation path that lead from the source to the target sentence in a case, to then apply it to the problem at hand (this is motivated by the fact that we want our system to reason from unannotated sentences).
- In order to deal with text modification, we use what we call *Text Transformation Routines* (TTR), which serve as our adaptation operators.

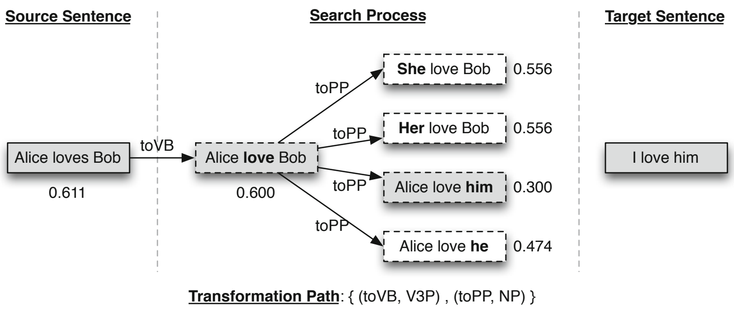


Fig. 3. Illustration of the search process required to obtain the set of TTRs required to reach the target sentence in a case, starting from the source sentence. toVB is the *verb-to-first-person* TTR, toPP is the *noun-to-pronoun* TTR, V3P means "Verb singular 3rd person", and "NP" means proper noun.

A TTR is an operator that given a word, e.g. "loves", and its given POS class, e.g. "verb, 3rd person singular, present tense", generates variations of it, e.g. "loved" or "loving". We used TTRs that enable word elimination, gender transformations; noun, pronoun and determiner transformations, etc. The specific TTRs we used are: *remove-word*, *pluralize*, *singularize*, *make-male*, *make-female*, *insert-determinant*, *verb-to-past-tense*, *verb-to-past-participle-tense*, *verb-to-first-person*, *verb-to-future-tense*, *verb-to-present-tense*, *verb-to-third-person*, *verb-to-gerund*, *verb-to-infinitive*, and *noun-to-pronoun*, with their intuitive effects.

In the rest of this paper we will represent a TTR as a pair $(type, POS)$, where $type$ is the type of TTR (*remove-word*, *noun-to-pronoun*, etc.), and POS is a POS-class (adjective, preposition, verb-3rd-person-singular, etc.), to which the TTR will be applied. Given a POS tagged sentence, we can generate variations of this sentence by applying TTRs to the different words of the sentence.

Given a collection of TTRs, the reuse phase in CeBeTA proceeds in two steps (as shown in Figure 1). In a first step (*Infer TTR*), the system infers which is the set of TTRs that needs to be applied to the source sentence in the retrieved case to produce the target sentence, we call this the *transformation path*. In a second step (*Apply TTR*), candidate solutions to the problem at hand are generated by applying the set of TTRs inferred in the first step. Some TTRs can produce more than one variation, and thus the outcome of the reuse phase is not just a single candidate solution, but a list of candidate solutions to the problem at hand, that will be evaluated by the revise process explained in the next section.

During the *Infer TTR* step, the CeBeTA system uses a hill climbing search method in order to find the sequence of TTRs that can better transform the source sentence in a case into the target sentence, as follows:

1. Set the current sentence to the source sentence in the retrieved case, the current transformation path is empty.
2. Select an applicable TTR to the current sentence, and generate all the possible variations of it using such TTR.

3. Measure the distance between the newly generated variations and the target sentence in the retrieved case.
4. If any of the newly generated variations is more similar to the target than the current sentence, we add the TTR to the transformation path, set that variation as the current sentence, and go back to 2.
5. If there is no applicable TTR that can generate a variation that is more similar to the target sentence than the current sentence, finish, and return the current transformation path.

An example of the execution of this process is shown in Figure 3. The source sentence is "Alice loves Bob", and the target sentence is "I love him". The distance between source and target is 0.611. "Alice loves Bob" is set as the current sentence, and the *verb-to-first-person* TTR is applied to obtain the sentence "Alice love Bob", which is more similar to the target (distance 0.600) than the current sentence, thus, the pair (toVB, V3P) is added to the current transformation path, indicating that the TTR type *verb-to-first-person* was applied to a singular 3rd person verb. Then "Alice love Bob" is selected to be the next current sentence. This process continues, until the system reaches "Alice love him", which is the closest it can get to the target using the available TTRs.

Notice that the target sentence in a case is the result of first applying some replacement to the source sentence and then fixing the grammatical errors. The TTRs used in our system are designed to cover only the latter, and thus, it is not expected that the transformation path can perfectly produce the target.

Moreover, since the search process required in the *Infer TTR* step might explore a very large space, we don't use the same distance measure used for retrieval, but a simpler distance measure based on the Ratcliff-Obershelp algorithm [3]; enough for the purposes of finding a good transformation path.

During the *Apply TTR* step, the transformation path is applied to the problem in order to obtain candidate solutions. This process is performed as follows:

1. Given the query (s_1, s_r), where s_r was obtained by replacing p_1 by p_2 in s_1, and the inferred transformation path: $\{(type_1, POS_1), ..., (type_n, POS_n)\}$.
2. Apply the TTRs in the transformation path $\{(type_1, POS_1), ..., (type_n, POS_n)\}$ one at a time to obtain all the possible variants of s_r, but protecting all the words in s_r that belong to p_2 from any change.

The result is a collection of sentence variants that are candidate solutions to the problem at hand. Notice that we protect any changes to occur to the words in p_2 since we assume that the user wants p_2 to be in the solution as-is.

For example, imagine that the problem is s_1 = "Mary likes cake", $p_1 \rightarrow p_2$ = "Mary" $\rightarrow$ "I", and the inferred transformation path is: {(toVB, V3P), (toPP, NP)}. In this situation s_r would be "I likes cake". Then, all the inferred TTRs would be applied one by one:

– Applying (toVB, V3P) we obtain: "I like cake"
– The second TTR, (toPP, NP), cannot be applied since there is no proper noun in the sentence, so, we obtain no more variants.

The output of the reuse process in this example would be all the variants generated by applying all the TTRS, which in this example is just two sentences: "I likes cake", and "I like cake", which would be handed to the revise process to select which one is the best. Moreover, notice that we have used very simple transformations in the examples in this section, but in principle problems may imply very complex transformations of the sentence. For example, consider the sentence: s_1 = "Once upon a time there was a knight in Scotland, his name was Robert", and the replacement "Robert" $\rightarrow$ "Alice and Bob". The correct result would be "Once upon a time there were two knights in Scotland, their names were Alice and Bob". Notice that the resulting sentence has suffered many transformations, like "one knight" $\rightarrow$ "two knights", which are semantical rather than grammatical, but may be covered by a very simple transformation path like {(remove-word, determinant), (pluralize, NN), (insert-determinant, NN)}.

3.3 Revise

The revise process is performed automatically by a *Probabilistic Evaluator* (PE) module. In a nutshell, the PE is a standalone module implementing a language model capable of predicting the likelihood that a sentence is grammatically correct and semantically coherent. Given the candidate sentences returned by the reuse process, the PE ranks them and the one that is more likely to be correct (according to the PE evaluation) is selected as the final answer. The PE has been designed as a self contained, pluggable module so that multiple evaluators can be switched, combined or reused in other systems. Two PE modules have been developed as a proof-of-concept for the system presented in this article, each implementing a language model using a different database constructed from readily available data.

Instead of implementing a full language model capable of computing probabilities for the sentences, our PE takes for input a sentence and splits it in smaller fragments of fixed size n by using a sliding window. Then, it checks the probabilities for each fragment against a text corpus and aggregates the results into a single value. We used the arithmetic mean for aggregation after some experimentation. Although a full language model would certainly improve the results of the system, it was out of the scope of this project. In our tests with simple language models, we observed they would favor shorter sentences as the conditional probabilities diminished with longer sentences.

We used two PEs from language models trained from different corpora. One uses a frequencies database (Google n-grams) and the other uses a probability database (from the CMU Sphinx system, using the Gigaword corpus). The larger the n-grams considered by the evaluators, the better the performance, however, increasing n increases the memory required to handle the database of the evaluator exponentially. The results from each evaluator differ slightly, since the data used varies in nature. Below we briefly describe the two databases used in our experiments.

- **Google n-gram**: This evaluator uses a database based on the Google Books n-gram English dataset. The original dataset was extracted from a corpus

Table 1. Example results that the two probabilistic evaluators used in our experiments obtain for three different sentences using a 3-gram model

Sentence	*Google n-gram PE*	*Sphinx PE*
I have a pet dog.	6.695	-2.759
I have a pet wolf.	-19.743	-3.564
I have a pet hamburger.	-44.743	-4.174

of scanned books. We gathered several datasets containing n-grams up to size 4, limited to books between 1980 and 2008, discarding year information, words with non-English characters, etc. Our Google n-gram database stores, the probability of each n-gram relative to that of the first word of the n-gram (computed in standard deviations form the mean of the first word).

– **CMU Sphinx's Gigaword ARPA corpus**: This evaluator uses a database computed for a language model based on the Linguistic Data Consortium Gigaword text corpus consisting of 1.200 million words from various sources. This dataset stores the conditional probability of each n-gram given the first $n-1$ words (stored as its natural logarithm). The database contains n-grams up to size $n = 3$ but we use only one of the three available datasets.

Table 1 shows three examples of the numbers obtained with each of the two evaluators for comparison purposes. For example, for the sentence "I have a pet dog" the *Google n-gram* evaluator returns a score of 6.695, which is the average between 0.03,1.23, 4.52, and 21.0, that this evaluator gives respectively to the following n-grams: "I have a", "have a pet", "a pet dog", and "pet dog.". It is worth noting that the PE do not just check for grammatical correctness, but also for semantical consistency, as can be seen in 1, where both evaluators score a sentence containing the phrase "pet hamburger" very low, since hamburgers are typically not pets, and thus the two words appear together very rarely in the corpora used for training the language models.

4 Empirical Evaluation

In order to evaluate CeBeTA, we designed 3 experiments, reported in the following three subsections respectively. The first experiment aims at evaluating the distance metric used for retrieval. A second experiment explores the performance of the PE used for revision. The third experiment tests the design of the system as a whole.

4.1 Case Retrieval

In order to evaluate the similarity metric between sentences, we devised the following experiment. We constructed 8 groups of sentences, where each group contained between 7 and 15 sentences. The sentences in each group were constructed by taking a base sentence and modifying it in different ways. The sentences in a

Table 2. Example group of sentences used in our experiments to test the distance measure used for case retrieval. We also show the distance between the first sentence and the rest as computed by the measure used for retrieval (*Custom SED*).

Sentence	*Custom SED*
Alice loves Bob.	0.0
Alice loves Mary.	0.1
Alice loves him.	1.0
Alice loves me.	1.1
Alice hates pasta.	2.2
Alice ran away.	6.1
Alice wants to be a painter.	7.7
Alice used to have a pet dog.	9.2
Alice is going to be a painter.	9.7
Alice has always wanted to be a painter.	10.7

group were manually sorted by us according to intuitive similarity with the base sentence: the first sentence in a group is the base sentence, the second is the most similar to the base, and so on, until the last sentence, which is the most different from the base sentence. For example Table 2 shows an example of one such groups of sentences.

Different groups were authored to check for certain features. Some groups of sentences were crafted to test for basic noun and pronoun compatibility, adjective insertions and deletions, morphology changes and semantic modifications.

In order to evaluate the distance metric used for retrieval, we compared the ordering that results by sorting the sentences in each group by their similarity with the base sentence, with the order that was provided for the sentences in each group (that we consider to be the ground truth). We measured the mean square error of the position of each sentence in the order generated by the similarity measure. For example is a sentence was in position 3 in the ground truth and in position 5 in the order generated by the similarity measure, the squared error is $(3-5)^2 = 4$. The resulting mean square error was 5.13, which means that sentences are in average 2.26 positions off where they should be. Moreover, we observed that most sentences were actually exactly in the same position or at most one off, except for a few out-layers that caused an increase in the error. 36.5% of the sentences were exactly in the same position as in the ground truth, and 67.66% where exactly in the same position, or at most one position off. This shows strong evidence that the similarity measure used for retrieval strongly resembles intuitive sentence similarity.

4.2 Revision

In order to test the PE performance for solution revision, we followed a methodology similar to the previous experiment. We crafted several groups of sentences, designed to test evaluators against different grammatical and syntactic features.

Table 3. Example group of sentences used in our experiments to test performance of the PE for solution revision. We also show the ranked score given by both evaluators using 3-grams.

Sentence	*Google*	*Sphinx*
I have a pet dog.	6.703986723	-2.7593
I have a pet wolf.	-19.73622328	-3.5635
I have a pet hamburger.	-44.73622328	-4.174458333

Some sentences in each group are incorrect in different ways, some of them are grammatically incorrect (e.g. "They plays the piano."), some others are semantically incorrect (e.g. "I have a pet hamburger."), and some are correct ("I run fast."). The sentences were ordered in the group by us in increasing order of incorrectness. Specifically, we constructed 8 groups containing 3 sentences each.

Different groups have been written to check for grammatical errors and semantic incoherence. As in the previous experiment, each group contains several sentences with some text modification. Each sentence is ran trough the evaluator and they are ranked according to the output of the evaluator. We then compared the ranking obtained by the evaluator against the ground truth provided by us using the same metric as in the previous section.

We compared the performance of the two evaluators described in Section 3.3 in the 8 groups of sentences. The Google PE using the 3-grams dataset gave the correct ordering for 4 out of the 8 test sets with an averaged squared error between the results and the ground truth of 0.333, while the PE using the CMU Sphinx database was able to correctly yield the expected order for 5 out of 8 test sets, with an average squared error between the results and the groudn truth of 0.250. Table 3 shows one example set and the results given by both evaluators.

The performance of both evaluators decreased as the length of the sentence increased, and, as expected, we observed that increasing the size of the n-gram dataset in use improves the evaluator performance. We ran the experiment using the 4-gram database for the Google PE and the numeric results matched the results from the CMU Sphinx. Going beyond 4-grams is hard in practice, due to memory requirements. For example, the *Google n-gram PE* requires 9GB of storage space for 3-grams and 14GB for 4-grams.

Moreover, the PE may use an external readily available database but using a dataset trained with domain-specific corpora, if available, could greatly improve the performance in certain scenarios.

4.3 CeBeTA

We designed a holistic experiment to confirm the validity of our system. For this experiment we collected a data set of 126 pairs of sentences, where each pair contains one source sentence, and a target sentence resulting from replacing a phrase in the source sentence by another phrase, and then fixing for grammatical and semantical errors. All the experiments reported here are the result of a 6-fold cross validation run.

In order to analyze the results, we used 3 different metrics:

1. Similarity between predicted solution and ground truth: we used the same simple distance metric based on the Ratcliff-Obershelp algorithm [3] we used for the reuse process to measure the distance between the solution provided by CeBeTA and the actual ground truth.
2. Percentage of times solution was amongst the candidates: the reuse process of CeBeTA returns a collection of candidate solutions. This metric measures whether the ground truth was one of those candidate solutions.
3. Probabilistic Evaluator: the third metric uses PE to evaluate the likelihood of the solution sentence being grammatically correct and semantically coherent.

During our experiments, we realized that one of the TTRs (*noun-to-pronoun*) was particularly problematic, since it would produce correct sentences towards which the PE was being biased, but which were further from the expected ground truth. Thus, we evaluated our system with and without that TTR. Our results are summarized in Table 4, as compared to a baseline approach that would just perform the requested text replacement without doing any further operations in the sentence.

Considering metric 1, we can see that both the baseline approach and the CeBeTA system produced sentences that were extremely similar to the ground truth. This is expected, since the task the system needs to perform is to replace a part of a sentence by another, and thus, most of the sentence is typically to be left unaltered. However, small differences, like verb morphology or a noun-pronoun coordination in a sentence are important, and thus, the improvement that we observe from the baseline system (obtaining sentences with average distance to ground truth of 0.0456) to the CeBeTA system (obtaining sentences with average distance to ground truth of up to 0.0367) is very significant. For example, the distance between a syntactically incorrect sentence as in our baseline like "Alice love Bob." to the ground truth ("Alice loves Bob.") is already very low, at 0.04. The numerical relevance of the improvement of a single letter is scaled down considerably as the sentences grow longer. Moreover, after carefully analyzing the results using this metric, we found out that sometimes the system was generating perfectly valid solutions that were being penalized because they were not actually matching the provided ground truth.

Considering metric 2, we see that a blind replacement only obtains the correct result a 30.952% of times, and that the CeBeTA system generates the correct result among its candidate solutions a much higher percentage of times (73.81%), even though the system only manages to yield the exact result as expected by the ground truth 55.96% of the time when using the Sphinx PE and 53.57% when using the Google PE.

Finally, metric 3 shows that sentences generated by the baseline approach are much less likely to be correct. The Sphinx PE gives the blind replacements an average score of -3.916 and the Google n-grams gives them a score of -32.031. Those scores go up to -3.522 and -22.337 respectively for the sentences produced by the CeBeTA system.

Table 4. Comparison of the performance of CeBeTA (both with and without the *noun-to-pronoun* (*toPP*) TTR activated) against a baseline system that would blindly perform the text replacement

	Metric 1	Metric 2	Metric 3
Baseline	0.0456	30.952%	-3.916 / -32.031
CeBeTA, Sphinx	0.0398	73.810%	-3.522
CeBeTA, Google	0.0397	73.810%	-22.337
CeBeTA (without *toPP*), Sphinx	0.0388	73.810%	-3.554
CeBeTA (without *toPP*), Google	0.0367	73.810%	-23.965

5 Related Work

Several areas of work are related to the approach presented in this paper, in this section we describe related approaches to assess similarity between sentences and CBR approaches using natural language.

Similarity assessment for natural language has been studied in the field of information retrieval. With a few exceptions, such as the work of Metzler et al. [8], similarity metrics for text focus on comparing large documents, and thus use representations, such as bag-of-words [7] or bag-of-concepts [13] that neglect the subtle syntactic and semantic details that are relevant for the work presented in this paper.

CBR approaches dealing with natural language (sometimes called *Textual CBR*, or TCBR) have been widely explored in the literature, but the vast majority of this work focuses on case retrieval. There has been an increased interested in the TCBR community in the past few years to seek and go beyond retrieval, although these are still the exception. Two representative efforts in this line are the CR2N system [2] that models reuse as selecting the appropriate portions of text from a retrieved document that could be reused, and some recent work in jCOLIBRI [10], that uses a mixed initiative approach where the user collaborates with the system in order to perform text reuse. In contrast, the adaptation approach presented in this paper is fully autonomous and automatically generates new sentences (using TTRs) to be candidate solutions, that are automatically evaluated using a probabilistic evaluator.

6 Conclusions

This paper has presented a case-based reasoning approach to natural language generation, and in particular to surface realization through text modification. We have presented the CeBeTA system, that combines a sentence similarity metric with a reuse approach based on text-transformation routines, in order to generate solutions to the text modification problem. Additionally, we have seen that by using an automatically trained statistical evaluator, it is possible to distinguish between correct and incorrect solutions.

The main contributions of this paper is a new approach to text adaptation based on text transformation routines (TTRs), and the idea of automatically inferring the *transformation path* from the retrieved case, as a way to capture the adaptations that need to be performed to the problem at hand in order to generate a solution. This idea allowed us to use unannotated cases in CeBeTA.

Our empirical evaluation showed promising results, although there is a lot of room for improvement. For example, a careful examination of the obtained results revealed that CeBeTA presents issues with entity handling, since the transformation path does not contain information concerning which specific entities in the sentence need to be transformed (it just contains their POS class). Existing work in the information extraction field on entity extraction could be used for this purpose [5]. Additionally, we want to evaluate the sensitivity of the system to the number of cases in the case-base, and integrate CeBeTA with the Riu system, which was the main motivation to develop CeBeTA.

References

[1] Aamodt, A., Plaza, E.: Case-based reasoning; foundational issues, methodological variations, and system approaches. AI Communications 7(1), 39–59 (1994)

[2] Adeyanju, I., Wiratunga, N., Lothian, R., Sripada, S., Lamontagne, L.: Case Retrieval Reuse Net (CR2N): An Architecture for Reuse of Textual Solutions. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 14–28. Springer, Heidelberg (2009)

[3] Apostolico, A.: String editing and longest common subsequences. In: Handbook of Formal Languages, pp. 361–398. Springer (1996)

[4] Catherine De Marneffe, M., Manning, C.D.: Stanford typed dependencies manual (2008)

[5] Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Unsupervised named-entity extraction from the web: An experimental study. Artificial Intelligence 165, 91–134 (2005)

[6] Gervás, P., Hervás, R., Recio-García, J.A.: The role of natural language generation during adaptation in textual cbr. In: Workshop on Textual Case-Based Reasoning: Beyond Retrieval, in 7th International Conference on Case-Based Reasoning (ICCBR 2007), Northern Ireland, pp. 227–235 (August 2007)

[7] Lewis, D.D.: Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 4–15. Springer, Heidelberg (1998)

[8] Metzler, D., Dumais, S., Meek, C.: Similarity measures for short segments of text. In: European Conference on Information Retrieval (2007)

[9] Ontañón, S., Zhu, J.: Story and Text Generation through Computational Analogy in the Riu System. In: AIIDE, pp. 51–56. The AAAI Press (2010)

[10] Recio-García, J.A., Díaz-Agudo, B., González-Calero, P.A.: Textual cbr in jcolibri: From retrieval to reuse. In: Wilson, D.C., Khemani, D. (eds.) Proceedings of the ICCBR 2007 Workshop on Textual Case-Based Reasoning: Beyond Retrieval, pp. 217–226 (August 2007)

[11] Reiter, E., Dale, R.: Building Natural Language Generation Systems (2000)

[12] Ristad, E.S., Yianilos, P.N.: Learning string-edit distance. IEEE Trans. Pattern Anal. Mach. Intell. 20, 522–532 (1998)

[13] Sahlgren, M., Cöster, R.: Using bag-of-concepts to improve the performance of support vector machines in text categorization. In: Proceedings of the 20th international conference on Computational Linguistics, COLING 2004. Association for Computational Linguistics, Stroudsburg (2004)

Case-Based Reasoning for Turbine Trip Diagnostics

Aisha Yousuf and William Cheetham

General Electric Global Research, 1 Research Circle, Niskayuna, NY 12309, USA
{yousuf,cheetham}@ge.com

Abstract. General Electric created a case-based reasoning system to diagnose unexpected shutdowns of turbines. This system is currently in use at a remote monitoring facility which monitors over 1500 turbines. There are multiple root causes for turbine shutdowns. The system has a reasoner for five of the most common root causes. These reasoners use either rule-based or case-based reasoning to produce a confidence value which specifies the likelihood that the root cause for that reasoner was responsible for the shutdown. Another case-based reasoner combines these individual confidences to determine the most likely root cause and its confidence.

Keywords: Case-Based Reasoning, Root Cause Analysis, Fusion, Diagnostics, Gas Turbine.

1 Introduction

The General Electric (GE) Energy Headquarters in Atlanta, GA has a remote monitoring facility to diagnose turbine trips (unexpected turbine shutdowns) for over 1,500 gas and steam turbines. The turbines produce large amounts of electricity and typically cost over $30 million each. When a turbine trip occurs, power is not being generated, therefore, it is imperative to get the turbines running as soon as possible. The goal of the operations center is to diagnose the cause of turbine trips within 10-30 minutes. Fig. 1 shows this facility.

Currently, the diagnostic process is completely manual and the operations center is staffed by people 24 hours for 365 days a year. The operations center staff diagnoses about 5,000 trips every year. When a trip alarm is reported to the operations center, the operators query the turbine onsite monitors (OSM) for the sensor data from the turbine for an interval of time around when the trip occurred. Then the operators look at the data, and sometimes plot it to diagnose the root cause of the trip. However, with the increase in number of turbines being sold, a faster and more accurate diagnosis system is needed to increase productivity. In addition, hiring more people for the operations center is costly and it takes a long time to train the operators.

The goal of the turbine trip diagnostic system described in this paper is to pull the data, diagnose the cause of the turbine trip more accurately and faster, and then display the results for the operators to verify. Having an automated system would streamline the diagnostic process at the operations center allowing faster diagnosis and would prepare it for diagnosing more trips with the business expansion. The diagnostics system should tell the operators its confidence in its decision or let them know when it cannot make a decision.

Creating the diagnostics system provides a few challenges. First, the data is numeric time series data from hundreds of sensors on the turbine. Proper feature extraction methods should be applied to convert the time series data from each sensor to values capturing the important information about the sensor. Second, converting the operator logic into machine logic also has varying levels of difficulties as some trips can be diagnosed by simple rules while others need reasoners. Finally, when a trip occurs due to one kind of failure and the turbine is shutting down, the sensor values from other sensors also start looking faulty so it seems like more than one kind of failure has occurred. Therefore, the logic needs to be able to identify which kind of failure is the actual failure cause.

Fig. 1. Remote Monitoring Facility

In this paper we describe multiple uses of case-based reasoning (CBR) [1] in this system. The system is designed to have separate reasoners for diagnosing each kind of trip. The CBR system is used as a reasoner to diagnose some individual trips. Next, a fusion module will combine the results from all reasoners to come up with one final decision. CBR is also used in this system as a fusion module to combine the outputs from each reasoner. The paper is organized as follows: Section 2 gives a background of similar CBR systems used for diagnostics and the use of CBR as a fusion module. Section 3 gives an overview of the system design and capabilities of the system we have designed. The CBR systems are discussed in detail in Section 4. The results of the system are discussed in Section 5. We conclude and the future improvement and development plans for this system are discussed in Section 6.

2 Background

CBR systems have been used extensively at GE for monitoring and diagnostics in various applications. One of these systems was designed to identify equipment failures in locomotives to prevent locomotives from being stranded on the tracks [2]. This system has been in use since 1995 and it does diagnostics, as well prognostics to predict equipment failures that are about to occur. The case base for this system consisted of information from historical fault logs and repair history.

Another application was created to diagnose issues with computer tomography scanners in 1992 [3]. The error logs from the computer processes controlling the machines were sent in as cases for evaluation. The CBR system would allow the medical equipment services to diagnose the problem remotely for the customer.

Similar applications of CBR for diagnostics have also been explored by other companies. A CBR system employed by Ford Motor company diagnoses issues with cars [4]. Two CBR systems were created by Ford, one which uses the direct signals from signal agents and another method that extracts segment features from these signals. Bach et al. [5] also used CBR for vehicle problem diagnostics. They used service reports to diagnose problems with the vehicles.

Another application of CBR is the system called Cassiopee [6] that was developed by Acknosoft (now Servigistics) for troubleshooting problems with the CFM56-3 engine in Boeing 737s that was developed by Senecma Services and General Electric. Troubleshooting takes about 50% of the airplane's downtime so the goal of the CBR system was to reduce this by 25%. The diagnostics procedure used a combination of CBR and fault trees generated using Boeing maintenance manuals or decision trees constructed from the stored cases.

From 2002 to 2004 GE created a CBR system to diagnose turbine trips [7, 8]. That system is the predecessor to the project described in this paper. In 2004 the data that could be used for diagnosis was not so detailed as the data that is available today. The lack of data quality and precision caused the 2004 CBR project to be put on hold until now.

3 System Overview

The current turbine trip diagnostics system is deployed in the GE Energy Remote Monitoring and Diagnostics facility and is currently undergoing pilot testing. This system is designed to diagnose five different kinds of most commonly occurring turbine trips that account for 1/3 of all the trips. Additional work is being done to extend the system to diagnose other kinds of trips beyond the most commonly occurring ones diagnosed during pilot testing. The five trips diagnosed in the current system include: Compressor, Balance, Control, Manual Stop, and Speed. The names of the turbine trips have been simplified for this paper. Five reasoners are included in this system, each one to diagnose one particular trip. There is also a fusion module that combines the outputs from all the reasoners. Four reasoners are simple rule-based reasoners while one of them (Control) is a case-based reasoner. The outputs from each reasoner are confidences ranging from 0-100% [3]. The fusion module, which is also a CBR system, looks at the confidences of each reasoner to come up with one final decision.

Fig. 2 shows the complete system architecture. The system operation can be divided into four different categories. These are:

1. Data Query
2. Feature Extraction
3. Reasoners
4. Fusion

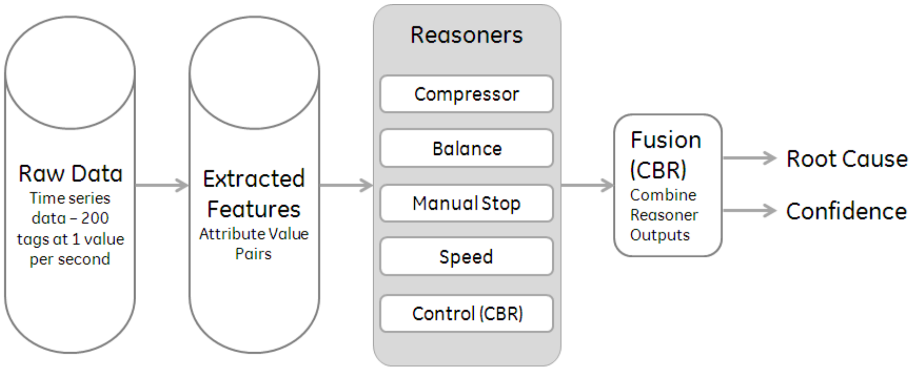


Fig. 2. System Architecture

The operator interaction with the system is done through a JSP based webpage where the operator first enters in the information about which turbine to pull the data from and for what time period. The data processed using this system is time series data (usually one value per second) and can have up to 200 different signals (or Tags) per turbine. Some of these tags contain information such as turbine speed and amount of power produced, or sensor values, or are simply indicators of a certain event occurring. These time series values can be real values or binary. Before any decisions can be made from the data, the data must first be converted from time series to attribute value pairs with values capturing the relevant aspects of the time series, for example, trends or average of last 10 seconds, etc. Hence, the webpage first calls a script that connects to the turbine OSMs to pull the data and do the feature extraction.

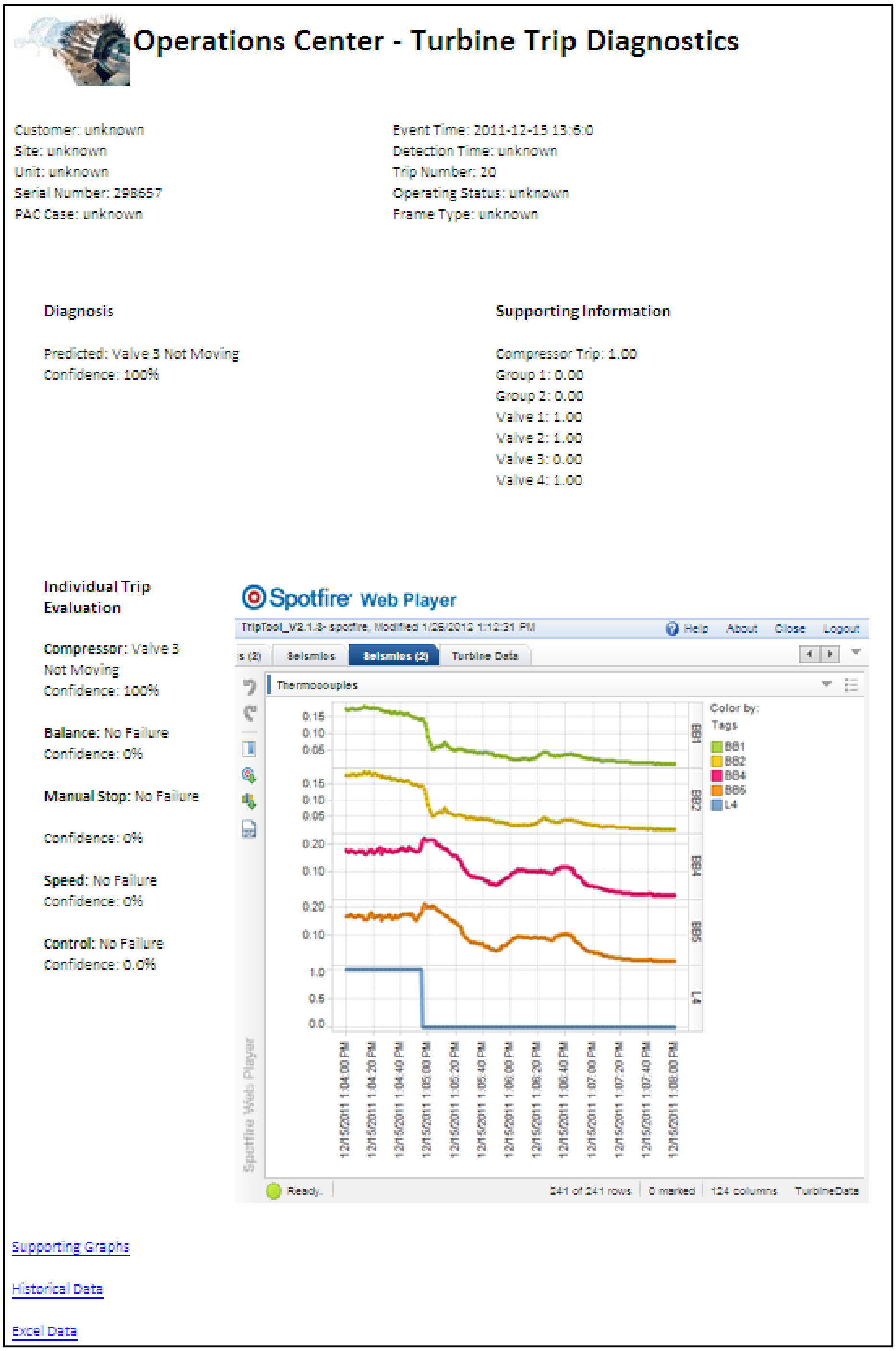
The feature extraction for some tags can be very simple which could be looking at the value of the sensor at trip time instead of the entire time series. For some other tags, such as the ones used for diagnosing Compressor trips, the values of the tags that are looked at are 10 seconds after the trip time. And yet some other tags require additional processing because the values need to be observed for a certain time period to see how they are changing. This is especially the case for Control where we look at the difference between the values of two different tags for the past 10 seconds and then average their difference.

The features extracted are passed to the individual reasoners and the CBR fusion system comes up with one final decision looking at the results from the reasoners. The fusion module and all the reasoners, including the CBR system, are written in java. Some of the reasoners in this system are simple rule-based systems while others are CBR systems. However, none of the reasoners have entire domain knowledge. Since each reasoner is focused on one kind of trip, it can only say it is that kind of trip or not. Therefore, each reasoner only has the features (knowledge) to diagnose that one trip type rather than all features. Each reasoner also gives confidence in how likely it is that it is that kind of trip. The confidence from each individual reasoner is sent to a fusion module which is another CBR system used to come up with one final decision.

The results for the fusion module, as well as all the other reasoners are then displayed on the webpage as shown in Fig. 3. The top of the display screen shows some basic information about the turbine such as the turbine number, trip number, and trip time. Then the diagnosis section shows the overall fusion result and the confidence in the final decision. The results from individual reasoners are sorted in the order from highest confidence to lowest confidence and also displayed in the Individual Trip Evaluation section. The reason that the results from all the classifiers are displayed in the webpage is because sometimes more than one reasoner would say it is that kind of trip. The operators can look at the results from the individual reasoners as well and decide if they agree with the final result. The supporting information section displays the tags for the trip that was considered to be the primary root cause.

Spotfire visualization is used for displaying the time series signals. The reason that the time series data is shown in the Spotfire visualization is if the results are incorrect, the operators can then view the entire dataset and challenge the results. The full data set is also made available to the operators for downloading in case they wanted to look at the data.

If there are any issues, such as a connection could not be established with the turbine OSM or if there is no data available to pull, the reasoners will not run and a human operator will be flagged to diagnose the turbine. Similarly, if the reasoners cannot come up with a decision or the confidence in the decision is not high enough, the operators will be asked to make the decision. It is important that the system not make a decision when it is unsure because the cost of misdiagnosis is over $100,000.00 for each trip.



Operations Center - Turbine Trip Diagnostics

Customer: unknown
Site: unknown
Unit: unknown
Serial Number: 298657
PAC Case: unknown

Event Time: 2011-12-15 13:6:0
Detection Time: unknown
Trip Number: 20
Operating Status: unknown
Frame Type: unknown

Diagnosis

Predicted: Valve 3 Not Moving
Confidence: 100%

Supporting Information

Compressor Trip: 1.00
Group 1: 0.00
Group 2: 0.00
Valve 1: 1.00
Valve 2: 1.00
Valve 3: 0.00
Valve 4: 1.00

Individual Trip Evaluation

Compressor: Valve 3 Not Moving
Confidence: 100%

Balance: No Failure
Confidence: 0%

Manual Stop: No Failure

Confidence: 0%

Speed: No Failure
Confidence: 0%

Control: No Failure
Confidence: 0.0%

Supporting Graphs

Historical Data

Excel Data

Fig. 3. Display of Results

4 Case-Based Reasoners

The current system uses two different case-based reasoners. The CBR software used is a generic internal GE tool that gives the result as well as the confidence in the result it comes up with. The first case-based reasoner is used for diagnosing the Control trip. The second case-based reasoner is used as the fusion module that takes the output (confidence) from each reasoner to come up with one final decision. The fusion case base comes up with a root cause and the confidence assigned to the determined root cause is the same as the confidence of the reasoner for that root cause.

The reasoners in the system that are not based on CBR are rule-based reasoners. Some trips are easy to detect as they have fewer features and simple rules. CBR is used when there are too many features or rules making the system complicated. CBR is also useful in handling a lot of special cases as it is easier to add cases rather than write additional rules. In addition, validating rules takes a lot of time. Hence CBR was used for reasoning which was more complex than simple rules, had more features, or for handling special cases.

4.1 CBR as a Trip Reasoner

The CBR system is used in the diagnosis for the Control trip type. There are five different valves with a similar diagnosing method for each one, however, each valve has a separate sensor monitoring it. This CBR system is called five times, once per valve with the information from sensors for that specific valve. Hence, this system not only determines whether or not it is a Control failure, but it can also tell which particular valve had the control problem.

The current system uses only the retrieval feature of a CBR system. A weighted Euclidean distance is used to determine the distance between the cases. The system looks at three closest neighbors. The maintenance of the case base is not automatic and the case base will have to be maintained by staff at the operations center as needed.

There are three sensors monitoring the control for each of the five valves. These tell information about the valve position command, the actual valve position, and the valve current. The three features sent to the CBR system are the average of the difference between the valve position and position command for the 10 seconds before the trip, the average of the valve current for the past 10 seconds and a Boolean variable indicating whether or not the valve is open. The two variables for the valve position and the current have the same weight and they are weighted higher than the valve open/close variable.

The case base for this system is fairly small. It has 160 total number of cases out of which only 12 are Control cases. There are so many other classes that the case base has to be aware of many non-Control situations, hence the large number of cases belonging to other classes. In addition, some of the cases in the case base represent cases with missing data so that the CBR system will have some example cases representing real-world data with missing information.

4.2 CBR as a Fusion Module

The five reasoners all output a confidence value between 0 and 100%. Each reasoner is not a full classifier since it only has knowledge of one trip type and not the others. Therefore, each reasoner can only say whether it is that one trip type it is focused on or not. If the reasoner has a confidence of 100%, it means that the reasoner is completely sure that it is that kind of trip. On the other hand, 0% confidence means that the system has no confidence that it is that kind of trip. Also, if the reasoner is unable to make a decision due to missing or faulty data, it outputs the confidence of 0% and also outputs an error displaying problems with data.

When a turbine trip happens, often it looks like more than one kind of failure has occurred so multiple reasoners can show confidence in the type of trip they are focusing on. Hence, the fusion CBR system is used for breaking the ties. In addition, the CBR system can handle a lot of special cases. For example, in some cases two trip types can occur together, but if they occur together, one trip type is always given higher precedence over the other. Therefore, the CBR system is a better choice than having many rules or simply selecting the highest confidence.

The inputs to the fusion CBR are only the confidences from each one of the reasoners. All the confidences are weighted equally. The fusion CBR system uses the weighted Euclidean distances as well. However, for this CBR system the decision is based on only one closest neighbor. The output from the fusion CBR is a text variable describing which kind of trip it is. Since the confidence in each kind of trip is given by the reasoner for that trip type, the fusion CBR does not calculate confidence. Instead the final output of the system is the kind of trip the fusion CBR determined it is, and the confidence from the reasoner for that trip type.

The case base captures three different kinds of variations in the data. First is when only one kind of reasoner shows confidence greater than 0, in which case it is obviously that particular trip type. Second, the case base contains cases that account for the confidence variation for each particular trip type. Prior to designing the case base, a test was conducted using the reasoners to see which trips occurred together. So lastly, the cases in the CBR system were then designed to account for these overlaps for helping make a decision when more than one reasoner has a confidence greater than 0. The fusion CBR is also maintained by the operations center staff and currently contains 35 cases.

5 Results

We created a test set that consisted of all expert verified trips for a given period of time. These trips are not included in the case base. The amount of test data for each trip type is shown in Table 1. The amount of data available is fairly small and the numbers of cases are not balanced by classes. This is mainly due to the fact that certain trip types occur more than others and verification is labor intensive.

Table 1. The amount of data for each trip type

Trip Type	Number of Cases
Control	17
Balance	77
Compressor	21
Manual Stop	13
Speed	3

The performance of the system was judged in two separate steps. First the performance of each individual reasoner was evaluated. Then the performance of the system as a whole with the classifier fusion module was evaluated. The results for the control reasoner are shown in Table 2. It can be observed that out of 17 total control test points available, only 1 was misclassified. In Table 2 the "other" cases don't necessarily mean other trip cases from Table 1, but they represent correctly running turbines, turbines with non-Control faults, and trips with missing data.

Table 2. Confusion matrix for Control trip type results

a	b	← Classified as
16	1	**a = Control**
0	172	**b = Other**

The results for the overall system are shown in Table 3. There are four performance metrics shown for measuring the success for each individual reasoner. The correct classifications are where the reasoner for the correct trip type is showing some confidence in that class and the reasoners for all the other classes do not show any confidence. Incorrect classification is when the reasoner for that particular kind of trip shows 0% confidence in the result, regardless of whether or not other reasoners show any kind of confidence in this trip. Mixed good results are when more than one kind of classifier is showing some confidence in the trip but the highest confidence is in the correct class. Mixed Bad results are when more than one kind of classifier is showing some confidence in the trip but an incorrect class has a higher confidence or the same confidence as the correct class. The overall fusion results show that the fusion module helps break ties by classifying the mixed results to correct or incorrect classes. Overall the system is showing 90.07% accuracy.

In addition to good performance, this system also improved the diagnosis time. Currently, the manual process takes about 10-30 minutes to diagnose one trip. Our system takes on average about 2 minutes to diagnose a trip. Out of the two minutes the CBR systems take about 1-2 seconds each. Most of the diagnosis time is connectivity. Since the turbines can be located anywhere in the world, connecting to the turbines to get the data can take a long time depending on the location and connectivity of the location.

Table 3. Results of the entire system

Trip Type	Individual Reasoner Result	Results After Fusion
Control	Correct: 15 Incorrect: 0 Mixed Good: 1 Mixed Bad: 1	Correct: 16 Incorrect: 1
Balance External factors	Correct: 65 Incorrect: 5 Mixed Good: 6 Mixed Bad: 1	Correct: 71 Incorrect: 6
Compressor	Correct: 19 Incorrect: 0 Mixed Good: 0 Mixed Bad: 2	Correct: 19 Incorrect: 2
Manual Stop	Correct: 5 Incorrect: 4 Mixed Good: 3 Mixed Bad: 1	Correct: 9 Incorrect: 4
Speed	Correct: 3 Incorrect: 0 Mixed Good: 0 Mixed Bad: 0	Correct: 3 Incorrect: 0

6 Conclusions and Future Work

Through the turbine trip diagnostics system we have demonstrated the use of CBR as an individual reasoner, as well as a method for classifier fusion. Our system shows reasonable accuracy with the results of the system being about 90% accurate. The system is integrated into GE Energy's system showing that it is capable of working in its operational environment.

Many improvements still need to be made to the system. In the next phase, we plan on increasing the accuracy of the system. We are also currently working on a more reliable way of connecting to the turbines that will pull all the data and we will minimize the cases where we cannot make a decision due to missing data. In addition, we are adding more reasoners to the system to diagnose more trip types. The fusion system will have to be reconfigured to handle other trips. In addition, more features will be added to the system such as trip alarms instead of just using sensor data to further help break the ties between overlapping classes.

Another future step for our system would be to either create an automated method of maintaining the case base or train operations center staff on how to maintain the case base. Since the case base is not updated by the system, there needs to be a system for adding more cases, revising the ones that already exist in the system, and deleting the ones that become obsolete.

Acknowledgements. We would like to acknowledge Michael Whitford and Robert Iasillo from GE Energy for their contribution to this project. They are field experts in diagnosing turbine trips and helped us greatly in understanding the turbine diagnosis logic. We would also like to acknowledge Mark Osborn from GE Energy for helping us with the code to connect to the turbines. Lastly, we would like to acknowledge George Williams from GE Global Research for developing the Spotfire visualization tool for viewing the turbine data.

References

1. Aamodt, A.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AICOM 7(1) (March 1994)
2. Varma, A., Roddy, N.: ICARUS: A Case-Based System for Locomotive Diagnostics. Engineering Applications of Artificial Intelligence Journal (1999)
3. Cuddihy, P., Cheetham, W.: ELSI: A Medical Equipment Diagnostic System. In: Althoff, K.-D., Bergmann, R., Branting, L.K. (eds.) ICCBR 1999. LNCS (LNAI), vol. 1650, pp. 415–425. Springer, Heidelberg (1999)
4. Wen, Z., Crossman, J., Cardillo, J., Murphey, Y.L.: Case Base Reasoning in Vehicle Fault diagnostics. In: Proceedings of the International Joint Conference on Neural Networks, pp. 2679–2684 (2003)
5. Bach, K., Althoff, K.-D., Newo, R., Stahl, A.: A Case-Based Reasoning Approach for Providing Machine Diagnosis from Service Reports. In: Ram, A., Wiratunga, N. (eds.) ICCBR 2011. LNCS, vol. 6880, pp. 363–377. Springer, Heidelberg (2011)
6. Heider, R.: Troubleshooting CFM 56-3 Engines for Boeing 737 Using CBR and Data-Mining. In: Smith, I., Faltings, B.V. (eds.) EWCBR 1996. LNCS, vol. 1168, pp. 513–518. Springer, Heidelberg (1996)
7. Cheetham, W., Price, J.: Measures of Solution Accuracy in Case-Based Reasoning Systems. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 106–118. Springer, Heidelberg (2004)
8. Devaney, M., Cheetham, W.: Case-Based Reasoning for Gas Turbine Diagnostics. In: The Eighteenth International FLAIRS Conference, Clearwater Beach, Florida (2005)

Author Index