Vadim Ermolayev Heinrich C. Mayr
Mykola Nikitchenko Aleksander Spivakovsky
Grygoriy Zholtkevych (Eds.)

Communications in Computer and Information Science 412

Information and Communication Technologies in Education, Research, and Industrial Applications

9th International Conference, ICTERI 2013
Kherson, Ukraine, June 2013
Revised Selected Papers

Springer

Communications in Computer and Information Science 412

Vadim Ermolayev Heinrich C. Mayr
Mykola Nikitchenko Aleksander Spivakovsky
Grygoriy Zholtkevych (Eds.)

Information and Communication Technologies in Education, Research, and Industrial Applications

9th International Conference, ICTERI 2013
Kherson, Ukraine, June 19-22, 2013
Revised Selected Papers

Springer

Volume Editors

Vadim Ermolayev
Zaporozhye National University, Ukraine
E-mail: vadim@ermolayev.com

Heinrich C. Mayr
Alpen-Adria-Universität Klagenfurt, Austria
E-mail: heinrich.mayr@aau.at

Mykola Nikitchenko
Taras Shevchenko National University of Kyiv, Ukraine
E-mail: nikitchenko@unicyb.kiev.ua

Aleksander Spivakovsky
Kherson State University, Ukraine
E-mail: spivakovsky@ksu.ks.ua

Grygoriy Zholtkevych
V.N. Karazin Kharkiv National University, Ukraine
E-mail: g.zholtkevych@karazin.ua

ISSN 1865-0929 e-ISSN 1865-0937
ISBN 978-3-319-03997-8 e-ISBN 978-3-319-03998-5
DOI 10.1007/978-3-319-03998-5
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013955355

CR Subject Classification (1998): K.3, D.2, H.4, I.2, H.3, F.3, D.3, C.2

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

It is our pleasure to present the post-proceedings of ICTERI 2013, the 9th International Conference on Information and Communication Technologies (ICT) in Education, Research, and Industrial Applications: Integration, Harmonization, and Knowledge Transfer. The conference was held at Kherson, Ukraine during June 19-22, 2013. This volume is composed of the substantially revised and extended versions of the best ICTERI 2013 papers. The selection was made based on the quality, anticipated reader interest, and coverage of the conference scope.

ICTERI as a conference series is concerned with interrelated topics that are vibrant for both the academic and industrial communities:

- ICT infrastructures, integration, and interoperability
- Machine intelligence, knowledge engineering (KE), and knowledge management (KM) for ICT
- Model-based software system development
- Methodological and didactical aspects of teaching ICT and using ICT in education
- Cooperation between academia and industry in ICT

This book begins with the three invited contributions presenting the substance of ICTERI 2013 invited talks given by Wolf-Ekkehard Matzke, Aleksander Letichevsky, and Gary L. Pratt. The rest of the volume is structured in four topical parts:

- Systems, Infrastructures, and Integration
- Semantics, Knowledge Engineering, and Management
- ICT in Teaching Methodologies and Didactics
- Model-Driven Software Development and Verification

Part one of the volume is dedicated to software systems, infrastructures development, and integration. It begins with a paper reporting on the development and deployment of an integrated framework for IT service management at a university. The second paper presents a novel technology for developing and optimizing parallel programs. The third paper provides a rigorous theoretical framework for the asymptotical assessment of the information in the consecutive series of tests of quantum information systems. The final paper in this part is focused on assessing the availability of computer systems using the formalism of stiff Markov chains.

The second part of the volume contains papers on semantics, knowledge engineering and management. It opens with a paper elaborating metrics for ontology fitness in the OntoElect methodology. The second paper deals with the evaluation of the ontology instance migration methodology applied to the industrial size ontologies in the construction domain. The third paper presents the semantic models of the relationships between different clocks under constraints.

Part three of the volume focuses on using ICT in teaching and learning and their effects on didactics. The first paper shares the observations and experiences of how computer science students use version control systems in their coursework projects. The second paper is concerned with the problem of the generation gap as an obstacle for teaching ICT and offers a way to bridge this gap. The third paper proposes a holistic approach to developing the didactics in the ICT curricula by including students, tutors, and an ICT-based teaching environment in the framework. The fourth paper focuses on the use of the emerging ICT for training educators.

The fourth part of the volume presents the advancements in model-driven software development and verification. It begins with a paper presenting the way to improve the efficiency of a synchronized product with infinite transition systems. The second paper formally investigates and proves the existence of total input-output pairs of abstract time systems. The third paper elaborates the aspect of specializations in symbolic verification. Finally, the fourth paper extends the Floyd–Hoare logic for partial pre-conditions and post-conditions.

On the whole, ICTERI 2013 had attracted 125 submissions out of which 50 were selected for presentation at the conference by the Program Committee. In the second selection phase the Steering Committee chose 24 papers as the candidates to be revised and extended for the post-proceedings volume. Out of these 24 extended submissions we finally accepted the 18 most interesting papers. This resulted in an acceptance rate of 14%.

This volume would not have materialized without the support of many people. First, we would like to thank the members of our board of reviewers for providing timely and thorough assessments, and also for being very cooperative in doing additional review work at short notice. We are very grateful to all the authors for their continuous commitment and intensive work. Furthermore, we would like to thank all the people who contributed to the success of ICTERI 2013. Without their effort there would have been no substance for this volume. Finally, we would like to acknowledge the proactive support of our editorial assistant Olga Tatarintseva.

Otober 2013

Vadim Ermolayev
Heinrich C. Mayr
Mykola Nikitchenko
Aleksander Spivakovsky
Grygoriy Zholtkevych

ICTERI 2013 Conference Organization

General Chair

Aleksander Spivakovsky	Kherson State University, Ukraine

Steering Committee

Vadim Ermolayev	Zaporozhye National University, Ukraine
Heinrich C. Mayr	Alpen-Adria-Universität Klagenfurt, Austria
Mykola Nikitchenko	Taras Shevchenko National University of Kyiv, Ukraine
Aleksander Spivakovsky	Kherson State University, Ukraine
Mikhail Zavileysky	DataArt, Russian Federation
Grygoriy Zholtkevych	V.N. Karazin Kharkiv National University, Ukraine

Program Co-chairs

Vadim Ermolayev	Zaporozhye National University, Ukraine
Heinrich C. Mayr	Alpen-Adria-Universität Klagenfurt, Austria

Workshops Chair

Mykola Nikitchenko	Taras Shevchenko National University of Kyiv, Ukraine

Tutorials Chair

Grygoriy Zholtkevych	V.N. Karazin Kharkiv National University, Ukraine

IT Talks Co-chairs

Aleksander Spivakovsky	Kherson State University, Ukraine
Mikhail Zavileysky	DataArt, Russian Federation

Publicity Chair

Nataliya Kushnir	Kherson State University, Ukraine

Technical Assistant to Volume Editors

Olga Tatarintseva	Zaporozhye National University, Ukraine

Program Committee

Rajendra Akerkar	Western Norway Research Institute, Norway
Eugene Alferov	Kherson State University, Ukraine
Costin Badica	University of Craiova, Romania
Tobias Buerger	PAYBACK, Germany
Andrey Bulat	Kherson State University, Ukraine
David Camacho	Universidad Autonoma de Madrid, Spain
Michael Cochez	University of Jyväskylä, Finland
Maxim Davidovsky	Zaporozhye National University, Ukraine
Anatoliy Doroshenko	National Technical University of Ukraine “Kyiv Polytechnic Institute”, Ukraine
Vadim Ermolayev	Zaporozhye National University, Ukraine
David Esteban	Techforce, Spain
Lyudmila Gavrilova	Slovyansk State Pedagogical University, Ukraine
Vladimir Gorodetsky	St. Petersburg Institute for Informatics and Automation of The Russian Academy of Science, Russian Federation
Marko Grobelnik	Jozef Stefan Institute, Slovenia
Brian Hainey	Glasgow Caledonian University, UK
Sungkook Han	Wonkwang University, South Korea
Mitja Jermol	Jozef Stefan Institute, Slovenia
Jason Jung	Yeungnam University, South Korea
Natalya Keberle	Zaporozhye National University, Ukraine
Nick Kings	Connected Shopping, UK
Christian Kop	Alpen-Adria-Universität Klagenfurt, Austria
Hennadiy Kravtsov	Kherson State University, Ukraine
Nataliya Kushnir	Kherson State University, Ukraine
Frédéric Mallet	Université de Nice-Sophia Antipolis, France
Mihhail Matskin	Royal Institute of Technology, Sweden
Heinrich C. Mayr	Alpen-Adria-Universität Klagenfurt, Austria
Mykola Nikitchenko	Taras Shevchenko National University of Kyiv, Ukraine
Andriy Nikolov	Knowledge Media Institute, The Open University, UK
Inna Novalija	Jozef Stefan Institute, Slovenia
Tomás Pariente Lobo	ATOS Origin, Spain
Vladimir Peschanenko	Kherson State University, Ukraine
Carlos Ruiz	Playence, Spain
Abdel-Badeeh Salem	Ain Shams University, Cairo, Egypt
Wolfgang Schreiner	Research Institute for Symbolic Computation (RISC), Johannes Kepler University, Austria
Vladimir A. Shekhovtsov	Alpen-Adria-Universität Klagenfurt, Austria

Mikhail Simonov	Istituto Superiore Mario Boella, Italy
Marcus Spies	Ludwig-Maximilians-Universität München, Germany
Aleksander Spivakovsky	Kherson State University, Ukraine
Martin Strecker	IRIT, Universite Paul Sabatier, France
Olga Tatarintseva	Zaporozhye National University, Ukraine
Vagan Terziyan	University of Jyväskylä, Finland
Nikolay Tkachuk	National Technical University “Kharkiv Polytechnic Institute”, Ukraine
Leo Van Moergestel	Utrecht University of Applied Sciences, The Netherlands
Maryna Vladymyrova	V.N. Karazin Kharkov National University, Ukraine
Paul Warren	Knowledge Media Institute, the Open University, UK
Iryna Zaretska	V.N. Karazin Kharkov National University, Ukraine
Mikhail Zavileysky	DataArt Solutions Inc., Russian Federation
Grygoriy Zholtkevych	V.N. Karazin Kharkov National University, Ukraine

Additional Reviewers

Fahdi Al Machot	Alpen-Adria-Universität Klagenfurt, Austria
Antonio Gonzalez-Pardo	Universidad Autonoma de Madrid, Spain
Alexey Vekschin	National Technical University “Kharkiv Polytechnic Institute”, Ukraine

Post-Proceedings Board of Reviewers

Costin Badica	University of Craiova, Romania
Anatoliy Doroshenko	National Technical University of Ukraine “Kyiv Polytechnic Institute”, Ukraine
Vadim Ermolayev	Zaporozhye National University, Ukraine
David Esteban	Techforce, Spain
Heinrich C. Mayr	Alpen-Adria-Universität Klagenfurt, Austria
Mykola Nikitchenko	Taras Shevchenko National University of Kyiv, Ukraine
Andriy Nikolov	Knowledge Media Institute, The Open University, UK
Aleksander Spivakovsky	Kherson State University, Ukraine
Martin Strecker	IRIT, Université Paul Sabatier, France
Olga Tatarintseva	Zaporozhye National University, Ukraine

Vagan Terziyan	University of Jyväskylä, Finland
Nikolay Tkachuk	National Technical University “Kharkiv Polytechnic Institute”, Ukraine
Leo Van Moergestel	Utrecht University of Applied Sciences, The Netherlands
Paul Warren	Knowledge Media Institute, the Open University, UK
Grygoriy Zholtkevych	V.N. Karazin Kharkov National University, Ukraine

Additional Reviewers for Post-Proceedings

Hennadiy Kravtsov	Kherson State University, Ukraine

ICTERI 2013 Sponsors

Kherson State University www.ksu.ks.ua
DataArt www.dataart.com

Table of Contents

Biotechnology, Synthetic Biology, and ICT Define the Emerging Knowledge-Based Bio-Economy

Wolf-Ekkehard Matzke

MINRES Technologies GmbH, Neubiberg, Germany
wolf@minres.com

Abstract. Researchers, policy-makers, and industry practitioners alike herald the Knowledge-Based Bio-Economy (KBBE) as the "Next Big Thing" in societal progress. The story, which receives a lot of attention, goes like this: Over the last decades, economies around the globe have transformed into a Knowledge-Based Economy (KBE). Knowledge became the Economy-Defining Resource (EDR) of the KBE and Information and Communication Technology (ICT) has served as its Economy-Defining Technology (EDT). Now modern biotechnology becomes another EDT – leading to the KBBE. Bio-resources jointly with knowledge take on the role of the KBBE EDRs. At the heart of this story lies the relationship between resources, technologies and economies, which, under distinct circumstances, may give rise to a major economic transformation. This raises a number of important questions: What constitutes the implied primacy of economics? Why is the KBBE the next big thing? What is the role of resources, namely knowledge and bio-resources, in the context of economic transformations? Why are ICT and biotechnology the EDTs for these economic transformations? With respect to the KBBE, what are the enabling technologies? Last but not least, given the overall ICTERI conference theme and given that ICT remains an EDT for the KBBE, what are the high-level and long-term KBBE ICT challenges? The paper discusses these challenges and attempts to answer these questions at a conceptual level.

Keywords: Economics, Economy-Defining Resource, Economy-Defining Technology, Knowledge-Based Bio-Economy, Biotechnology, Synthetic Biology, Information and Communication Technology, Computational Thinking, Complexity Engineering, Bio-Design Automation.

1 Introduction

This discussion paper is based on the keynote talk given to the ICTERI 2013 conference[1] at Kherson, Ukraine. The paper is conceptual in its nature and aims to analyze some aspects of the relationship between resources, technologies and economies in light of current and upcoming advanced economies, namely the Knowledge-Based Economy (KBE) and the Knowledge-Based Bio-Economy (KBBE). The particular

[1] http://www.icteri.org/page/icteri-2013

focus of this material is on the KBBE. Therefore, the biological dimension of the KBBE will receive more attention than its knowledge dimension. Given the range of topics to be covered and the original conference audience, the paper chooses breath over depth. This allows obtaining a unifying view across different subjects; even so they are only touched briefly. At the same time, an interested reader can find starting points for more detailed discussions in the comprehensive list of references.

In a recent popular science book on economics [27][2] the author finds out that "Economics is one of the most useful and one of the least known sciences." This seems to describe the situation well. Indeed, people experience economics, but rarely understand what is happening. Hence, the paper starts with a basic background in economics in terms of resources, engineering, and technologies. Given the importance of the subject for our topic (and in general), it appears appropriate to devote more space to this section.

The paper proceeds with discussing knowledge and bio-resources as distinguished resources. Their favorable economic properties qualify them as Economy-Defining Resources (EDRs) for the KBE and KBBE. In fact, the names KBE and KBBE are derived from the names of these resources. However, for a resource to become an EDR of an economy, it takes appropriate technologies that utilize this resource optimally and allow for corresponding businesses to develop and flourish. If these businesses eventually constitute a new economy, the driving technologies take the status of Economy-Defining Technologies (EDTs). One of the major EDTs for the KBE is Information and Communication Technology (ICT). For the KBBE, biotechnology is added as an EDT. The principal enabling technology of biotechnology is synthetic biology, which is increasingly based on ICT and knowledge technologies.

The paper concludes with the discussion of high-level and long term ICT-related challenges in the context of KBBE. The analysis is structured along the dimensions of education, research, and industry applications. For the facet of education the challenge of empowering synthetic biologists with computational thinking is discussed. In the context of KBBE-related research the major challenge is seen in the complexity of bio-systems. Therefore, complexity engineering and its use in KBBE research and development is offered as an important focal point. For industrial applications, the inspiration is taken from the developments in Engineering Design Automation for mechanical and electronic designs. Based on that, the emerging Bio-Design Automation is promoted as a way that enables Synthetic Biology designs and solutions leaving the lab and gaining industrial strength.

2 Economics

Economics appear to be a distant concept for most people. This is unfortunate (at least for these people). Why? Perhaps the main reason is that economics are omnipresent across all dimensions and levels of society. Economics affect every person, every day

[2] It should be noted that this book, even popular in nature, is highly recommended in a review by James E. Miller, editor-in-chief of the well-respected Ludwig von Mises Institute of Canada (`http://mises.ca/posts/articles/in-review-blondie-economics/`).

of their lives. Economics rule the day. Without knowledge of economics people are likely to be reduced to mere objects (rather than acting as subjects). Thus, it is not surprising that various 21st century skills frameworks, e.g. [49], [58], [14] identified economics literacy consistently as a fundamental 21st century skill, a skill “people need for work, citizenship, and self-actualization” [19].

Given the apparent societal primacy of economics, what is the underlying cause? The answer lies in the objective scarcity of resources relative to human needs and wants. This has come to be known as the axiom of resource scarcity.

A resource is everything which contributes directly or indirectly to the satisfaction of human needs or wants.

A human need is a basic necessity of life in a given society. Note the societal dependence of human needs. The more advanced a society is, the higher becomes the variety of human needs beyond basic biological necessities (physiological or psychological). For example, electricity has become a necessity of life in developed countries. A major and lasting blackout will result at best in enormous economic damage, at worst in extensive social disorder and potentially the destruction of the society [40]. It should be noted that the general evolution of ICT is towards a basic utility service, hence becoming a human need.

Contrarily to human needs, a human want is not a necessity. A want is described as “having a strong desire for something.”[3] Based on this a human want can be defined as the expression of a desire shaped by a person’s socio-cultural context and individual development. Here too, the more advanced a society is, the higher becomes the variety of human wants.

The axiom of resource scarcity is the founding axiom of economics. No resource scarcity, no need for economics. If there is not enough of a required resource, it is straightforwardly clear that some trade-offs between alternative choices have to be made. Given that people have competing goals (and values), how can it be decided for what purpose and to whom the scarce resources will be allocated? This is where economics come into play. In his influential book “An Essay on the Nature & Significance of Economic Science” [59], English economist Lionel Robbins defined economics as "the science which studies human behavior as a relationship between given ends and scarce means which have alternative uses." This definition tells us what economics is and what it does, but not why it is useful. Bishop describes the societal promise of economics (the usefulness of economics) as follows [6]: “At its best, economics helps people to make the right choices; at least, it shows them the most efficient way to use scarce resource in the process of achieving their goals.”

Economics appear to be a distant concept for many engineers. This is not only unfortunate; it’s a serious problem for engineering as a discipline. Why? The answer lies in the definition of engineering. On a side note, computer scientists, ICT researchers, ICT practitioners, biotechnologists, genetic engineers etc. are all engineers. One definition of engineering is “the application of scientific and mathematical principles to practical ends such as the design, manufacture, and operation of efficient and

[3] Merriam-Webster Dictionary: `http://www.merriam-webster.com/dictionary/want`

economical structures, machines, processes, and systems."[4] Another definition describes engineering similarly as "the application of science and mathematics by which the properties of matter and the sources of energy in nature are made useful to people."[5] There are, perhaps and not surprisingly, more definitions of engineering in the literature, e.g. [15]. Notably, all definitions position engineering consistently between resources (scarce means) and technology (given ends). Note that "scarce means" and "given ends" represent the terminology used by Robinson in his definition of economics, as quoted above. At this position engineering takes on the critical role of transforming resources into technology aiming at satisfying human needs and wants. Engineering does so by constantly optimizing the use of resource including replacing them by more favorable ones. Without dispute, this represents a key economic function. Of course, for technologies to have an economic impact, that is to satisfy human needs and wants, it takes businesses that successfully market these technologies. For engineering not understanding economics will result at best in a suboptimal performance of its core function. At worst valuable resources will be wasted. Henry Towne was perhaps the first researcher and industrialist recognizing the essential importance of economics for engineering. Considerably more than a century ago, on May 26, 1886, he delivered a talk entitled "The Engineer as an Economist" to the annual meeting of the American Society of Mechanical Engineers [63]. In this talk Towne stresses the importance of competencies in economics to engineers' work. Further, in his address of February 24, 1905, at the Purdue University, Towne emphasizes that "… the engineer is, by the nature of his vocation, an economist. His function is not only to design, but also so to design as to ensure the best economical result" [64][6]. It's really puzzling that there is still a lack of understanding of economics in the engineering community, even though the issue has been recognized for such a long time.

Understanding economics is described by Zieger in [78] as "a process of gathering information, making sense of information, building conceptual models, and using these models to evaluate and analyze different situations and alternatives."

Last but not least, it is clear from the above, that resources are the crux of the matter.

3 Economy-Defining Resources (EDRs)

The key property of a resource in an economic setting is its scarcity, ranging from scanty to abundant. Distinguished resources are those which exhibit significantly more favorable scarcity properties than classical economic resources. EDRs are

[4] American Heritage Dictionary: `http://www.ahdictionary.com/word/search.html?q=engineering`

[5] Merriam-Webster Dictionary: `http://www.merriam-webster.com/dictionary/engineering`

[6] Also available at: `http://www.stamfordhistory.org/towne-1905-print.htm`

distinguished resources that form the primary resource base of an economy. They are not replaceable.

3.1 Knowledge as a Resource

The very nature of knowledge as a resource is radically different from any other economic resource.

Firstly, and most importantly, knowledge is not consumed or depleted when it is used (unlike physical resources). It would be wrong, however, to conclude that knowledge is abundant per se. Knowledge can very well be scarce in many instances.

Secondly, one of the most striking qualities of knowledge is that it grows with its utilization and application (appreciation) [13]. Based on an argumentum e contrario, it follows that knowledge, which is not constantly shared challenged and renewed, will eventually lose value (depreciation) [7].

Thirdly, yet another remarkable quality of knowledge is that it can be created by anyone, anywhere, anytime. Yet, the dissemination of knowledge might be very well controlled by temporal and spatial constraints.

It must be noted that knowledge is an intangible resource. That is knowledge needs to be always complemented by a tangible resource to satisfy a human need or want. As van Moost put it [66]: “The idea that knowledge in isolation from materials and transforming power is wealth is patent nonsense. You cannot eat knowledge and it won't keep the rain off your head or warm your family on a cold night.”

3.2 Bio-resources

In terms of scarcity, bio-resources are distinguished by two unique characteristics.

First off, bio-resources are distinguished by their intrinsic ability for highly nonlinear reproduction (exponential, hyperbolic, etc.) within environmental constraints. Therefore bio-resources exhibit a bounded nonlinear renewability. For example, under ideal conditions (no environmental constraints) a growing bacterial population doubles at regular intervals (generation time T_g). At generation g $(g = 1,2,3,...)$ and time $t_g = gT_g$ the number n_g of the bacteria is simply:

$$n_g = 2^g . \tag{1}$$

To illustrate the power of exponential growth, the bacterium, Escherichia coli (E. coli), commonly found in intestine of human, is used as an example. Note, that E. coli is extensively used in modern biotechnology. The generation time of E. coli is about $T_g = 20\,\mathrm{min}$ under ideal laboratory conditions. After 20 min the bacterium divides into 2 $(g = 1)$. After another 20min, these 2 bacteria each divide into 2. Therefore at generation $g = 2$ the total population of bacteria equals 4. The process of cell-division continues as follows: after 1 hour – into 8 $(g = 3)$; after 2 hours – into 64 $(g = 6)$; after 4 hours – into 4 096 $(g = 12)$; after 8 hours – into 16 777 216

$(g = 12)$; and after a day, at generation $g = 72$, the total population would reach the size of 4.722×10^{21}. The mass of one Escherichia coli bacterium is about 10^{-12} g. Therefore, after one day only, the total population mass would have been increased from virtually zero to 4 722 366 kg. At generation 123, after 41 hours or less than 2 days, the total population mass (1.063×10^{25} kg) would surpass the mass of the Earth (which is about 5.97219×10^{24} kg). Of course, this is purely theoretical, as in reality environmental constraints will stop the process at early stages. Having said this, the example demonstrates impressively the power of exponential growth.

Secondly, at the cell level bio-resources are Nano-factories. A handy technical definition describes a Nano-factory as "a machine that can make a product atom by atom (molecular nanotechnology). Considered the ultimate manifestation of nanotechnology, the concept is that if parts can be built at the molecular level, the Nano-factory can build almost anything, even more versions of itself, which would be its first task."[7] Biological cells match this definition nearly perfectly (except until now the almost anything part). For example, bacteria are Nano-factories that replicate themselves "in an exponential proliferation of manufacturing" [39]. The concept of Nano-factories includes the notion of programmability of bio-resources at the genetic level (not meant as an abuse of the concept of programmability [16]). For example, genetic modifications of plants have been demonstrated to improve yield, pest and herbicide tolerance, physical robustness, and heat and drought tolerance, e.g. [33].

Perhaps the best thing about bio-resources is that you can eat them and they can keep the rain of your head or warm your family on a cold night.

4 Economy-Defining Technologies (EDTs)

Certain techno-scientific innovations, that are able to unleash the potential of distinguished resources, are so disruptive as to initiate a lasting leap in societal progress by enabling the creation of new businesses. If these innovative technologies form the principal technological basis of an economy, they shall be called EDTs.

On a side note, the concept of General Purpose Technologies (GPTs), often found in the literature, e.g. [8], [44], seem to be at a first glance a competing concept to the proposed concept of EDTs. GPTs are adopted by a wide range of sectors and "induce the massive relocation (and concomitant reorganization) of economic activity, which brings about widespread productivity gains and hence long-term growth" [61]. However, GPTs can come and go within one and the same economic framework. EDTs, to the contrary, are intrinsic to the corresponding economy and stay. For example, the steam engine initiated unquestionably the industrial revolution and represents a GPT. However, steam is not a distinguished resource. Therefore, there is no such thing like a Steam Economy. Similarly, there is no Electricity Economy etc. The corresponding economy is rather referred to as (capitalist) industrial economy. Consequently, EDTs are by their very nature GPTs, but not the other way around.

[7] YourDictionary: `http://computer.yourdictionary.com/nanofactory`

4.1 ICT and the KBE

One could argue that knowledge was always a central resource of any preceding economy. The key point in the context of the KBE is however, that Information and Communication Technologies (ICTs) free knowledge from its temporal and spatial constraints. This is obvious as far as it concerns explicit knowledge. But even for the transfer of tacit knowledge, ICTs soften significantly the temporal and spatial constraints. This is what allows knowledge to unfold its powers as a resource to the utmost, eventually leading to the KBE.

Based on this, the Organization for Economic Co-operation and Development (OECD), the main promoter of the KBE, defines the KBE in the following way [57]: "The knowledge based economy is an expression coined to describe trends in advanced economies towards greater dependence on knowledge, information and high skill levels, and the increasing need for ready access to all of these by the business and public sectors." It follows that in the KBE knowledge takes primacy for individuals and organizations.

4.2 Biotechnology and the KBBE

Simply speaking, biotechnology refers to the utilization of bio-resources to satisfy human needs and wants. As such, bio-technology is one of the oldest technologies. People have been manipulating living things to satisfy human needs and wants for millennia. For example, the history of beer dates back thousands of years. In [26] Feasey points out that "the earliest written account of beer and how to make it is thought to be in the 'Hymn to Ninkasi[8]' detailed on a Sumerian stone tablet about 4000 years ago." Beer production involves the biotechnological steps of saccharification of starch and the fermentation of the resulting sugar. Even longer back in history (ca. 10.000 years ago) early agrarian societies began to till land and domesticize wild emmer wheat. The domestication process involved the biotechnological step of artificially selecting strains of wheat with increased grain sizes [30].

Modern biotechnology was made possible by recombinant DNA technology, that was developed in the early 1970s [42]. Recombinant DNA technology is a technology that allows introducing targeted genetic changes into organisms. These changes are considered to more predictable than those introduced through conventional breeding techniques, e.g. [43]. Modern biotechnology is a diverse and broad field. It has applications in pharmaceuticals and health care, environmental protection, agriculture, and synthesis of products such as biofuels and bio-degradable plastics. Biotechnology applications even go across and bridge these important areas in a mutually beneficiary way. [18] provides an excellent summary of topical examples, such as: "modifying energy crops properties, or analyzing circulating nucleic acid elements, bringing benefits for all through increased food production, supporting climate change adaptation and the low carbon economy, or novel diagnostics impacting on personalized medicine and genetic disease treatment. Cross-cutting technologies in the field, such

[8] Note that Ninkasi was the Goddess of Beer for the Sumerians and Babylonians.

as PCR, novel sequencing tools, bioinformatics, transcriptomics and epigenetics, are in the vanguard of biotechnological progress leading to an ever-increasing breadth of applications." It is foreseen that Biotechnology will deliver solutions to unimagined problems, providing food security, health and well-being to mankind for centuries to come.

Today "Biotechnology uses substances, materials or extracts derived from living cells, employing 22 million Europeans in a €1.5T€ endeavor, being the premier global economic growth opportunity this century" [18]. As rightfully highlighted in [70], "in actuality, everything from the food we eat to the fuel we use to heat our homes or power our automobiles is likely to have an element of biotechnology associated with it."

Biotechnology is defined by the OECD as "the application of science and technology to living organisms, as well as parts, products and models thereof, to alter living or non-living materials for the production of knowledge, goods and services" [65]. This definition includes all modern biotechnology but also many traditional biotechnological activities.

Due to the master narrative of the European Union (EU) [25], Biotechnology and ICT represent the EDTs of the Knowledge-Based Bio-Economy (KBBE). KBBE – using Biotechnology and ICT – aims at the "sustainable production and conversion of biomass, for a range of food, health, fiber and industrial products and energy", where "renewable biomass encompasses any biological material to be used as raw material."[9]

Synthetic biology [32], [48], [12], [1] is a revolutionary scientific and engineering discipline in the field of biotechnology that is expected to have considerable impact on the society. The ultimate goal of synthetic biology is the rational design of artificial biological systems that provide novel useful functions. Biological systems can be "new biological components, such as enzymes genetic circuits, and cells or the redesign of existing biological systems" [79]. Synthetic biology is a cross-functional discipline. It "borrows wet-lab techniques from genetic engineering, modeling techniques from systems biology, and design concepts from electrical and control engineering" [32].

The technology stack of KBBE is not static, but strongly influenced by dynamic dependencies and interactions between the participating technologies in the process of forming the KBBE. These dynamics are perhaps best captured (qualitatively) and illustrated using a metaphor of a "green triple-helix", where "green" stands for environmental consciousness and "triple-helix" visualizes the interdependency of biotechnology, synthetic biology, and ICT as the helical strands. The concept of the triple-helix model was introduced Etzkowitz and Leydesdorff in the context of the KBE [24], [23] to analyze the dynamic relationships between universities (research), industry and government.

[9] THE EUROPEAN BIOECONOMY IN 2030. Delivering Sustainable Growth by addressing the Grand Societal Challenges: `http://www.epsoweb.org/file/560`

5 KBBE ICT Challenges

The KBBE produces a wide range of challenges to ICT. However, this paper focuses on what the author believes are the potential highest impact areas of ICT for the KBBE in general and synthetic biology in particular along the dimensions of education, research, and industry applications.

5.1 Education – Computational Thinking (CT)

Given the KBBE requirements, what are the perceived educational gaps for biologists in general and synthetic biologists in particular? Based on the professional literature, the vote seems to be unambiguous. It's the missing proficiency in mathematics (whatever this really means). In fact, the importance of mathematics in biology has been discussed for a long time. The problem is traced back to Charles Darwin and a remark in his autobiography [17]: "I have deeply regretted that I did not proceed far enough at least to understand something of the great leading principles of mathematics, for men thus endowed seem to have an extra sense. But I do believe that I should ever have succeeded beyond very low grade." In 1920, distinguished evolutionary biologist Ellis Michael stated: "The problem cries for solution. What is to be done? The only way out of this dilemma that I can conceive is to insist upon proficiency in mathematics, just as we now insist upon proficiency in English, as prerequisite to a major in biology. This means reform." [52]. But nearly 150 years after Darwin's comment, the situation doesn't seem to have changed substantially to the better [50]. Bialek complains in [5] that "with significant exceptions (e.g. population genetics, structural biology, and some areas of neuroscience), biologists today rarely achieve mathematical competence beyond elementary calculus and maybe a few statistical formulae." If so, what are the reasons?

From the author's perspective, there seem to be only two ways to explain this situation. First, mathematics is too difficult for biologists. "Math anxiety" [3] of biologists is often mentioned in this context. Second, mathematics is not as important for the daily work of biologists as it has been considered. The first explanation appears to be a daring hypothesis; even so it is supported by Darwin's statement. Indeed, a recent study indicated that math anxiety is not a predictor of mathematics achievements [11], nor is there any obvious reason to assume that biologists refuse mathematics per se . Therefore, the second hypothesis is probably much closer to reality. Back to the above quote of Charles Darwin and in light of the hypothesis of the second option, Robertson states that "the case for Darwin's theory of evolution by natural selection would have been strengthened had he been mathematician enough to recognize Mendelian ratios, but this scarcely diminishes his monumental achievement" [60]. In Layman's terms, for biologists to have solid mathematical skills might be useful and important, but certainly not to the discussed extreme.

Then it becomes the question, whether there is any other educational gap for biologists, which when overcome, would make a substantial contribution to the advancement of synthetic biology and consequently the KBBE. The author proposes computational thinking (CT) as an educational area which would be most beneficial to biologists and synthetic biologists alike.

The notion of CT was introduced by Jeanette Wing in an influential Communications of the ACM Viewpoint paper a few years ago [72]. In this paper Wing frames CT out as: "Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. Computational thinking includes a range of mental tools that reflect the breadth of the field of computer science." She argues that computational thinking "will be a fundamental skill used by everyone in the world by the middle of the 21st Century … just like reading, writing, and arithmetic" [72]. Wing's paper has received a considerable attention "in the last few years, resulting in numerous workshops, conference panels and online discussions" [41]. A major part of these efforts as well as various discussions in journal and conference papers has focused on finding a robust and universally acceptable definition of CT, e.g. [10], [20], [29], [36]. This work has been so far not (very) successful, e.g. [31]. Having said this, it needs to be mentioned that there are two definitions that received more widespread support. The first definition is offered by the Computer Science Teachers Association[10] [4][11]. The second definition is provided by Google's Exploring Computational Thinking[12] initiative. In [69] Weinberg provides a thorough literature review of CT research and discusses the above mentioned definitions of CT in more detail.

For this paper a different approach has been chosen. Rather than adding to the already substantial body of definition papers, here the focus is on CT as a thought framework. This is in line with the pragmatic approach towards computational thinking. For example, Henderson positions CT as "a universal metaphor of reasoning used by both mankind and machines. From this perspective it has the potential to be a comprehensive umbrella for capturing the intrinsic nature of computing and conveying this in an understandable way to students and the general public" [37]. Hu promotes a focus on the practical aspects of CT (rather than on defining it) [41]. In fact, Wing herself provides a description of CT with a focus on the implied thought process: "Computational thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information processing agent."[13] Note that the information processing agent can be any type of "machine … (human or computer, virtual or physical)."[14]

If one interprets computational thinking as a distinguished mode of thought, then providing a normative definition takes a backseat. With a reference to the report of the Committee for the Workshops on Computational Thinking of National Research Council of the U.S. [55], Yeh summarizes practical CT characteristics as follows [77]: abstractions and automations of abstractions, precise representations, systematic analysis, and repetitive refinements. In particular, Yeh describes these characteristics as follows:

[10] `http://csta.acm.org/`

[11] See also `http://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf`

[12] `http://www.google.com/edu/computational-thinking/`

[13] `http://www.cs.cmu.edu/ourcs/presentations/ct.pdf`

[14] `http://www-cgi.cs.cmu.edu/afs/cs/usr/wing/www/CT_at_CMU.pdf`

- Abstractions and automations of abstractions refer to “the ability to manage complex situations by generating abstractions and maintaining the relationships among them”;
- In order to generate abstractions, one need to have precise representations, that is “formal representations that reflect … cognitive processes and structures (discerning aspects of the situation)”;
- The systematic analysis characteristic of CT denotes the fact, that CT enables “to generate hypotheses and search for a plausible solution systematically”;
- The repetitive refinements points to the iterative nature of the problem solving process, where the current situation is consistently evaluated “against … previous experience or … prediction until the best solution is reached”.

This explains the broad applicability of CT. As Wing positions it: “Computational thinking will influence everyone in every field of endeavor” [73].

From the above it is obvious that CT is a central topic of computer science education aka ICT education. It is also clear that CT aims to reach beyond computer science. For this, synthetic biology is a perfect candidate. This holds especially true in light of the math proficiency discussion in biology. Given that biologists do think in hierarchies of abstractions and processes, it should be relatively straightforward to familiarize them with the notion of CT, which in turn opens the door to naturally introduce more advanced concepts from computer science such as algorithms, data structures, control structures, logic and semantics etc.

The way for computer science aka ICT education to achieve this is to start off with the native abstractions of synthetic biology, such as promoters, ribosome-binding site, terminators, etc. and proceed by mapping these abstractions to computational abstractions. This could be starting point for ICT education to establish CT as the primary problem solving approach in synthetic biology. The positive impact on the future progress of synthetic biology and therefore the KBBE can hardly be overestimated. Imagine an ICT professional and a synthetic biologist talking the same language based on shared understanding of CT! So, the call goes out to ICT professionals especially those how are concerned with education to team up with synthetic biologists to establish CT as the primary thought framework for problem solving in synthetic biology.

5.2 Research – Complexity Engineering (CE)

The French philosopher and mathematician Rene Descartes formalized almost 400 years ago the concept of reductionism. The reductionist principle states that the whole is the sum of its parts. Since then reductionism has become the cornerstone method of science and engineering alike. Indeed, it can be said that classical engineering is founded on the reductionist approach. Systems are built using components with well-defined behaviors and interfaces. The components do not exhibit any uncertainty as long as they are operated within their specifications, and neither does the system. No surprises! Uncertainty is the nemesis of classical engineering.

Biological systems are intrinsically complex [46] and exhibit significant uncertainty. For example, "even within a clonal population of genetically identical cells there can be cell-to-cell variability for many parameters" [53]. Simply speaking, a group of seemingly identical cells under the same environmental conditions exhibit different behaviors. Surprise! Briefly, uncertainty can be categorized into two types, namely aleatory uncertainty and epistemic uncertainty, e.g. [38], [56]. Biological systems exhibit both types of uncertainty. Aleatory uncertainty is the consequence of the inherent randomness of nature and is therefore irreducible. In biology, aleatory uncertainty might be, for example, the consequence of intrinsic gene expression noise [22]. Epistemic uncertainty is the consequence of incomplete knowledge and is therefore reducible. Current knowledge about biochemical cell reactions is made explicit by mechanistic mathematical models. However, a lack of knowledge about molecular composition, interaction patterns, missing kinetic parameters, noisy data etc. manifest the presence of epistemic uncertainty in biological systems [45]. Therefore, when dealing with biological systems, one should be always prepared for surprises.

Complex systems display behavioral manifestations that are completely inexplicable by applying the reductionist principle. They are governed by a principle that shall be called complexity principle. This principle states that the whole is more than the sum of its parts. By the means of an argumentum e contrario, it follows that reducing a complex system to its components results in an irretrievable loss of what characterizes the system in the first place [35].

It should be noted, that distinguished biologists did realize the limitations of the reductionist approach. For example, in the early 20th century Ludwig von Bertalanffy and Paul A. Weiss, questioned the validity of the reductionism approach to biology and "began to develop their own perspectives towards a system theory of life" [21]. This work culminated in Bertalanffy's seminal General System Theory [68]. In 1997, a number of biologists and other researchers met to discuss the future of the reductionist approach in biology. The meeting relativized the singular importance of reductionism in biology "to just one of the many tools biologists will need to fathom the mysteries of nature" [71]. The latest step towards a system level understanding in biology is the emergent field of systems biology [74]. Systems biology aims to obtain a system level (holistic) understanding of living biological systems by analyzing their behavior and interrelationships [62]. As Kitano, one of the early promoters of systems biology, explains: "To understand biology at the system level, we must examine the structure and dynamics of cellular and organismal function, rather than the characteristics of isolated parts of a cell or organism. [47]

None, of the above mentioned efforts has led to a reliable framework for engineering synthetic biological systems and were, perhaps, not supposed to anyway. However, they do make it very clear that biological systems are complex systems and need to be dealt with appropriately. Therefore, synthetic biology needs an engineering framework that allows for the engineering of biological systems that are inevitable complex. Currently, the main stream of synthetic biology is still enforcing classical engineering methods (despite knowing better?). One reason is certainly the set of preferable attributes of a classical engineered system, which are: stability, predictability, reliability, transparency, controllability [54]. By trying to squeeze synthetic biology into the classical engineering framework, one hopes that the above attributes are maintained in the engineered biological system. The central question is then: If one is to engineer a biological system, how far can one get with classical engineering approaches, that is with "unmitigated reductionism" [67], in meaningful economic time frames[15]? Realistically, one cannot expect to get very far, at least in respect to the stated ambitious objectives of synthetic biology. Therefore, engineering needs to be extended beyond its classical limitations, such as the inability to account for emerging phenomena. The emerging discipline of complexity engineering (CE) [9] aims to provide methods, tools and techniques to not only cope with complexity, but to use it "to the engineer's favor" [28]. Note, that CE is not meant to replace classical engineering, but rather to complement it. Therefore, classical engineering should be used whenever it is adequate.

From the above, the imperative conclusion is that CE as a discipline is indispensable for synthetic biology to succeed. Any synthetic biology engineering framework must support CE approaches. However, even so that CE is the right answer, it needs to be noted that the discipline is still in its infancy. Given this, the provision of a workable CE framework for synthetic biology constitutes, perhaps, the biggest challenge for synthetic biology to progress. If not addressed timely and appropriately this will become most likely a major barrier to the development and growth of KBBE.

This is where ICT can help in a substantive and significant way. Compared to other engineering disciplines, ICT is, perhaps, the one which has been most exposed to engineered complex systems, even so these systems were originally designed to comply with classical engineering rules and any complex behavior was unintended. Examples of such systems are the World-Wide Web, the Internet (routers and domains), software, etc. ICT had to learn the hard way how to cope with the complexity issues that stem from the sole application of classical engineering methods. Now, ICT is in the process of paving the way of CE. Software engineering is a prime example. In [75] Jay Xiong describes the situation in software engineering as follows: "… unfortunately, the old-established software engineering paradigm based on linear thinking and simplistic science (which assumes that the whole is nothing but the sum of its parts …) itself is incomplete, unreliable, and blind in responding to changes without support for preventing side-effects." In appreciating the need for CE approaches, he proposes the Nonlinear Software Paradigm, a CE method, rooted in complexity science [76].

Given the supposed leading role of ICT in establishing CE as a proper discipline and given the indispensability of such a discipline for synthetic biology, ICT research is perfectly positioned to drive synthetic biology forward. ICT is the EDT for the KBE. By signing up to the synthetic biology CE agenda, ICT can make again a substantial impact on economic and societal progress (transformation of the KBE to the KBBE). So, the call goes out to ICT researchers and practitioners alike to participate in the endeavor of developing a workable CE practice for synthetic biology.

[15] Meaningful economic time frames are dictated by time-to-market and profitability requirements.

5.3 Industry Applications – Bio-Design Automation (BDA)

Design Automation (DA) technologies are an indispensable part of any (hi-tech) industry. Given the highly complicated nature of modern product designs, it is simply impossible to perform the required design process without DA technologies. DA technologies provide the foundation for Computer-Aided Design (CAD), where CAD refers to the process of doing design with the aid of computers (or more precisely with the aid of ICTs incl. DA technologies). Prominent examples for DA technologies are Architecture-Engineering-Construction (AEC), Mechanical Computer-Aided Design (MCAD), and Electronic Design Automation (EDA). In essence, DA frees design from the laboratory by allowing for virtual product realizations before any physical implementation takes place.

Take the example of aircraft design. The A300 was the first aircraft developed by Airbus Industries. It made its maiden flight in 1972. The first production model entered into service 1974[16]. Mas describes in [51] the increasing utilization of CAD for the design of Airbus aircrafts. In fact, the A300 design started in 1969 with paper drawings! At this stage, aircraft design was still to a large extent heuristic and experiment-driven (laboratory-based design). The A320 design was the first using DA technologies (initially to facilitate the drawings production process). This was the crystallization point for MCAD to become indispensable for aircraft design at Airbus Industries. Now MCAD is used for the complete functional design of all Airbus aircrafts. To design an aircraft like the A400M which "is composed of 700 000 parts (excluding standard components and joint elements)" [51] would be simply impossible without MCAD.

Take the example of the design of integrated electronic circuits. After the fabrication of first junction transistor in 1951, Jack Kilby built in 1958 the first integrated circuit ever (an oscillator) with both active and passive components fabricated from semiconductor material. All this work was done in the laboratory. Perhaps, there might have been some paper drawings. It took until 1966 before the first EDA technology was developed by IBM[17]. From then on EDA developed rapidly according to the productivity requirements dictated by the Moore's Law[18]. Quite recently, the billion transistor landmark has been crossed. The Intel 62-Core Xeon Phi has a transistor count of 5 billion. NVidia's Graphical Processing Unit (GPU) GK110 Kepler [19] holds the world record with about 7 billion transistors. Note, that today's design of high-end computer chips is often limited by the available EDA capabilities. In other words, the number of available transistors (due to the advances

[16] `http://en.wikipedia.org/wiki/Airbus`

[17] More details about the timeline of semiconductors and EDA can be found at the respected Computer History Museum webpage `http://www.computerhistory.org/semiconductor/`

[18] According to Moore's Law, the transistor count of the integrated circuits doubles every two years. The transistor count of a device is the number of transistors in the device. Transistor count is the most common measure of integrated circuit complexity.

[19] `http://www.nvidia.com/content/PDF/kepler/NVIDIA-Kepler-GK110-Architecture-Whitepaper.pdf`

in semiconductor process technologies) grows much faster than the ability to design them meaningfully. This "has become the greatest threat to the growth of semiconductor industry" [2], a phenomenon which is well known as design productivity gap or design productivity crisis[20].

Bio-Design Automation (BDA) denotes the DA technologies for synthetic biology. Given a large amount of experience in DA technologies, one could optimistically assume that developing BDA is a straightforward task. But it is not at all. The underlying fundamental reasons where discussed in the previous Complexity Engineering section. Designing computer chips or aircrafts is highly complicated, but still can be tackled with classical engineering approaches. In short, it is design under certainty. Designing biological systems is not only highly complicated but also complex. It it's design under uncertainty. Designing computer chips or aircrafts is model-driven design, a process which builds on the availability of pre-constructed, validated and verified predictive models. The situation in synthetic biology is completely different. Designing biological systems is at the stage of laboratory design (heuristic and experiment driven). Because of the intrinsic complexity of biological systems, it is more than unlikely that BDA can take the same route like EDA, MCAD, etc. In synthetic biology, the predictive power of models at any abstraction level is usually very limited and will continue to be limited for some foreseeable time. As Hassoun describes it [34], "Biological systems and electronics are different. Biological cells grow changing size and functional characteristics, evolve through mutations, die, reproduce, exhibit robustness to environmental variables such as nutrients, temperature, and PH balance, and adapt due to redundant components." Therefore, BDA needs to start afresh, finding its own way. In [32] an interested reader finds an overview of the current (very early) stage of BDA technology. So, the call goes out to ICT researchers, practitioners and industrialists alike to participate in the endeavor of creating BDA technologies and businesses. In whatever time frame, the grand vision is to eventually free the design of synthetic biological systems from the laboratory. The first pragmatics step, however, is to provide BDA technologies and businesses that support and complement the current laboratory based design. And this is already a challenge on its own.

6 Concluding Remarks

Perhaps the only way for a human society to be brought into harmony with Nature is to follow a biological way of its development. This is why researchers, policy-makers, and industry practitioners herald the emerging Knowledge-Based Bio-Economy as the "Next Big Thing" in societal progress. However, the society has made just a few first steps on a thousand mile road to its flourishing and many cross-cutting challenges still have to be overcome.

This discussion paper took the basics from the economic thought and explained that the KBBE, like any other kind of economy, has to be built upon the pragmatic economic principles. Those include the account for resource scarcity, economy defining resources and technologies. Knowledge and bio-resources have been identified as the economy defining resources for KBBE.

[20] International Technology Roadmap for Semiconductors:
`http://www.itrs.net/home.html`

Further, the paper argued that having the pool of economy defining resources at hand is not sufficient. Economy defining technologies are also required to make the economy effective. Such technologies for KBBE are ICT and biotechnology. The principal enabling technology of biotechnology is Synthetic Biology, which is also heavily based on ICT and knowledge technologies.

The paper was concluded with the discussion of high-level and long term ICT-related challenges in the context of KBBE. The analysis was structured along the dimensions of education, research, and industry applications. For the facet of education the challenge of empowering synthetic biologists with computational thinking was discussed. In the context of KBBE-related research the major challenge was seen in the complexity of bio-systems. Therefore, complexity engineering and its use in KBBE research and development was offered as an important focal area. For industrial applications, the inspiration had been taken from the developments in Engineering Design Automation for mechanical and electronic designs. Based on that, the emerging Bio-Design Automation was offered as a way that enables Synthetic Biology designs and solutions leaving the lab and gaining industrial strength.

References

1. Andrianantoandro, E., Basu, S., Karig, D.K., et al.: Synthetic biology: new engineering rules for an emerging discipline. Mol. Syst. Biol. 2 (2006)
2. Anuradha, A.K.: Power, Interconnect and Complexity Crises in Fuiture VLSI: From a Designer's Point of View. International Journal of Electrical Engineering & Technology 3, 210–222 (2012)
3. Ashcraft, M.H., Moore, A.M.: Mathematics anxiety and the affective drop in performance. Journal of Psychoeducational Assessment 27, 197–205 (2009)
4. Barr, V., Stephenson, C.: Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? ACM Inroads 2, 48–54 (2011)
5. Bialek, W., Botstein, D.: Introductory Science and Mathematics Education for 21st-Century Biologists. Science 303, 788–790 (2004)
6. Bishop, M.: Essential Economics: An A to Z Guide. Bloomberg Press (2009)
7. Boisot, M.H.: Is your firm a creative destroyer? Competitive learning and knowledge flows in the technological strategies of firms. Research Policy 24, 489–506 (1995)
8. Bresnahan, T.F., Trajtenberg, M.: General purpose technologies "Engines of growth"? J. of Econometrics 65, 83–108 (1995)
9. Buchli, J., Santini, C.C.: Complexity engineering, harnessing emergent phenomena as opportunities for engineering. Reports of the santa fe institute's complex systems summer school (2005)
10. Bundy, A.: Computational thinking is pervasive. J. Scientific and Practical Computing 1, 67–69 (2007)
11. Capraro, R.M., Capraro, M.M.: Commonality analysis: Understanding variance contributions to overall canonical correlation effects of attitude toward mathematics on geometry achievement. Multiple Linear Regression Viewpoints 27, 16–23 (2001)
12. Cheng, A.A., Lu, T.K.: Synthetic biology: an emerging engineering discipline. Annual Review of Biomedical Engineering 14, 155–178 (2012)
13. Clarke, T.: The knowledge economy. Education+Training 43, 189–196 (2001)

14. Coil, C.: Teaching Tools for the 21st Century. Pieces of Learning (2005)
15. Commission, E.A.: Engineering Criteria 2000. ABET, Baltimore (1998)
16. Conrad, M.: Molecular Automata. In: Physics and Mathematics of the Nervous System, pp. 419–430. Springer (1974)
17. Darwin, C.: The Autobiography of Charles Darwin. Barnes & Noble Publishing (2005)
18. Gartland, K.M.A., Bruschi, F., Dundar, M., Gahan, P.B., Viola Magni, M.P., Akbarova, Y.: Progress Towards the 'Golden Age' of Biotechnology. Current Opinion in Biotechnology 24(suppl. 1), S6–S13 (2013)
19. Dede, C.: Comparing frameworks for 21st century skills. 21st Century Skills: Rethinking How Students Learn, 51–76 (2010)
20. Denning, P.J.: The profession of IT: Beyond computational thinking. Commun. ACM 52, 28–30 (2009)
21. Drack, M., Apfalter, W., Pouvreau, D.: On the making of a system theory of life: Paul A Weiss and Ludwig von Bertalanffy's conceptual connection. The Quarterly Review of Biology 82, 349 (2007)
22. Elowitz, M.B., Levine, A.J., Siggia, E.D., et al.: Stochastic Gene Expression in a Single Cell. Science 297, 1183–1186 (2002)
23. Etzkowitz, H., Leydesdorff, L.: The dynamics of innovation: from National Systems and "Mode 2" to a Triple Helix of university–industry–government relations. Research Policy 29, 109–123 (2000)
24. Etzkowitz, H., Leydesdorff, L.: The Triple Helix-University-Industry-Government Relations: A Laboratory for Knowledge Based Economic Development. Easst Review 14, 14–19 (1995)
25. European Commission. Life Sciences and Biotechnology-a Strategy for Europe, Third Progress Report annd Future Orientation, Report from the Commission to the European Parliament, the Council, the Committee of the Regions, and the Economic and Social Committee. Office for Official Publications of the European Communities, Luxembourg (2005)
26. Feasey, P.: A brief history of beer. Food Australia 59, 191 (2007)
27. Moreno, F., Gil, Ó.S.: Blondie Economics, p. 73. Innisfree Editorial Ltd. (2012)
28. Frei, R., Serugendo, G.D.M.: Concepts in complexity engineering. International Journal of Bio-Inspired Computation 3, 123–139 (2011)
29. Garcia, D.D., Lewis, C.M., Dougherty, J.P., et al.: If ____________, you might be a computational thinker? In: Proceedings of the 41st ACM Technical Symposium on Computer Science Education, pp. 263–264. ACM, Milwaukee (2010)
30. Gegas, V.C., Nazari, A., Griffiths, S., et al.: A genetic framework for grain size and shape variation in wheat. The Plant Cell Online 22, 1046–1056 (2010)
31. Gouws, L.A., Bradshaw, K., Wentworth, P.: Computational thinking in educational activities: an evaluation of the educational game light-bot. In: Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education, pp. 10–15. ACM, Canterbury (2013)
32. Habibi, N., Hashim, S.Z.M., Rodriguez, C.A., et al.: The emerging field of synthetic biology: A review. In: 4th International Conference Intelligent and Advanced Systems 2012, pp. 160–164 (2012)
33. Hallauer, A.R.: Corn breeding. Iowa State Research Farm Progress Reports, paper 549 (2008)
34. Hassoun, S.: Genetic/bio design automation for (re-)engineering biological systems. In: Design, Automation & Test in Europe Conference & Exhibition 2012, pp. 242–247 (2012)

35. Hawe, P., Shiell, A., Riley, T.: Complex interventions: how “out of control” can a randomised controlled trial be? British Medical Journal 328, 1561 (2004)
36. Henderson, P.B.: Ubiquitous Computational Thinking. Computer 42, 100–102 (2009)
37. Henderson, P.B., Cortina, T.J., Wing, J.M.: Computational thinking. In: Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education, pp. 195–196. ACM, Covington (2007)
38. Hoffman, F.O., Hammonds, J.S.: Propagation of Uncertainty in Risk Assessments: The Need to Distinguish Between Uncertainty Due to Lack of Knowledge and Uncertainty Due to Variability. Risk Analysis 14, 707–712 (1994)
39. Holley, S.E.: Nano Revolution–Big Impact: How Emerging Nanotechnologies Will Change the Future of Education and Industry in America (and More Specifically in Oklahoma) An Abbreviated Account (2009)
40. Holmgren, Å.J.: A framework for vulnerability assessment of electric power systems. Critical Infrastructure, pp. 31–55. Springer (2007)
41. Hu, C.: Computational thinking: what it might mean and what we might do about it. In: Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education, pp. 223–227. ACM, Darmstadt (2011)
42. Jackson, D.A., Symons, R.H., Berg, P.: Biochemical method for inserting new genetic information into DNA of Simian Virus 40: circular SV40 DNA molecules containing lambda phage genes and the galactose operon of Escherichia coli. Proceedings of the National Academy of Sciences 69, 2904–2909 (1972)
43. Jonas, D.A., Elmadfa, I., Engel, K.H., et al.: Safety Considerations of DNA in Food. Annals of Nutrition and Metabolism 45, 235–254 (2001)
44. Jovanovic, B., Rousseau, P.L.: Chapter 18 General Purpose Technologies. In: Philippe, A., Steven, N.D. (eds.) Handbook of Economic Growth, pp. 1181–1224. Elsevier (2005)
45. Kaltenbach, H.-M., Dimopoulos, S., Stelling, J.: Systems analysis of cellular networks under uncertainty. FEBS Letters 583, 3923–3930 (2009)
46. Kitano, H.: Computational systems biology. Nature 420, 206–210 (2002)
47. Kitano, H.: Systems Biology: A Brief Overview. Science 295, 1662–1664 (2002)
48. Kitney, R., Freemont, P.: Synthetic biology – the state of play. FEBS Letters 586, 2029–2036 (2012)
49. Lemke, C.: enGauge 21st Century Skills: Digital Literacies for a Digital Age (2002)
50. Madlung, A., Bremer, M., Himelblau, E., et al.: A Study Assessing the Potential of Negative Effects in Interdisciplinary Math–Biology Instruction. CBE-Life Sciences Education 10, 43–54 (2011)
51. Mas, F., Menéndez, J., Oliva, M., et al.: Collaborative Engineering: an Airbus case study. In: 5th Manufacturing Engineering Society International Conference (2013)
52. Michael, E.L.: Marine ecology and the coefficient of association: a plea in behalf of quantitative biology. Journal of Ecology 8, 54–59 (1920)
53. Milo, R., Jorgensen, P., Moran, U., et al.: BioNumbers – the database of key numbers in molecular and cell biology. Nucleic Acids Research 38, D750–D753 (2010)
54. Mina, A.A., Braha, D., Bar-Yam, Y.: Complex Engineered Systems: A New Paradigm. In: Braha, D., Minai, A.A., Bar-Yam, Y. (eds.) Complex Engineered Systems, pp. 1–21. Springer, Heidelberg (2006)
55. National Research Council Report of a Workshop on the Scope and Nature of Computational Thinking. National Academies Press (2010)
56. Oberkampf, W.L., Helton, J.C., Joslyn, C.A., et al.: Challenge problems: uncertainty in system response given uncertain parameters. Reliability Engineering & System Safety 85, 11–19 (2004)

57. OECD The Measurement of Scientific and Technological Activities Oslo Manual: Guidelines for Collecting and Interpreting Innovation Data. OECD Publishing (2005)
58. Partnership for 21st Century Skills Framework for 21st Century Learning (2013), http://www.p21.org
59. Robbins, L.: An essay on the nature and significance of economic science. Macmillian, London (1932)
60. Robertson, M.: Biologists who count*. Journal of Biology 8, 34 (2009)
61. Rosenberg, N., Trajtenberg, M.: A general-purpose technology at work: The Corliss steam engine in the late-nineteenth-century United States. The Journal of Economic History 64, 61–99 (2004)
62. Sauro, H.M., Harel, D., Uhrmacher, A., et al.: Challenges for modeling and simulation methods in systems biology. In: Proceedings of the Winter Simulation Conference, WSC 2006, pp. 1720–1730. IEEE Press (2006)
63. Towne, H.R.: The Engineer as an Economist. Transactions of the American Society of Mechanical Engineers VII, 428–431 (1886)
64. Towne, H.R.: Industrial engineering. Ingeniería Industrial), discurso pronunciado en la Universidad de Purdue el 24 (1905)
65. Van Beuzekom, B., Arundel, A.: OECD Biotechnology Statistics 2009 (2009)
66. Van Moorst, H.: The multi-function polis – Incubator of elitism. Japanese Studies 10, 26–35 (1990)
67. Van Regenmortel, M.H.: Reductionism and complexity in molecular biology. EMBO Reports 5, 1016 (2004)
68. Von Bertalanffy, L.: An outline of general system theory. British Journal for the Philosophy of Science 1, 134–165 (1950)
69. Weinberg, A.E.: Computational Thinking: An investigation of the existinig scholarship and research. Colorado State University (2013)
70. Wells, J.: Defining biotechnology. The Technology Teacher 54, 11–14 (1995)
71. Williams, N.: Biologists cut reductionist approach down to size. Science 277, 476–477 (1997)
72. Wing, J.M.: Computational thinking. Communications of the ACM 49, 33–35 (2006)
73. Wing, J.M.: Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 366, 3717–3725 (2008)
74. Wolkenhauer, O.: Systems biology: The reincarnation of systems theory applied in biology? Briefings in Bioinformatics 2, 258–270 (2001)
75. Xiong, J.: A Complete Revolution in Software Engineering Based on Complexity Science. Software Engineering Research and Practice, pp. 109–115 (2009)
76. Xiong, J.: New software engineering paradigm based on complexity science: an introduction to NSE. Springer (2011)
77. Yeh, K.-C., Ying, X., Ke, F.: Teaching computational thinking to non-computing majors using spreadsheet functions. In: Frontiers in Education Conference 2011, pp. F3J-1–F3J-5 (2011)
78. Ziegert, A.L.: The role of personality temperament and student learning in principles of economics: Further evidence. The Journal of Economic Education 31, 307–322 (2000)
79. Keasling, J.D.: Synthetic Biology for Synthetic Chemistry. ACS Chemical Biology 3(1), 64–76 (2008)

Theory of Interaction, Insertion Modeling, and Cognitive Architectures

Alexander Letichevsky

Glushkov Institute of Cybernetics, Academy of Sciences of Ukraine
let@cyfra.net

Abstract. The challenge of creating a real-life computational equivalent of the human mind is now attracting the attention of many scientific groups from different areas of cybernetics and Artificial Intelligence such as computational neuroscience, cognitive science, biologically inspired cognitive architectures etc. The paper presents a new cognitive architecture based on insertion modeling, which is one of the paradigms of a general theory of interaction, and a basis for multiagent systems development. This architecture is now under development on the base of Insertion Modeling System IMS, developed in Glushkov Institute of Cybernetics. Intelligent agent developed on a base of this architecture is represented as a multilevel insertion machine which realizes itself as a high level insertion environment. It has a center to evaluate the success of its behavior which is a special type agent that can observe the interaction of a system with external environment. The main goal of a system is achieving maximum success repeated. As an agent this machine is inserted into its external environment and has the means to interact with it. The internal environment of intelligent cognitive agent creates and develops its own model and the model of external environment. If the external environment contains other agents, they can be modeled by internal environment which creates corresponding machines and interprets those machines using corresponding drivers, comparing the behaviors of models and external agents.

Keywords: Cognitive Architecture, Agent Based System, Distributed Artificial Intelligence, Reasoning, Formal Methods, Simulation.

1 Introduction

General theory of interaction is a theory of information interaction in complex distributed multi-agent systems. It has a long history. Contemporary part of this history can be considered as starting from neuro networks of McCulloch-Pitts [27]. The model of neuro nets caused the appearance of abstract automata theory, a theory which helps study the behavior and interaction of evolving systems independently of their structure. The Kleene-Glushkov algebra [8, 16] is the main tool for the description of the behaviors of finite state systems. Automata theory originally concentrated on the study of analyses and synthesis problems, generalization of finite state automata and complexity. Interaction in explicit form appeared only in 70s as a general theory of interacting information processes. It includes the CCS (Calculus of Communicated Processes) [28, 29] and the π-calculus of R. Milner [30], CSP (Communicated Sequential Processes) of T. Hoare [13], ACP (Algebra of Communicated Processes) [4] and many other various branches of these basic theories. Now all these calculi and algebras are the basis for modern research in this area. Fairly complete survey of the classical process theory is presented in the Handbook of Process Algebras [5], published in 2001. At the same time the models of concurrent computation began to develop under the influence of the practical queries of parallel programming. The most abstract models are Petri nets [33], actor model [12], as well as the widespread ideas of parallel object-oriented programming.

Insertion modeling is a trend that is developing over the last decade as an approach to a general theory of interaction of agents and environments in complex distributed multi-agent systems. The first works in this direction have been published in the middle of 90s [7, 18, 19]. In these studies, a model of interaction between agents and environments based on the notion of insertion function and the algebra of behaviors (similar to some kind of process algebra) has been proposed. Insertion modeling generalizes the most of traditional theories of interaction. Each of them can be obtained by specializing of insertion function which is a perameter of insertion model. Moreover in one model we can use several insertion functions (multilevel environments). This makes it possible to combine defferent theories of interaction.

The paradigm shift from computing to interaction was extensively discussed in computer science that time, and our work was in some sense a response to this trend. But the real roots of the insertion modeling should be sought even earlier, in a model of interacting of control and operational automata, proposed by V. Glushkov back in the 60s [9, 10] to describe the structure of computers. In the 70s the algebraic abstraction of this model were studied in the theory of discrete processors and provided a number of important results on the problem of equivalence of programs, their equivalent transformations and optimization. Macroconveyor models of parallel computing, which were investigated in 80s years [14], even more close to the model of interaction of agents and environments. In these models, the processes corresponding to the parallel processors can be considered as agents that interact in an environment of distributed data structures.

In recent years, insertion modeling has been applied to the development of systems for the verification of requirement specifications of distributed interacting systems [3, 15, 22–24]. The system VRS, developed in order from Motorola, has been successfully applied to verify the requirements and specifications in the field of telecommunication systems, embedded systems, and real-time systems. A new insertion modeling system IMS [20], which has been developed in the Glushkov Institute of Cybernetics of the National Academy of Sciences of Ukraine, is intended to extend the area of insertion modeling applications. We found many common features of the tools used in software development area based on formal methods and techniques used in biologically inspired cognitive architectures. This gives us hope to introduce some new ideas to the development of this subject domain.

This paper presents the main principles of insertion modeling and the conception of cognitive architecture based on insertion modeling. To understand the formal part of the paper reader must be familiar with the concepts of labeled transition system, bisimilarity and basic notions of general process theory. The mathematical foundation of insertion modeling is presented in [21].

2 The Basic Principles

Insertion modeling deals with the construction of models and study the interaction of agents and environments in complex distributed multi-agent systems. Informally, the basic principles of the paradigm of insertion modeling can be formulated as follows.

1. The world is a hierarchy of environments and agents inserted into them.
2. Environments and agents are entities evolving in time and having observable behaviors.
3. Insertion of agent into environment changes the behavior of environment and produces new environment which is in general ready for the insertion of new agents.
4. Environments as agents can be inserted into higher level environment.
5. New agents can be inserted from external environment as well as from internal agents (environments).
6. Agents and environments can model another agents and environments on the different levels of abstraction.

All these principles can be formalized in terms of transition systems, behavior algebras, and insertion functions. This formalization can be used as high level abstractions of biological entities needed for computer modeling of human mind.

The first and the second principles are commonly used in information modeling of different kinds of systems, for example as in object oriented or agent programming. They are also resembling to M. Minsky's approach of the society of mind [31].

The third principle is clear intuitively, but has a special refinement in insertion modeling. We treat *agents* as transition systems with states considered up to bisimilarity (or up to behavior, which is the same). The *type of an agent* is the set of actions it can perform. The term *action* we use as a synonym of label for transitions, and it can denote signals or messages to send, events in which an agent can participate etc. This is the most general notion of agent which must be distinguished from more specific notion of autonomous or intellectual agent in AI.

Transition system consists of states and transitions that connect states. Transitions are labeled by *actions* (signals, events, instructions, statements etc.). Transition systems are evolving in time changing their states, and actions are observable symbolic structures used for communication. We use the well-known notation $s \xrightarrow{a} s'$ to express the fact that transition system can evolve from the state s to s' performing action a. Usually transition systems are nondeterministic and there can be several transitions coming from the same state even labeled by the same action. If we abstract from the structure of states and concentrate only on (branching) sequences of observable actions we obtain the state equivalence called *bisimilarity* (originated from [32] and [25], exact definition can be found in [21]). Bisimilar states generate the same behavior of transition systems.

Environment by definition is an agent that possesses the *insertion function.* Given the state of environment s and the state of agent u, insertion function computes the new state of environment which is denoted as $s[u]$. Note that we consider states up to bisimilarity and if we have some representation of behaviors, the behaviors of environment and agent can be used as states. The state $s[u]$ is a state of environment and we can use insertion function to insert a new agent v into environment $s[u] : (s[u])[v] = s[u, v]$. Repeating this construction we can obtain the state of environment $s[u_1, u_2, \ldots]$ with several agents inserted into it. Insertion function can be considered as an operator over the states of environment, and if the states are identified with behaviors, then the insertion of a new agent changes the behavior of environment.

Environment is an agent with insertion function, so if we forget the insertion function, then environment can be inserted as agent into a higher level environment and we can obtain hierarchical structure like

$$s[s_1[u_{11}, u_{12}, \ldots]_{E_1}, s_2[u_{21}, u_{22}, \ldots]_{E_2}, \ldots]_E$$

Here notation $s[u_1, u_2, \ldots]_E$ explicitly shows the environment E to which the state s belongs (environment indexes can be omitted if they are known from the context). This refines the fourth principle.

Environment is an agent which can be inserted into external environment and having agents inserted into this environment. The evolution of agents can be defined by the rules for transitions. The rules

$$p \to t[v, s[u]] \xrightarrow{a} t[s[u, v]]$$

and

$$q \to s[t[u, v]] \xrightarrow{a} s[t[u], v]$$

can be used for the illustration of the 5-th principle. The first rule asserts that under condition p an agent v can move from higher level environment t to lower level environment s, and the second rule shows how an agent v can move from the low level environment t to the higher level environment s.

We consider the creation and manipulation of the models of external and internal environments as the main property of cognitive processes of intellectual agent. Formalization of this property in terms of insertion modeling supports the 6-th principle and will be considered later.

Cognitive architecture will be constructed as a multilevel insertion environment. Below we shall define the main kinds of blocks used for construction of cognitive architecture. They are *local description unites* and *insertion machines.*

3 Multilevel Environments

To represent behaviors of transition systems we use *behavior algebras* (a kind of process algebra). Behavior algebra is defined by the set of actions and the set of behaviors (processes). It has two operations and termination constants. Operations are prefixing $a.u$ (a - action, u - behavior) and nondeterministic choice $u + v$ (u and v - behaviors). Termination constants are successful termination Δ, deadlock 0, and undefined behavior $\perp$. It has also approximation relation $\sqsubseteq$, which is a partial order with minimal element $\perp$, and is used for constructing a complete algebra with fixed point theorem. To define infinite behaviors we use equations in behavior algebra. These equations have the form of recursive definitions $u_i = F_i(u_1, u_2, \ldots), i = 1, 2, \ldots$ and define left hand side functions as the components of a minimal fixed point. Left hand sides of these definitions can depend on parameters $u_i(x) = F_i(u, x)$ of different types. In complete behavior algebra each behavior has a representation (normal form)

$$u = \sum_{i \in I} a_i.u_i + \varepsilon_u$$

which is defined uniquely (up to commutativity and associativity of nondeterministic choice), if all $a_i.u_i$ are different (ε_u is a termination constant).

The system of behaviors

$$u_i = \sum_{j \in J(i)} a_{ij}.u_j + \varepsilon_i, i \in I, J(i) \subseteq I$$

can be considered as a system of recursive definitions. If it is finite, then behaviors u_i are called of *finitely-defined.* Such behaviors can be identified with the states of finite tranzition system with the transition relation $u_i \xrightarrow{a_{ij}} u_j, j \in J(i) \subseteq I$ and the set of final states $U_\Delta = \{u_i | \varepsilon_i = \Delta\}$.

The *type of environment* is defined by two action sets: the set of environment actions and the set of agent actions. The last defines the type of agents which can be inserted into this environment: if the set of agent actions is included in the set of agent actions of environment then this agent can be inserted into this environment. This relation is called *compatibility* relation between agents and environments (agent is compatible with environment if it can be inserted into this environment). *Multilevel environment* is a family of environments with distinguished the most external environment. The compatibility relation on the set of environments defines a directed graph and we demand for multilevel environment that the outermost environment would be reachable from any environment of the family in this graph.

Extend the algebra of behaviors, adding to it the insertion functions, considered as binary operations on the corresponding sets. Now the expressions considered at the beginning of previous section can be considered as an algebraic expressions of corresponding multisorted algebra. There are no special identities for insertion function common to all extended behavior algebras. The algebraic properties of this function as operation are consequences of the definitions of insertion functions in multilevel environments.

4 Main Properties of Insertion Functions

We shall consider only those environments, for which there is an identity $e[u, \Delta] = e[u]$. The state e of environment is called *indecomposable* if from $e = e'[u']$ it follows that $e' = e, u' = \Delta$. The set of indecomposable states of environment is called its *kernel*. Indecomposable state of environment corresponds to the state before inserting into it any agents. If an environment has empty kernel (all states are decomposable) then there is an infinite number of agents originally inserted into this environment. Environment is called finitely-decomposable if each of its state can be represented in the form $e[u_1, ..., u_m]$ with indecomposable e. Hereinafter we will consider only finitely-decomposable environments unless otherwise stated.

One-step insertion is defined by the rule

$$\frac{e \xrightarrow{a} e', u \xrightarrow{b} u'}{e[u] \xrightarrow{c} e'[u']} P(a, b, c)$$

Here - $P(a, b, c)$ is allowing predicate. The rule is applicable only in the case when the allowing predicate is true.

For finitely-decomposable environment in which many agents are inserted one should use the derivative rule:

$$\frac{e[u_1, ..., u_{m-1}] \xrightarrow{c} e'[u'_1, ..., u'_{m-1}], u_m \xrightarrow{b} u'_m}{e[u_1, ..., u_m] \xrightarrow{d} e'[u'_1, ..., u'_m]} P(c, b, d)$$

where e - decomposable state of the environment.

Another form of one-step insertion rule:

$$P(a, b, c) \Rightarrow (a.e' + e'')[b.u' + u''] \xrightarrow{c} e'[u']$$

One-step rule can be understood as follows. Let the agent "wants" to perform the action b (one of possible). Then environment permits this action, if it has two actions a and c, such that $P(a, b, c)$ is true. In this case, the transition depends only on what the environment can do in one step.

The more general rule can be formulated, if the behavior of an environment is added to the allowing predicate: $P(e, a, b, c)$. But that would ensure the continuity of the insertion function, we need to require the continuity of the allowing predicate: *it should only depend on the behavior of finite prefix of behavior* e. Such a rule is called the rule of *head insertion*. Finally, there is the more general case obtained if allowing predicate depends on the agent's behavior: $P(e, u, a, b, c)$. Such an insertion is called *look-ahead* insertion.

Consideration of the rules is not enough to fully describe the insertion function. Additional rules are required for the termination states of the environment and the agents as well as for a non-deterministic choice. Examples of such rules can be identities $0[u] = 0, \Delta[u] = \Delta, e[\Delta] = e$, which can completed by the identity $e[0] = 0$ for indecomposable state of the environment e.

Insertion function is called *additive* if

$$e[u+v]=e[u]+e[v],(e+f)[u]=e[u]+f[u]$$

Additivity of insertion function may mean that environment as well as agent make their choices independently. It should be noted that if there is no rule on which a transition from $e[u]\neq \Delta, \perp$ then $e[u]=0$. In addition to the additive, in practice there are commutative insertions with identities $e[u,v]=e[v,u]$ and, as a special case of commutative parallel insertion: $e[u,v]=e[u||v]$. Here $()||()$ denotes the parallel composition of behaviors.

The above classes of functions do not exhaust the insertion functions that are interesting from the point of view of modeling of cognitive processes. The most important are insertion functions, the calculation of which is a combination of the unfolding of recursive definitions and rewriting techniques in the extended behavior algebra. Such function are considered in more details in [21]

At a given moment of time an agent belongs (is inserted) to only one environment. But if the type of an agent is compatible with the type of another environment it can move to this environment. Such movements can be described by the following types of rules:

$$\frac{u \xrightarrow{moveup\ E} u'}{E[F[u,v],w] \xrightarrow{moveup(F\to E)} E[F[v],u',w]} P_1(E,F,u,moveup(E))$$

moving from internal to external environment;

$$\frac{u \xrightarrow{movedn\ F} u'}{E[F[v],u,w] \xrightarrow{movedn(E\to F)} E[F[u',v],w]} P_2(E,F,u,movedn(F))$$

moving from external environment to internal one;

$$\frac{u \xrightarrow{moveto\ G} u'}{E[F[u,v],G[w]] \xrightarrow{moveto(F\to G)} E[F[v],G[u',w]]} P_3(E,F,u,moveto(F))$$

moving to another environment on the same level. In all cases allowing conditions must include the compatibility conditions for corresponding agents and environments. The rules above define the property of a system called *mobility* and underlies the calculus of mobile ambients of Luca Cardelli [6].

5 Attributed Environments

For many practical applications, the concept of the environment and the agent is too abstract, because it ignores the structure of environments and agents. Furthermore, at the transition to the terminal state information is lost about environment state if it has the structure. For example, let nonterminal states be functions. The terminal states Δ or 0 are not functions, but the information about the function of a state before terminating may be interesting.

All necessary information can, of course, be passed through actions, but it often looks unnatural and awkward. In order to solve this problem, in [24] the concept of attributed transition system was introduced. The difference between labeled and attributed transition systems is that attributed transition systems have not only labeled transitions but also labeled states. The behavior algebra for attributed system is different in that behavior can be marked with the attribute markups. For this purpose, another operation, the operation of marking $\varphi : u$ is introduced except of prefixing where u – behavior, φ – marking. Now, all the information about the state of the environment needed for observer can be transmitted through its markup. In particular now there can be many termination constants corresponding to a successful termination, or deadlock, since they may have different markings.

Attributed environments are based on some logical framework. This framework includes a set of types (integer, real, enumerated, symbolic, behavioral, etc.), interpreted in some data domains, it includes symbols to denote constants of these domains, and a set of typed functions and predicate symbols. Some of these symbols are interpreted (e.g., arithmetic operations and inequalities, equality for all types, etc.). Uninterpreted function and predicate symbols are called *attributes.* Uninterpreted function symbols of arity 0 are called *simple attributes*, the other - *functional attributes* (uninterpreted predicate symbol is considered as a functional one with the binary values domain $0, 1$). Function symbols are used to define data structures such as arrays, lists, trees.

The *basic logical language* is built over a logical framework of attributed environment. Usually it is the first order language. If necessary, it may include some of the modalities of temporal or fuzzy logic if they simplify the reasoning in specific application domains. *Attribute expression* is a simple attribute or an expression of the form $f(t_1, ..., t_n)$, where f is a functional attribute of arity n, $t_1, ..., t_n$ - already constructed attribute expressions or constants of suitable types. If all expressions are constants, then the attribute expression is called a *constant expression.*

In general, the kernel of attributed environment consists of the formulas of basic language. Attributed environments are divided into two classes: the *concrete* and *symbolic* ones.

Indecomposable state of concrete attributed environment is a formula of the form $t_1 = a_1 \wedge ... \wedge t_n = a_n$, where - $t_1, ..., t_n$ are different constant attribute expressions, $a_1, ..., a_n$ are constants. Typically, such a formula is represented as a partial mapping with domain $\{t_1, ..., t_n\}$ and range equal to the set of all constants. Concrete attribute environments are useful to formalize the operational semantics of programming languages and specification language of software systems. Environment state is a memory state (data structures, objects, network structures, channels, etc.). Programs are agents inserted into this environment. For parallel programming languages interaction is implemented via shared memory or message passing. When interacting via shared memory parallel composition of behaviors (asynchronous or synchronized) plays a major role. Message passing involves the use of special data structures in the environment for the organization of interaction. Actions of the agents ultimately reduced down to the checking conditions (*check* α) or assignments ($x := t$ where x is an attribute expression, t – algebraic expression of basic language). The corresponding rules are:

$$\frac{u \xrightarrow{check\ \alpha} u'}{e[u] \xrightarrow{check\ \alpha} e[u']} e \models \alpha$$

$$\frac{u \xrightarrow{x:=t} u'}{e[u] \xrightarrow{x:=t} e'[u']}$$

where e' is the result of the assignment to the state e. Allowing condition for the assignments in concrete attributed environments may impose a restriction on the uniqueness of the values for the left side arguments (for the functional attributes) and the uniqueness of the values of the expressions of the right-hand side of assignments.

Program constructions (different kinds of loops, branching, etc.) in most cases can be just modeled by recursive equations in behavior algebra. For example,

$$if\ \alpha\ thenP\ else\ Q = check\ \alpha + check\ \neg\alpha.Q$$

$$while\ \alpha\ do\ P = if\ \alpha\ then(P; while\ alpha\ do\ P) else\ \Delta$$

Indecomposable states of symbolic environments are formulas of the basic language of arbitrary form. Concrete and symbolic states of attributed environment of a type $e[u_1, u_2, ...]$are also considered as formulas. Namely, it is assumed that all inserted agents have unique names (this may be, for example, their attribute marks). Further, among the attributes of the base language there is a functional attribute "state", defined on the names of agents and taking values in their behaviors. If $m_1, m_2, ...$ are the names of agents $u_1, u_2, ...$, then $F[u_1, u_2, ...] \Leftrightarrow F \wedge (state(m_1) = u_1) \wedge (state(m_2) = u_2) \wedge ....$

6 Local Description Units

The most famous ways of specification of interacting systems (software, technical, biological) are in the area of temporal, dynamic and some other types of modal logics. In most cases, they are used when a detailed model of the system is already known and the problem of checking properties of the system, given in a logical form is being solved (model checking).

Another way of system specification is to describe the local behavioral properties of a system by means of *local description units.* In mathematical form, local description units describe the properties of transition relation of a transition system, and when the system is presented in the form of composition of agents and environments, we are talking about the local properties of the insertion function.

We will consider the local properties of the system represented as a concrete or symbolic attributed environment. Local description unit of the attributed environment is a formula $\forall x(\alpha \rightarrow < P > \beta)$, where x is a list of (typed) parameters,

α and β – the basic language formulas, P – process (finite behavior of the specified system). The formula α is called the precondition, and the formula β – the postcondition of local description unit. Both the conditions and the behavior of the unit may depend on the parameters. Local description unit can be considered as a temporal logic formula that expresses the fact that, if (for suitable values of parameters) the system satisfies the precondition, the behavior P can be activated and, after its successful termination, the new state will satisfy the postcondition. It is not difficult to see the analogy between the Hoare triples (formulas of dynamic logic) and the local description units. Another analogy are production rule systems that are widely used in the description of the behavior of artificial intelligence and expert systems.

Postconditions can contain assignments $x := y$ where x is an attribute expression and y is an algebraic expression (term). This assignment is considered as a simple temporal logic statement which asserts that a new value of attribute expression x is equal to an old value of an algebraic expression y. Therefore the local description unit $\forall v(\alpha \rightarrow < P > (x := y) \wedge \beta)$ is equivalet to $\forall (u, v)(\alpha \wedge y = u \rightarrow < P > (x = u) \wedge \beta)$.

Local properties which are used in the input language of VRS system to represent the distributed models of interacting systems are called basic protocols. They are expressed in the basic language of the system VRS, and to represent the behaviors of the system MSC diagrams are used. Typically, the basic protocols are defined independently of each other and are not a priori connected by any sequencing relation that would determine the order of their application. Therefore, the semantics of the specifications by means of the basic protocols are not obvious. The study of semantics underlying protocols several approaches were developed for the construction of such a semantics. These approaches are described in [24, 25].

Local description units are the main units of knowledge representation in cognitive architecture. A set of local description units can be used for the definition of transitions of environments. In this case they can be considered as procedural knowledge units. Logic knowledge can be represented as environment with the states representing the current knowledge, and the local description units corresponding to the rules of inference in corresponding calculus. Local description units can be applied in forward and backward modes. Forward mode can be used for the generating of new knowledge, backward mode – for answering queries.

7 Network Environments

Agents in *network environments* posess lists of input and output attributes corresponding to channels that are used to receive and transmit signals of certain types. We call such agents as *network agents.* Actions of network agents are the pairs x/y, where x is a list of the values of input attributes, and y is a set of values of output attributes. The transition relation $s \xrightarrow{x/y} s'$ is interpreted as follows: if the agent is in a state s and the list of its input attributes is assigned to x, then the list of its output attributes can be assigned to y, and the agent transit to s'. In an obvious way a network agent can be considered as non-deterministic automaton, and its attributes can be considered as its input and output channels.

The state of network environment (network) defines the relationship between the values of input and output attributes of network agents inserted into this environment. Local properties of insertion for an attribute network environment might look like this:

$$\forall(x, y, u, v, r')(\alpha \rightarrow < P(r, r') > \beta)$$

where

$$\alpha = (connect(m, r, x, y) \wedge (state(m) = u) \wedge u \xrightarrow{x/y} v \wedge newval(m, v, r, r'))$$

and

$$\beta = (r := r') \wedge (state(m) = v)$$

Here r is some list of simple attributes of environment and agents. It includes those attributes that determine a list x of values of input attributes of agent m and attributes that may change at the next moment of time. The predicate *connect* defines the possible values of the input and output attributes of agent m, and the predicate *newval* defines new values r' of attributes r at the next moment. The process $P(r, r')$ makes observable for external environment certain attributes from the list r and their new values r'. Local description defines the functioning of agent m in the environment in which it is inserted.

The predicates *connect* and *newval* determine the structure of the network environment. They may be considered as attributes of functional type, and in this case the network can change its structure, for example, when new agent is inserted into this network. The rules of corresponding changes can also be expressed in terms of local properties.

If in a set of attributes of the network environment the input and output attributes of the network are distinguished, the network can be considered as a network agent, and it can be inserted into the network environment of the higher level. In this case the local properties must be such that they permit arbitrary changes of the values of input attributes of the network, and the network actions must be the pairs x/y, where x is the list of values of input attributes of the network and y is the list of values of its output attributes.

The most typical representatives of the class of network environments are automata networks. In this case, network agents are automata, and their input and output attributes are identified with their input and output channels, respectively. Likewise, the input and output attributes of a network are identified with its input and output channels. To determine the network structure the intermediate network attributes are used which are called internal channels, ports, nodes, etc., and predicates *connect* and *newval* are defined by the equalities that identify some nodes of the network with input and output channels. In particular, each internal channel is identified with one of the output channels of an automaton or a network input channel, and with several input channels of network automata and output channels of a network.

The foregoing local description of the functioning of the network agent in a network environment corresponds to the case when in a moment of discrete time only one network agent can change its state. To describe more complex transitions one can use the rules that are obtained as a composition of simple rules.

Another important class of systems that can be represented as network environments are biological or artificial neural networks. In this case, the network agent of the lower level is neuron. The input attributes of the model of the biological neuron are dendrites, and a single output attribute of the network agent is an axon. Network attributes which are used for the organization of the connections between neurons are synapses. An artificial neuron is simply organized automata that computes the sum of its input values and produces on its output the value according to its insertion function. The range of the input and output attributes of a neuron is either a set of real numbers, or a finite set of numeric signals. Neurons and neural networks can be considered as agents operating in continuous or discrete time. To switch from continuous to discrete time it is convenient to represent the values of input and output attributes as functions defined on real intervals. The weights of synapse connections are considered as environment attributes which are used to implement learning algorithms, self learning, pattern recognition, etc. All these algorithms are implemented by environment.

8 Insertion Machines

Other construction blocks for cognitive architecture are insertion machines intended for implementation of insertion environments. The input of insertion machine is the description of a multilevel environment (a model of an environment) and its initial state, an output depends on the goal that is put to this machine.

Multilevel environments are represented in cognitive architecture by means of *environment descriptions* for different levels and a set of local description units for insertion functions. Environment description contains the signature of environment that includes types of attributes, types of inserted agents, and also the description of sets of environment and agent actions. Local description units used for the definition of insertion function are organized as a knowledge base with special data structures providing efficient access to the needed descriptions and history of their use.

To implement multilevel environment different kinds of insertion machines are used. But all of them have the general architecture represented on the Fig.1. Insertion machine simulates the behavior of input model by trversing the tree of this behavior implementing depth first search or breadth first search.Symbols 'Curr' and 'Next' denote the current and the next state of environment.

Three main components of insertion machine are *model driver* (MD), *behavior unfolder* (Unf), and *interactor* (Intr). Model driver is a component which controls the machine traversal along the behavior tree of a model. The state of a model is represented as a text in the input language of insertion machine and is considered as an algebraic expression. The input language includes the recursive

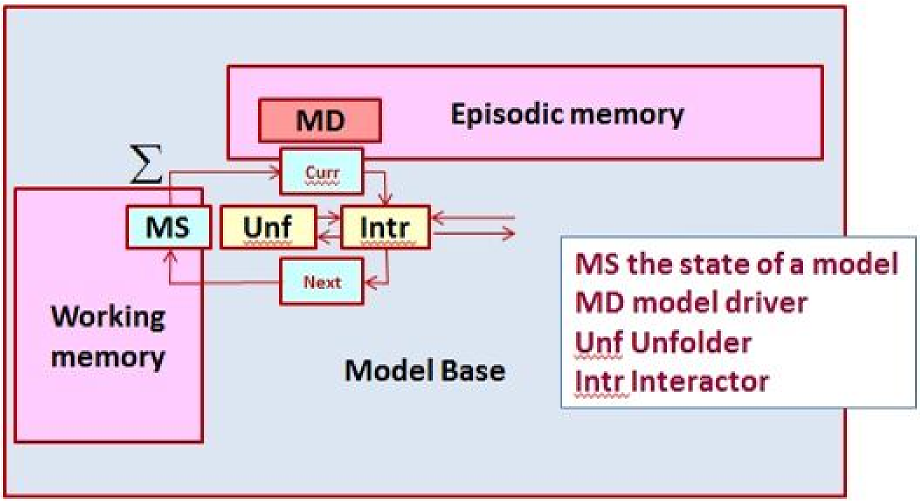


Fig. 1. Architecture of Insertion Machine

definitions of agent behaviors, the notation for insertion function, and possibly some compositions for environment states. Before computing the insertion function, the state of a system must be represented in the form $s[u_1, u_2, \ldots]$. This functionality is performed by agent behavior unfolder. To make a movement (transition), the state of environment must be reduced to the normal form

$$E = \sum_{i \in I} a_i.E_i + \varepsilon$$

where a_i are actions, E_i are environment states, ε is a termination constant. This functionality is performed by the module *environment interactor*. It computes the insertion function calling recursively if it is necessary the agent behavior unfolder. After reducing to normal form model driver can select the direction of movement $i \in I$ and perform transition $E \xrightarrow{a_i} E_i$.

Simple insertion machine reproduces the behavior of a model inserted into it, without changing anything in this model, but only determining it in the case of nondeterministic choice. More complex cognitive machines will be considered later.

Two kinds of insertion machines are distinguished: *real time* or *interactive* and *analytical* insertion machines. The first ones are functioning in the real or virtual environment, interacting with it in the real or virtual time. Analytical machines intended for model analysis, investigation of its properties, solving problems etc. The drivers for two kinds of machines correspondingly are also divided into interactive and analytical drivers.

Interactive driver after normalizing the state of environment must select exactly one alternative and perform the action specified as a prefix of this alternative. Insertion machine with interactive driver operates as an agent inserted into external environment with insertion function defining the laws of functioning of this environment. On the other hand, if the agent model inserted into machine is identified with the agent itself, then interactive driver can be regarded as an environment for this agent. If the driver is used as an interactive environment for the sequence of agents, they can be combined with a composition, such as parallel, and get an agent whose behavior will be brought to the normal form using the definition of the composition.

It is natural to assume that the external environment does not know the structure of the machine and inserted model. Therefore, the interaction of machine and environment takes place at the level of the head insertion:

$$\frac{E \xrightarrow{x} E', M[u] \xrightarrow{y} M'[u']}{E[M[u]] \xrightarrow{c} E'[M'[u']]} P(E, x, e, c)$$

Here E - environment, M - Insertion Machine, P - allowing predicate.

Cognitive interactive driver performs the nondeterministic choice of the current action, the most complicated and misterious function of the human mind. Therfore the cognitive module driver can be organized in a quite complicated way. It can have criteria of successful functioning in external environment, accumulate the information about its past in episodic memory, develop the models of external environment, uses some learning algorithms to improve the strategy of selecting actions and increase the level of successful functioning. The simple interactive driver possesses entire model and it can unfold the behavior tree of this model unrestrictedly, predicting the future. In other words the insertion function for interactive driver is a look ahead function:

$$\frac{M \xrightarrow{z} M', u \xrightarrow{a} u'}{M[u] \xrightarrow{b} M'[u']} Q(M, u, z, a, b)$$

Allowing predicate Q defines a method for selecting the direction of movement in the case of a non-deterministic behavior of the agent. The simplest mechanism of choice of direction for interactive driver - is the use of priorities. Suppose that a continuous function $pr : F(A) \times A \times F(A) \to U$ of priority is defined on the set of agents that are inserted in an environment, where U is a partially ordered set. Then, after reduction to a normal form $E = \sum_{i \in I} a_i.E_i + \varepsilon$ the behavior of a model, interactive driver selects in a non-deterministic way an index $i \in I$ such that $pr(u, a_i, u_i)$ takes one of the maximum values. In case of several maximum values some additional mechanisms of determinization should be applied, including the use of probability criteria.

In more complex cases, the driver of insertion machine can gain experience and change the priority function in accordance with certain criteria for the success of its functioning.

Analytical insertion machine, as opposed to interactive, can consider various alternatives for the choice of actions, returning to the choice points and considering different paths in the tree of system behavior. The model of a system can also include a model of the environment for this system. In general, the analytical driver searches for states with given properties (reachability of goal states) or states in which the desired properties are violated. The search result will be one or more traces, confirming the reachability of goal states.

External environment of analytical machine can be represented by a user who interacts with a machine, formulating the problems and controlling machine activity. Analytical machines, enriched with logic and deductive tools are used for symbolic simulation of systems.

VRS system in addition to the insertion machine using a specialized language for describing the input requirements and specifications, provides a means for static model analysis, generation of test sets, code generation and a number of other tools that support the development of distributed interactive systems.

Insertion modeling system IMS, developed at Glushkov Institute of Cybernetics is a development environment for insertion machines of various types, including cognitive architectures. It contains in its structure a number of prototypes of insertion machines for models with different types of insertion functions and tools for their development. Basic programming system for IMS is the algebraic programming system APS, which provides flexibility and efficiency in the development of new insertion machines. The article [Letichevsky12], describes insertion machine for evidential programming. It is focused on the verification of imperative annotated programs using various programming languages.

9 Cognitive Architecture ICAR

Like well-known cognitive architectures such as Soar [17], ACT-R [2] or many other from the list of BICA society [34] insertion cognitive architecture ICAR is an environment for construction of cognitive agents. As with any complex system, system architecture ICAR has a multilayer structure represented on the Fig.2. At the heart of the pyramid representing the architecture of ICAR system, is a programming language. Currently it is a C++ on Windows or Unix, but because the upper layers remain unchanged, the transition to other platforms, including platforms with real parallelism, is technically not a major problem.

The upper level of the system ICAR is an insertion modeling system IMS. It is used as a tool for the development of insertion machines, and one of these machines is an intelligent agent ICAR. If it does not lead to ambiguity, we use the same name for the cognitive architecture and the intelligent agent, which is created on the base of this architecture.

IMS system is implemented on the base of algebraic programming system APS [1], which can be characterized by the following key words: the system of rewriting rules and rewriting strategies. The next layers are the language APLAN - the input language of APS with interpreter and compiler, and APLANC - a library of basic functions of the APS system in C++.

The technical basis of the agent ICAR is a system of insertion machines that implements the behavior of the various subsystems of the agent ICAR and is organized as a multi level environment. To represent the models implemented by insertion machines of ICAR, the system uses recursive definitions in the corresponding behavior algebras for low-level subsystems, local description modules, attribute and network environments.

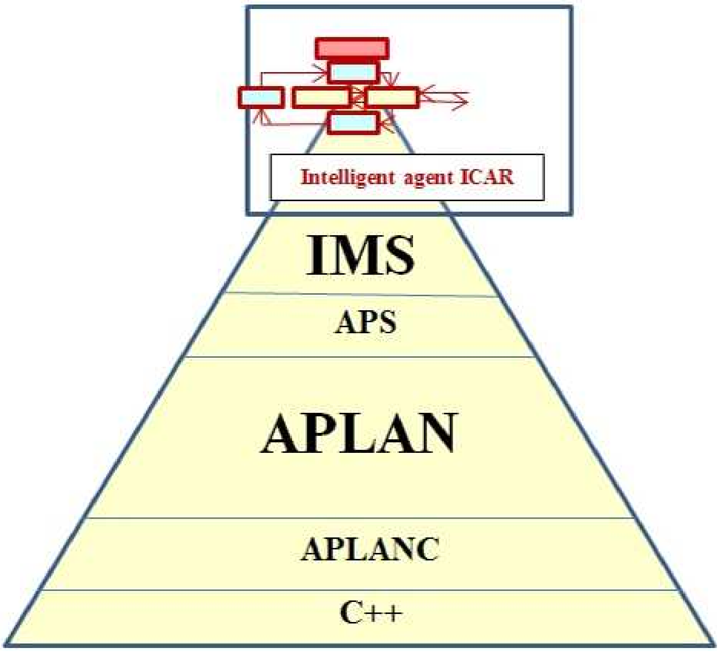


Fig. 2. Cognitive architecture ICAR

The intelligent agent ICAR by definition is a cognitive real time insertion machine, which is aware of itself, has a center of evaluating the success of its behavior and tends to repeatedly achieve maximum success. As an agent this machine is inserted into its external environment and has the means to interact with it.

Cognitive insertion machine is different from the simple one so that it can transform a model inserted into it, refining this model based on the experience and perfecting it according to its own criteria, which may also change over time. The initial model, inserted into the environment of cognitive machine is a model of the external environment, into which the machine itself is inserted. This model includes the model of cognitive machine itself. In the case of biological systems, this model is transmitted genetically, in the case of a computer model, it contains the initial knowledge base that the developer provides.

Cognitive driver of ICAR forms its internal environment, and is designed as a multi level environment filled with machines and cognitive models inserted into these machines. The structure of an intelligent agent ICAR is shown in Fig.3.

Level Self is the top level of the internal environment of an agent ICAR. It provides control of all agents, inserted into it, and through monitoring of all subsystems at lower levels. At this level there is also a model driver of the *main model* of the outside world, with the inserted ICAR model. ICAR continuously monitors all the changes occurring in the external environment, analyzing the incoming imaginative and language information, and generates a response in the

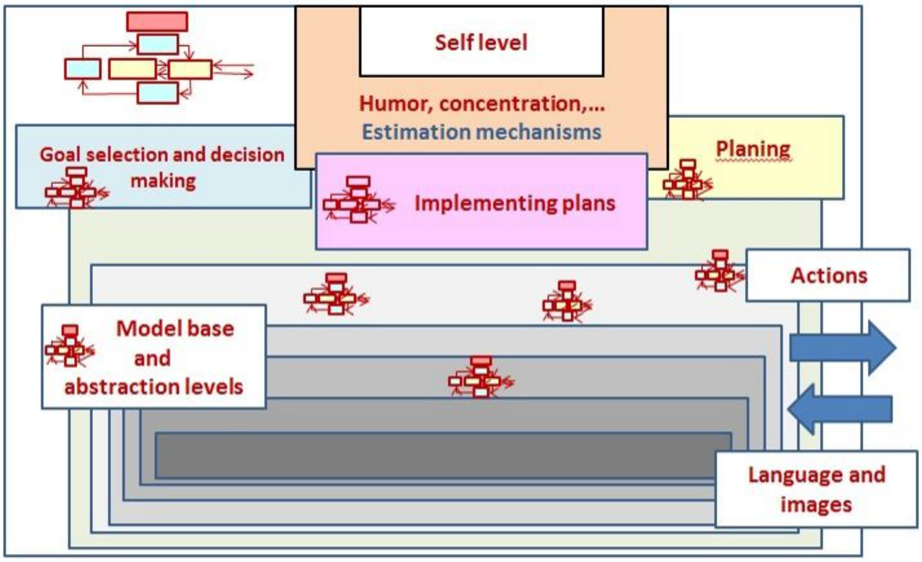


Fig. 3. Insertion cognitive architecture

form of their actions (spatial moving, the generation of linguistic or imaginative information) using the main model. "ICAR aware of itself" means that it distinguishes between its controlled and managed internal environment with agents occupying it, with one hand, and the external environment on the other hand. The models of emotions and feelings are realized on the upper level, to the extent that they are necessary to ensure effective and efficient interaction with the people from external environment of ICAR.

On the upper level there are also mechanisms for evaluating the success of ICAR activities. Key success criteria laid developers of ICAR, but they depend on the success criteria of functioning component of the lower levels that apply to different types of activities and cannot be selected in advance. These criteria should be formed through experience, learning, and self learning by the cognitive driver.

The scope of activity defines the interaction of ICAR with the external environment. It is a part of the main model and includes traditional components of autonomous agents, as the choice of goals and subgoals, planning and implementing the plan, the interaction with other agents, inserted into the environment. The inputs of ICAR (its input attributes) are continuously perceive imaging information about the location of an agent in a three-dimensional (virtual or real) space and the locations of other items and agents in it, as well as symbolic (language) information from the users and other agents from the environment. ICAR outputs (output attributes) are perceived by the external environment and lead to a change in its state in accordance with the insertion function (movement, interaction with objects and agents, transmission and perception of symbolic information by external environment).

Language (symbolic) Communication. ICAR has to communicate with users in natural language. ICAR receives symbolic information about the external environment, its state and the events taking place in it in addition to the imaging information. Perception and understanding of this information forms a change in the main model. The main model includes in addition to the behavioral aspects of a memory of the past, forecasts of possible future as well as the logical model of external and internal environment of ICAR. On the other hand, ICAR statements that it generates, communicating with its interlocutors, refer to the description of its internal state that is directly or indirectly related to the state of its main model. However, the states of all models have descriptions in the language of insertion modeling. Thus, the language of insertion modeling serves as the semantic basis of language spoken ICAR, and the language of communication can also be used as a meta-language for the language of insertion modeling.

Memory. The main types of memory used by ICAR are a working memory, episodic memory, semantic memory, and the main models of different levels of abstraction and insertion machines. All these types of memory are connected via main model. The state of a model is located in the working memory. Usually it is a formula of basic language. It also provides information from semantic memory required for the unfolding of recursive function definitions and computing insertion function. It is defined by functional and predicate symbols used in the formula, as well as the consequences that define the relevant local descriptions of the functions necessary for the computing of insertion function. The result of the machine is a symbolic trace transmitted to episodic memory. Here is the past experience of the functioning of the basic model. Lots of traces generated by the basic model in the process of interaction with the environment, are presented at different levels of abstraction and combined into data structures that are supplied with additional information that characterizes the probabilistic assessment of the type of Bayesian networks, as well as information about the successes and failures. All this information is used to predict the behavior of the environment and the creation of preferences when choosing a continuation of the main model. It can be used by Jeff Hawkins' ideas on the structure of the hierarchical temporal memory [11], he proposed as a model of the neocortex.

The main part of the memory of ICAR consists of a base of insertion machines and models. Here, in particular are machines for solving of various classes of problems and corresponding models. If necessary, they can be activated by the main model to be inserted into its environment to participate in the interaction with the environment. Another example - the machine to work with language constructs that perceive and recognize the linguistic knowledge, generate answers to questions, provide an explanation in the spirit of expert systems in natural language. There may be a machines to participate in the games, works in spatial conditions, etc.

Learning. The main objective of ICAR is to improve its main models in accordance with the criteria set by its creator. The main role in this process is played by the methods of learning and selflearning. In addition to well-known methods that are adapted to the insertion modeling, it is assumed by explaining the different training techniques and problem-solving techniques, rules and guidelines of behavior in different situations. Such lessons should be provided with appropriate exercises and examples. By performing these exercises, ICAR must provide appropriate abstraction that would expand the scope of applicability of the appropriate methods for the situations which are similar but different from the context that accompanies the lessons.

ICAR must also own methods of changing the main model and specialized machines and models on the basis of experience, design new useful models and machines.

10 Conclusions

The paper presents the basics of insertion modeling, a new paradigm in the general theory of interaction. At present, the main application of insertion simulation is verification of the technical systems, but the potential of this trend is not confined to this area.

Insertion modeling paradigm is being used to build a new cognitive architecture ICAR. Development of ICAR is still at the level of requirements capturing, some of requirements are presented in the paper. The basic building units are ICAR insertion machines and models. The construction and improvement of these models and machines is the basis of reasonable activity of an intelligent agent.

The description of cognitive architecture in the last section is a very tentative reflection of our far goals. The nearer goals include the further development of our system of proving program correctness [26], communication in natural language, and living in virtual reality. As a zero approximation of ICAR the insertion modeling system [20] is used.

References

1. APS and IMS systems, `http://www.apsystem.org.ua`
2. Anderson, J.R., Lebiere, C.: The Atomic Components of Thought. Lawrence Erlbaum Associates, Mahwah (1998)
3. Baranov, S., Jervis, C., Kotlyarov, V., Letichevsky, A., Weigert, T.: Leveraging UML to deliver correct telecom applications in UML for Real. In: Lavagno, L., Martin, G., Selic, B. (eds.) Design of Embedded Real-Time Systems, pp. 323–342. Kluwer Academic Publishers (2003)
4. Bergstra, J.A., Klop, J.W.: Process algebra for synchronous communications. Information and Control 60(1/3), 109–137 (1984)
5. Bergstra, J.A., Ponce, A., Smolka, S.A. (eds.): Handbook of Process Algebra. North-Holland (2001)
6. Cardelli, L., Gordon, A.D.: Mobile ambients. In: Nivat, M. (ed.) FOSSACS 1998. LNCS, vol. 1378, pp. 140–155. Springer, Heidelberg (1998)

7. Gilbert, D.R., Letichevsky, A.A.: A universal interpreter for nondeterministic concurrent programming languages. In: Gabbrielli, M. (ed.) Fifth Compulog Network Area Meeting on Language Design and Semantic Analysis Methods (September 1996)
8. Glushkov, V.M.: On an algorithm of abstract automata synthesis. Ukrainian Mthematical Journalvol 12(2), 147–156 (1960)
9. Glushkov, V.M.: Automata theory and questions of design structure of digital machines. Cybernetics 1, 3–11 (1965)
10. Glushkov, V.M., Letichevsky, A.A.: Theory of algorithms and descrete processors. In: Tou, J.T. (ed.) Advances in Information Systems Science., vol. 1, pp. 1–58. Plenum Press (1969)
11. Hawkins, J., Blakeslee, S.: On intelligence. Times Books, Henry Holt and Co. (2005)
12. Hewitt, C., Bishop, P., Steiger, R.: A Universal Modular Actor Formalism for Artificial Intelligence. IJCA (1973)
13. Hoare, C.A.R.: Communicating Sequential Processes. Prentice Hall (1985)
14. Kapitonova, J., Letichevsky, A.: Mathematical theory of computational systems design. Moscow, Science, 295 (1988) (in Russian)
15. Kapitonova, J., Letichevsky, A., Volkov, V., Weigert, T.: Validation of Embedded Systems. In: Zurawski, R. (ed.) The Embedded Systems Handbook. CRC Press, Miami (2005)
16. Kleene, S.C.: Representation of Events in Nerve Nets and Finite Automata. In: Shannon, C.E., McCarthy, J. (eds.) Automata Studies, pp. 3–42. Princeton University Press (1956)
17. Laird, J.E., Newell, A., Rosenbloom, P.S.: SOAR: an architecture for general intelligence. Artifitial intelligence 33, 1–64 (1987)
18. Letichevsky, A., Gilbert, D.: A general theory of action languages. Cybernetics and System Analyses 1 (1998)
19. Letichevsky, A.A., Gilbert, D.: A Model for Interaction of Agents and Environments. In: Bert, D., Choppy, C., Mosses, P.D. (eds.) WADT 1999. LNCS, vol. 1827, pp. 311–328. Springer, Heidelberg (2000)
20. Letichevsky, A.A., Letychevskyi, O.A., Peschanenko, V.S.: Insertion Modeling System. In: Clarke, E., Virbitskaite, I., Voronkov, A. (eds.) PSI 2011. LNCS, vol. 7162, pp. 262–273. Springer, Heidelberg (2012)
21. Letichevsky, A.: Algebra of behavior transformations and its applications. In: Kudryavtsev, V.B., Rosenberg, I.G. (eds.) Structural Theory of Automata, Semigroups, and Universal Algebra, NATO Science Series II. Mathematics, Physics and Chemistry, vol. 207, pp. 241–272. Springer, Heidelberg (2005)
22. Letichevsky, A., Kapitonova, J., Letichevsky Jr., A., Volkov, V., Baranov, S., Kotlyarov, V., Weigert, T.: Basic Protocols, Message Sequence Charts, and the Verification of Requirements Specifications. In: WITUL (Workshop on Integrated Reliability with Telecommunications and UML Languages), ISSRE 2004, Rennes (November 4, 2005)
23. Letichevsky, A., Kapitonova, J., Letichevsky Jr., A., Volkov, V., Baranov, S., Kotlyarov, V., Weigert, T.: Basic Protocols, Message Sequence Charts, and the Verification of Requirements Specifications. Computer Networks 47, 662–675 (2005)
24. Letichevsky, A., Kapitonova, J., Volkov, V., Letichevsky Jr., A., Baranov, S., Kotlyarov, V., Weigert, T.: System Specification with Basic Protocols. Cybernetics and System Analyses 4 (2005)
25. Letichevsky, A.A., Kapitonova, J.V., Kotlyarov, V.P., Letichevsky Jr., A.A., Nikitchenko, N.S., Volkov, V.A., Weigert, T.: Insertion modeling in distributed system design. Problems in Programming 4, 13–38 (2008)

26. Letichevsky, A., Letichevskiy, O., Morokhovets, M., Peschanenko, V.: System of Programs Proving in Problems of computer intellectualization. In: Velichko, V., Volosin, A., Markov, K. (eds.), pp. 133–140. Kyiv: V.M.Glushkov Institute of Cybernetics (2012)
27. McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bull. of Math Biophy. 5, 115–133 (1943)
28. Milner, R.: A Calculus of Communication Systems. LNCS, vol. 92. Springer, Heidelberg (1980)
29. Milner, R.: Communication and Concurrency. Prentice Hall (1989)
30. Milner, R.: The polyadic π-calculus: a tutorial, Tech. Rep. ECS–LFCS–91–180, Laboratory for Foundations of Computer Science, Department of Computer Science, University of Edinburgh, UK (1991)
31. Minsky, M.: The Society of Mind. Touchstone Book, p. 339 (1988)
32. Park, D.: Concurrency and automata on infinite sequences. LNCS, vol. 104. Springer (1981)
33. Petri, C.A.: Kommunikation mit Automaten, Bonn: Institut fur Instrumentelle Mathematik. Schriften des IIM Nr. 2 (1962)
34. Samsonovich, A.V.: Toward a unified catalog of implemented cognitive architectures (review). In: Samsonovich, A.V., Johansdottir, K.R., Chella, A., Goertzel, B. (eds.) Biologically Inspired Cognitive Architectures 2010: Proceedings of the First Annual Meeting of BICA Siciety. Frontiers in Artifitial Intelligence and Applications, vol. 221, pp. 195–244 (2010)

From Collegial to Collaborative – A Model for Building Trust and Partnership in Information Technology

Gary L. Pratt

Eastern Washington University, Cheney, Washington, USA
gpratt@ewu.edu

Abstract. Colleges and universities compete for students, faculty, and business, industry, and research partnerships with quality programs, strong faculty, research opportunities, affordable cost, and high student success factors. Yet, at the infrastructure level, most of these institutions provide many similar information technology services and support. On top of this, many of these institutions struggle to provide this quality infrastructure because of a variety of factors, including: shrinking budgets, minimal strategic planning, and a lack of institutional vision of information technology as a strategic asset. This presentation will showcase best practice examples of how higher education institutions can band together, to create strong consortium relationships that can help all partners in this relationship move forward as a strong force. Examples will include actual successes experienced by the Kentucky Council on Postsecondary Education's Distance Learning Advisory Committee (DLAC), the Washington Legislative Technology Transformation Taskforce (TTT), and the Washington Higher Education Technology Consortium (WHETC). These successes range from statewide strategic planning efforts, to significant consortial purchasing contracts, to collaborative technology systems, services, and training opportunities. This presentation will show that institutions can be stronger working together than working individually.

Keywords: University consortium, best practice, competition, infrastructure, information technology, strategic asset, strategic planning, collaborative technology system.

1 Introduction

Through efforts at a number of colleges and universities throughout the United States of America, the author has typically experienced entering an environment of collegiality, both at the institutional level as well as at the statewide level. Yet, as friendly as peers are in the higher education information technology realm, much of the time, there is little trust or willingness to work together towards a collaborative success.

Because the benefits of collaboration are so significant, the author has taken the lead role in a number of initiatives, leading groups of information technology leaders to the realization of the benefits that come from working collaboratively together.

This article will share a number of stories about moving higher education institutions and the information technology organizations that support them towards a world of collaboration.

2 Challenges to Higher Education Success

"Higher education is facing its own perfect storm . . . These disruptive forces are compelling higher education to adapt" [1, p. 7]. There are many disruptive forces in today's environment that are compelling higher education to adapt and change, including:

- Growing enrollments
- Increasing student expectations
- Rising student debt
- Changing student demographics
- Declining financial resources
- Increased competition
- Breakneck speed of technological change

These forces are causing institutions to ask, how can colleges and universities evolve to, not just mitigate these factors, but move their institutions forward successfully?

Another issue increases the complexity of these disruptions; our institutions are competitors. They compete for students and enrollments. Regionally, institutions establish turf boundaries. Institutions 'sell' their programs of study, creating a 'brand' focused on the institution's niche markets. This all begs the question, 'why work together?'

Yet, below the line of competition, the information technology organizations within these higher education institutions are doing many of the same things. There is a commonality in what these organizations do:

- Run similar infrastructures (data and telecommunications networks, telecommunications systems, data centers)
- Purchase, implement, maintain, and troubleshoot similar applications, technology, and research systems
- Provide similar services and support for institutional constituents

3 Challenges to Information Technology Success

Besides the external factors affecting colleges and universities listed above, the information technology divisions that support them are facing their own challenges. In many institutions, there is not a current, relevant Institutional Strategic Plan. Without a true understanding of the institution's mission, vision, goals, and priorities, information technology organizations must follow their 'best guess' as to where the institution is headed. Without an institutional strategic plan, it is a challenge to create an information technology strategic plan.

Another concern at many colleges and universities is that information technology is not considered as a strategic asset. There is no vision as to how information technology can help the institution to achieve its mission. Information technology is seen only as a utility. As stated in Does IT Matter [2], in this environment, information technology just becomes another cost of doing business.

Invariably, information technology is seen as a utility, like lights, heating/cooling, and plumbing. When someone walks into a room and flips a switch, he/she expects the lights to come on; turn the knob and water pours out; turn on the computer and the applications and Internet are there, ready to work.

This is not a complete fallacy. A large portion of information technology is a utility. Two-thirds to three-fourths of the information technology organization is a utility, including all of the primary applications and systems, most of the infrastructure, and all of the service and support provided by the information technology team. All of this happens as a standard course of business and is just an accepted part of the operation. This is where information technology becomes invisible to the user. It is just expected to be there and available. Users don't think about it. This is a very important part of the institution's operations, but it is not strategic [3].

The difficult message to get across is that there is another important component of information technology, where information technology is the strategic asset. It comes with the ever-changing nature of technology. New systems allow for new efficiencies, effectiveness, and streamlining processes. Innovation comes with new technologies driving change in all aspects of the operation of campus. These innovations are where institutions start to differentiate themselves [3].

Research is another major component in most universities. Research focuses on the creation of new knowledge. Information technology plays a significant role in research with high capacity computing, enormous bandwidth, and the development of experimental systems and applications. This is the visible part of information technology. This is where information technology becomes strategic [3].

Lastly, because of the declining financial resources institutions are facing, information technology organizations are also having to do more with less.

4 Strategic Planning

So, how does an information technology organization overcome these challenges? First, become a part of the institutional strategic planning process.

At Eastern Washington University, early in the author's time there, Information Technology was the only division that developed and successfully implemented a divisional strategic plan. As a result of leading this process, the author was charged with developing the institution's strategic plan, and now has institutional strategic planning as a part of his job description.

To do this successfully, one must ensure that the Information Technology organization is aligned with all aspects of campus. A college or university campus is not a single business. The higher education institution represents a number of different businesses, each with a unique need for information technology support. From the more traditional business models of the business and finance and student affairs divisions to the more uniquely focused businesses of the advancement, athletics, marketing, and (of course) information technology divisions, there are drastically different technology needs. Within the academic side of the institution, there are many different, and unique, academic disciplines that each have differing information technology requirements. Throw academic research into the mix, and the information technology organization has quite the challenge aligning across the institution.

An institutional strategic plan is of key importance to all divisions and departments within the institution. Key strategic thinking is important in changing or affirming the institution's mission, vision, values, and key performance indicators. Getting assessment and input from all directions will help leaders assess significant issues facing the institution; perform an analysis of the institution's strengths weaknesses, opportunities, and threats; develop strategies and actions; and articulate the vision of the institution in the future. All of this opens the doors to operational and tactical planning [4].

Aligning the Information Technology Organization with the institutional planning process ensures that it is a key consideration throughout the institution's planning continuum. Alignment happens horizontally:

- Year-to-year Institutional Tactical/Action Plans
- Budget Planning
- Capital Planning
- Accreditation and Program review
- Continuous Process Improvement
- Performance Management

Alignment also happens vertically, throughout the organization hierarchy:

- College and Division Level Planning
- Department Level Planning

5 Collaboration

"col· lab· o· ra· tion [kə làbbə ráysh'n] the act of working together with one or more people in order to achieve something" [5]. One of the main factors that can overcome the challenges to information technology success is collaboration. Whether it is within the institution, regionally, or on a grand scale, by working together with other divisions, other universities, or with outside partners, the Information Technology organization can make itself strategic, overcome dwindling fiscal resources, and enhance its staffing and skill resources.

Unfortunately, because of institutional competition, pride, or just plain indifference, collaboration tends to be the last thing on the minds of institutional and information technology leaders. They would rather be collegial than collaborative.

But, collaboration is a real way to show that the whole is much greater than the sum of the parts. Collaboration takes effort. It often requires leaders to take a risk, putting their own needs on the shelf for a bit to focus on the greater good. It requires commitment on the part of all partners.

6 Examples of Successful Collaborations

The author has participated in and led many successful collaborations during his career. These are examples that will show how collaboration has worked.

6.1 Learning Commons

An exciting example that showcases the strength of collaboration within the institution was the creation of the Learning Commons at Eastern Washington University. The university's strategic plan showcases its highest priority as student success. The Learning Commons initiative, for the first time, brought together disparate groups on the campus that focused on various aspects of student success into a single facility:

- Student tutoring – 1-on-1 tutor working with student on any number of discipline support
- Assistance with composition – assistance for students on all aspects of composition
  - Developing concepts
  - Writing
  - Publication
  - Graphics
- Multi-Media lab – Providing students with access to high-end multimedia tools and technology
- Library - the center was built in the intellectual heart of Campus

The collaboration among and between the partners has also significantly increased the quality of the programs that are provided for students. As an example, the Writer's Center faculty are working directly with the multi-media lab technical experts to help students create high-end graphics, animation, and audio-infused projects, papers, and presentations.

Within a single academic year, this facility has quadrupled the number of students who have accessed these services when they were operated separately. Because of the success of the Learning Commons, more partners are being included, building a larger presence.

6.2 Kentucky State-Wide Distance Learning

Prior to Eastern Washington University, the author was the Chief Information Officer at Northern Kentucky University. During this time, the Commonwealth of Kentucky had a desire to develop a strong, statewide mandate to deliver technology-enhanced education. In 1997, Kentucky, through legislation, created the Kentucky Virtual

University, a collaboration among public colleges and universities in Kentucky focused on developing and offering online courses and programs. As a part of the same legislation, they created the Distance Learning Advisory Committee.

Sally Johnstone, Director at the time of the Western Cooperative for Educational Telecommunications (WCET) spoke to the Statewide Distance Learning Advisory Committee. Her focus was on how statewide collaborations of colleges and universities could successfully build a framework for technology and eLearning. Johnstone stated, "policy development begins with a question, created by a problem. Progress means moving from: 'should we?' to 'can we?' to 'we will'?" [6].

Although they successfully developed the Kentucky Virtual University, they had a number of issues, not the least of which was a problem providing a single learning management system (LMS) infrastructure. The Kentucky Virtual University had a big failure trying to offer an LMS to the state. As a result, each institution went its own way, implementing individual systems.

Because the author had begun to develop a reputation for collaboration, he was asked to lead a group conversation with the state's public colleges and universities about the possibility of moving back to a standard LMS to support the Kentucky Virtual Campus and the individual institutions.

The idea was that, the success of the Kentucky Virtual University was hampered by the fact that there was little rhyme or reason in how online classes and programs were offered. Unfortunately, because of the past failure, there was little trust among the group for a centrally facilitated system. Everyone involved was up in arms, thinking that their approach was the best. They were strongly holding on to their turf.

But, it was interesting that, regardless of how strong their attitudes were, they didn't want to stop the conversation. They kept talking about potential opportunities for how the group could work together. In fact, the conversation continued for hours beyond the original scope of the meeting. Bringing the meeting to a close, everyone agreed to continue the conversation at a follow-up meeting.

Before the meeting, the author was approached by a person at the state-level, who suggested throwing out the possibility of providing a financial incentive for moving to a standard, in other words - how would these people respond to getting the system for free?

The second meeting began with a review of the last conversation. The group focused on potential collaborations that would help statewide eLearning. There were many great potential projects, including shared training, shared content repositories, multi-institution courses, etc. This was when the concept of a funded LMS was brought forward. The response was very positive. During the next 9 months, the collaboration was extended to include the private higher education community, and the public primary and secondary schools. The result was a single standard system that was fully funded by the Council on Postsecondary Education for the public higher education institutions, fully funded by the Kentucky Department of Education for the public primary and secondary schools, and was available for all private higher education institutions. The institutions had the choice of implementing their own instance, or join a hosted solution, hosted by a state-run entity.

This important collaboration provided for a greatly discounted price, 65% off list price, saving the Commonwealth of Kentucky a significant amount of money; provided for a state-run hosted solution that allowed small schools to join in; and set up a collaborative structure that focused on other opportunities, including shared training, shared content repository, and multi-institutional academic programs.

6.3 Washington Technology Transformation Taskforce

In 2009, the Washington State House of Representatives approved legislation that created a group called the Technology Transformation Taskforce. The author was asked to co-chair this statewide committee, along with the President of Pierce College. This taskforce consisted of 20 community college and university faculty, information technology staff, and academic and student affairs administrators. In addition, there were more than 60 additional participants that participated in a number of sub-committees.

The purpose of the Technology Transformation Taskforce was to study a number of factors within the state of Washington's public higher education colleges and universities, including:

- Enterprise Applications
- Enrollment Services
- Library Systems
- IT Infrastructure Systems

The need of this Taskforce arose from the economic downturn that happened to the USA's and state of Washington's economy.

The Taskforce developed a survey instrument that was sent to each public institution to inventory all of the mainline technologies at each campus. From the compiled data, the Taskforce identified a number of potential initiatives that would benefit the state. These initiatives were prioritized based on a ranking of risk, cost, complexity, time to implement versus benefit, and impact. Initiatives that fell lower on the cost, complexity, and time categories showcased potential 'low hanging fruit.'

Recommended Game Changer initiatives included:

- Create a Collaborative Governance Framework
- Statewide Online Academic Planning System
- State Management of Online Education
- Statewide Repository for Business Intelligence
- Federated Identity Management System
- Consolidated Disaster Recovery Plan

Quick Win Recommendations

- Statewide Professional Development Portal
- Implement Student Relationship Management System
- Expand participation in Library Alliance
- Leverage Buying Power

"Truly transformative initiatives will require enhanced IT governance structures and focused collaboration among institutions" [1]. Many of these initiatives came about as a result of the creation of the collaborative governance group that was established soon after this effort. This group is discussed next [1].

6.4 Washington Higher Education Technology Consortium

One of the major recommendations from the Technology Transformation Taskforce included the development of a collaborative governance framework. As a result, the author led the development of and served as the inaugural chair of the Washington Higher Education Technology Consortium (WHETC).

The mission of WHETC is to improve the efficiency, effectiveness, quality, and innovation of learning, research, and service though the strategic and collaborative use of technology in higher education. Its goals include:

- Developing practices and relationships that effectively utilize statewide, institutional and agency resources
- Leading the collaboration and coordination of technology use across the State's public higher education institutions
- Identifying, developing, and delivering high-quality technology solutions for Washington's learners, staff and faculty.

Guiding Principles include:

- Find common ground; respect unique missions
- Be guided by quality
- Focus on customer/stakeholder service
- Build and maintain a solid technology infrastructure
- Provide responsible stewardship of public resources
- Support seamless/accessible technology-enhanced education
- Assure security and privacy
- Manage all discussions/initiatives in a collaborative manner
- Ensure communication among all parties is seamless

Within the first 3 years of the WHETC, successes include:

- The State of Washington created a new State Chief Information Officer whose responsibility it was to coordinate and consolidate State IT resources. The State's Higher Ed institutions were exempt of many of these efforts, except for large project oversight (>$1 million). WHETC, working with the state CIO, developed a draft peer oversight structure for higher education institutions.
- The first collaborative purchasing initiative was for a lecture capture technology. By coordinating as a whole state, WHETC was able to secure a 75% discount on the product.
- A recent and large collaborative purchasing initiative had all of the state's public colleges and universities coming together to purchase a single LMS. This consortial effort provided many of the same benefits as the Kentucky LMS project mentioned above.

- The last collaborative initiative that WHETC took on is the negotiation with Microsoft for the purchase and implementation of MS Office365, a cloud-based email, calendaring, MS Office, SharePoint, VoIP, and much more. This negotiation provided the mid-level version of the product for free. The cloud-based technology makes for a decreased footprint in individual data centers, reduced staff administration of the systems, and greatly reduced costs in systems management.

7 Conclusion

This quote from the United States Department of Education 2010 National Educational Technology Plan says it all: "Education is the key to America's economic growth and prosperity and to our ability to compete in the global economy. It is the path to good jobs and higher earning power for Americans. It is necessary for our democracy to work. It fosters the cross-border, cross-cultural collaboration required to solve the most challenging problems of our times" [7].

References

1. Pratt, G.L., Yochum, D.: Higher Education Technology Transformation Taskforce Report (2010)
2. Carr, N.G.: Does IT Matter? Information Technology and the Corrosion of Competitive Advantage. Harvard Business School Publishing Corporation, Boston (2004)
3. McNaughton, J.R. (Speaker): The Future Life of the SLG CIO?!?! In: IPMA Conference, Chelan (2012)
4. Norris, D.M., Poulton, N.L.: A Guide to Planning for Change. Society for College and University Planning (2008)
5. BING, `http://www.bing.com/search?q=collaboration&PQ=collaboration&SP=1&QS=AS&SK=&sc=8-13&form=LEMBSS&pc=MALC` (2013)
6. Johnstone, S.: Presentation to the Kentucky Distance Learning Advisory Committee. In: Western Cooperative for Educational Telecommunications (WCET) (2004)
7. O. E. T - U. S. D. O. E.: Transforming American Education: Learning Powered by Technology, National Technology Plan 2010, Office of Educational Technology, US Department of Education, p. ix (2010), `http://www.ed.gov/sites/default/files/netp2010.pdf`

An Integrated Development Framework for Advanced IT-Service Management: Proof-of-Concept Project in Universities Domain

Nikolay Tkachuk, Vladyslav Sokol, and Kateryna Glukhovtsova

National Technical University "Kharkov Polytechnic Institute",
Frunze str., 21, Kharkov, Ukraine
tka@kpi.kharkov.ua, vladislav.sokol@gmail.com, kat_1109@mail.ru

Abstract. An integrated knowledge-based framework for improving IT-services management in organization is elaborated, which includes 3 interconnected tasks to be resolved: 1) to choose an effective modules configuration of ITSM-system (ITSMS) according to specific features and needs in application domain; 2) to integrate a given ITSMS with an existing enterprise architecture; 3) to provide an advanced incidents management in ITSMS. The serviceability of this approach was tested successfully as a proof-of-concept project at the National Technical University "Kharkov Polytechnic Institute" (Ukraine).

Keywords: IT-service management, effectiveness, multi-criteria ranking, data integration, adaptive ontology, case-based reasoning, e-learning.

1 Introduction: Problem Actuality and Research Objectives

In recent years the concept of ITIL (IT Infrastructure Library) [1], and respectively a new kind of computer-aided management systems: IT Service Management Systems (ITSMS) became a growing and perspective platform to resolve a very important and complex technical problem and, at the same time, a business-focused one: how to organize a well-structured and controllable IT-environment at an appropriate organization?

According to ISO/IEC 20000 [2] an IT Service Management System (ITSMS) provides "*…a framework to enable the effective management and implementation of all IT-services*". Due to highly complex and multi-dimensional structure of IT-services in large modern business organizations, which ITSMS are dealing with, a lot of publications in this R&D domain present sophisticated approaches to design and to use these facilities. One such important topic in ITIL-ITSM domain is the integration of ITSMS functionality into an enterprise architecture (see, e.g. in [3,4]). Another recent trend in ITSMS-development is the usage of ontology-based models and model-driven architectures (MDA) [5, 6] to handle domain knowledge and to re-use software components. Especially, their authors emphasize the actual need to elaborate and to apply several knowledge-oriented approaches to requirements engineering within

ITSMS-development framework, and to provide quantitative quality assessment of appropriate software solutions.

Taking into account some ITSMS-issues mentioned above, the main objective of the research presented in this paper is to propose the first vision for intelligent complex approach to increase efficiency of typical ITSMS, with a proof of concept based on the ITSMS university case-study. The rest of this paper is organized in the following way: Section 2 analyses some existing ITSMS, introduces our vision about their typical functionality, and shows the list of prioritized tasks to be resolved to increase an efficiency of an ITSMS. In Section 3 we present the method elaborated for effective ITSM-modules configuring in a target business organization, and Section 4 reports the first version of ITSMS - ontologies to integrate the selected modules into enterprise architecture (EA). In Section 5 the design perspective for the combination case-based reasoning (CBR) with ontology-based approach to advanced incident management in ITSMS is briefly outlined. In Section 6 we present the university case-study for our method to estimate the effectiveness of different ITSMS configurations and discuss the results achieved. In Section 7 the paper concludes with a short summary and with an outlook on the next steps to be done in the proposed development framework.

2 Critical Analysis of ITSMS Typical Functionality and an Integrated Framework to Increase Its Efficiency

In order to elaborate a way how to provide a complex approach to increase the efficiency of ITSM-system operating, it is necessary to analyze critically its typical functionality and to understand its specific features.

2.1 Overview of Existing ITSMS

We have analyzed some already existing ITSMS [7-10], and basically, all such systems can be divided into 3 groups, namely:

(a) advanced business ITSM-products;
(b) open source ITSM-solutions; and
(c) company-oriented individual ITSM-systems.

To the group (a) belong such systems as, e.g., HP OpenView Service Desk [7] and BMC Remedy [8]. The first software product is the absolutely leader in this market segment, because the most part of organizations which prefer ITSM-business solutions from the group (a), are using exactly HP-platform. The number of installations running the second product is less than for HP, at least because of more expensive costs of Remedy ITSM Suite. For our purpose: to understand and to analyze critically a typical ITSMS-functionality, exactly such this product-group is important, and their main options are:

- An enhanced data modeling toolkit that enables to identify all necessary informational attributes for domain modeling elements and relationships between them;

- A workflow-editor which allows to describe business processes running in a target organization, their possible changes and requests for IT- services needed for customers;
- An advanced data visualization;
- A special facility to analyze the mutual influence of changes that occur in different IT-services configurations, which helps to prevent conflicts in their development.

Besides that, the typical features of these systems are the following:

- Access via Web-interfaces;
- High degree of system scalability due to the possibility for installation with distributed business logic processing on several application servers and with data synchronization using appropriate backup/restore and data replication procedures;
- Availability of full system documentation suite, including a description of all key processes and their relationships, procedures, roles using RACI (Responsibility-Accountable-Consult before-Informed) matrix.

The summarized results of this study are presented in the Table 1 (where the marks from 1 to 5 are used for excellence estimation).

Table 1. Results of comparison for some ITSMS

Criteria / Systems	**BMC Remedy ITSM Suite 7.5**	**Axios Assyst 7.5**	**HP Service Manager 7.10**	**OMNINET OmniTracker ITSM Center 2.0**
Basic functionality	5	5	5	4
Maintainability	5	4	5	4
Documentation suite	4	5	5	4
Scaleability	4	2	3	5
Web-interface	5	5	5	5

ITSM-solutions from the group (b) also are used in practice, but they definitely have limited functionality and provide a lower level of IT-services management. The typical open source ITSMS are, for instance, GLPI [8], OTRS [9], and some others, which are listed at the Web-resource SourceForge [10].

And, objectively, the business organizations, which are not ready to buy advanced software products from group (a), and which are not satisfied with functionality provided by ITSM-systems from group (b), because they have some specific IT-needs and challenges, exactly these companies try to develop their own ITSM-solutions to be considered as members of the group (c). The more detailed comprehensive study of some existing ITSM-systems is presented in [11].

2.2 The Typical ITSMS-Functionality

Based on the given analysis of the real ITSMS (see above), we have elaborated the following vision for their typical functionality (see Fig. 1).

There are 5 main subsystems (or packages) of system functions, namely:

1. *IT Business Alignment*: this subsystem is supposed to implement a IT-strategy in given business organization with respect to its main goals and needs, and to provide a base for costs assessment to whole IT-infrastructure;
2. *Service Operations*: this facility is responsible for customer's requests management (regarding to a current incident and to a related problem), and for providing of ITSM-support functions;
3. *Service Delivery Assurance*: this functional package implements a configuration and change management of all ITSM-software tools thus is extremely important for a stable IT-environment;
4. *Service Design and Management*: this ITSMS-functionality provides detailed information about new perspective IT-services to be designed with respect to their availability and quality for IT-customers;
5. *Service Development and Deployment*: this subsystem allows to create and to test new ITSM-services and appropriate IT-infrastructure solutions, including installation of new hardware components, development of additional software applications, and training programs for ITSM-staff' and for end-users as well.

As we can see on the structure presented in Fig.1, each of these 5 subsystems is built from several functional modules (they are depicted as UML-classes). The most important of them are the following ones (they are labeled in Fig. 1 accordingly):

- *Module Ì1* ="*Incident and service request management*": it includes organizational procedures and appropriate tools to resolve current incidents, which IT-service users are facing with (hard-and software errors, network connection problems, request for consultations, etc.);
- *Module Ì2* = "*Problem management*": this facility provides tools to detect and to eliminate any problem situation which is a reason for different incidents;
- *Module Ì3* = "*Configuration management*": this module supports all operating subschemes in the IT-infrastructure of given business organization;
- *Module Ì4* = "*Change management*": it supervises and coordinates all changes which arise in IT-infrastructure; .
- *Module Ì5* ="*Service level management*": this unit is responsible for definition and implementation of an appropriate level of IT-services to be provided for customers.

In ITIL-best practice manuals (see e.g. in [12]) the following 3 main schemes are considered to introduce these modules into IT-infrastructure of a target organization: a classic scheme (S1); a contract scheme (S2); an infrastructure-centered scheme (S3).
A *classic scheme* S1 is the most applied solution in the ITSM-domain, and it supposes the following sequence of modules *Ì1-Ì5*:

$$S1 = (M1, M3, M4, M2, M5) \tag{1}$$

This approach quickly allows to resolve the most actual communication problems between IT-service department and customers basing on incident management (module *Ì1*), and it provides some tools for all IT-services support (the modules *Ì3* and *Ì4*), and after that a platform for future IT-infrastructure development is introduced (modules *Ì2* and *Ì5* respectively). However this scheme is a most expensive way for a

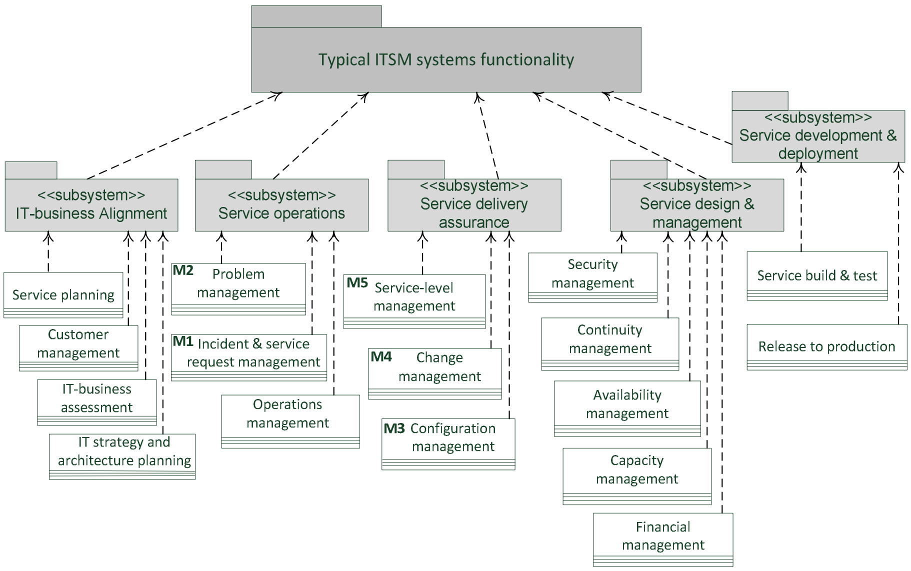

Typical ITSM systems functionality
<<subsystem>> IT-business Alignment
<<subsystem>> Service operations
<<subsystem>> Service delivery assurance
<<subsystem>> Service design & management
<<subsystem>> Service development & deployment
Service planning
Customer management
IT-business assessment
IT strategy and architecture planning
M2 Problem management
M1 Incident & service request management
Operations management
M5 Service-level management
M4 Change management
M3 Configuration management
Security management
Continuity management
Availability management
Capacity management
Financial management
Service build & test
Release to production

Fig. 1. Typical ITSMS functionality

given business organization, and it requires a lot of resources exactly at an initial phase of whole ITSM-configuring framework.

A *contract scheme* S2 actually aims to formalize a communication process between IT-service department and customers, and it has the following modules-workflow:

$$S2 = (M5, M3, M1, M4, M2) \tag{2}$$

In this case all customer requirements to IT-services have to be collected and specified (in module *Ì5*), and appropriate IT-infrastructure sub-schemes can be built (using module *Ì3*), in order to define prospective IT-strategy in the target organization, next an operative ITSM-functionality is provided, including incident management (in module *Ì1*), change management (in module *Ì4*), and problem management (in module *Ì2*). Obviously, this scheme definitely has some risk factors regarding its efficiency, if the initial IT-service specifications were done not correctly (in module *Ì5*).

And, finally, *an infrastructure-centered scheme* S3 proposes the modules sequence indicated as following:

$$S3 = (M3, M4, M2, M1, M5) \tag{3}$$

that is, firstly, to provide tools for all IT-services support (modules *Ì3* and *Ì4* respectively). Secondly, this approach allows to manage all typical problem situations (in module *Ì2*), and already based on this one to detect and to resolve corresponding incidents by IT-service customers (in module *Ì1*). Thirdly, it creates an opportunity to define in computer-aided way the necessary composition and the IT-service level management (in module *Ì5*).

It is necessary to note that besides some empirical recommendations concerning the possible ITSM-modules configurations defined as (1)-(3), in the appropriate technical documentation there are no more or less proved suggestions about possible quantitative estimations for effectiveness of these alternative approaches.

2.3 The Complex of Intelligent Tasks to Increase of ITSM-System Efficiency

Taking into account the results of the analysis (see above), and based on some modern trends in the domain of ITSMS-development (see Section 1), the following list of prioritized problems can be composed in order to increase ITSMS-efficiency, namely

(I) to provide *an effective configuring* of ITSM-modules for a target organization, taking into account its specific features and needs;
(II) to elaborate *an integration framework* for a given ITSM-system's configuration and for an existing enterprise architecture (EA);
(III) to support *an advanced incidents management* in the already installed ITSM-system.

In our opinion, the problem (I) can be resolved basing on some expert methods for multi-criteria ranking, with respect to specific IT-infrastructure's features and customer needs in a concerned business organization [13,14].

The problem (II) belongs to already well-known integration issues in distributed heterogeneous information systems, and e.g. an ontology-based approach can be used for this purpose (e.g. in [3,6,15]).

And, finally, to solve the problem (III) an additional decision-making functionality for typical ITSM-services (see Fig.1) has to be elaborated, e.g. based on the combination of case-based reasoning (CBR) approach with ontologies [16-18]. Below these problems and their possible solutions are presented and discussed in more detail.

3 The Method for Effectiveness Estimation of Alternative ITSM-Module Configurations

To formalize the problem (I) from their list considered in the Section 2.3, namely: to *provide an effective configuring* of ITSM-modules for a target business organization, the following factors have to taken into account:

- Such a problem has a high complexity grade and it is semi-formalized;
- Estimation criteria for it are of different nature and they are multi-valued;
- An information base to solve this task mainly can be collected basing on expert data only;
- An available expert data could be quantitative and qualitative values both.

To solve this problem we have chosen one of the multi-criteria ranking methods, which is presented in [14]. Accordingly to this approach the following steps have to be performed:

Step 1. A set of possible alternatives,

$$X = \{x_1, x_2, ..., x_n\} = \{x_i, i = \overline{1,n}\},\tag{4}$$

and a set of global importance criteria to characterize these alternatives

$$K = \{K_1, K_2, ..., K_m\} = \{K_j, j = \overline{1,m}\}\tag{5}$$

have to be defined. In our case as such alternatives the possible configurations of ITSMS-modules should be considered (see in 2.2 above).

Step 2. Each global criteria K_j is characterized by a subset of appropriate local criteria (see the example of their possible definitions and values given in Section 6, in Table 5 and Table 6 respectively)

$$K_j = \{k_{j1}, k_{j2}, ..., k_{jQ}\} = \{k_{jq}, q = \overline{1,Q}\},\tag{6}$$

further, a set of membership functions according to all local criteria alternatives

$$\{\varphi_{k_{j1}}(x_i), \varphi_{k_{j2}}(x_i), ..., \varphi_{k_{jQ}}(x_i)\} = \{\varphi_{k_{jq}}(x_i), q = \overline{1,Q}, j = \overline{1,m}\},\tag{7}$$

and the weight coefficients of their relative importance for these local criteria

$$\{w_{j1}, w_{j2}, ..., w_{jQ}\} = \{w_{jq}, q = \overline{1,Q}\}\tag{8}$$

have to be determined (e.g., using pair-wise comparison in AHP method [13]), where the following condition (9) has to be fulfilled.

$$\sum_{q=1}^{Q} w_{jq} = 1\tag{9}$$

Step 3. To determine membership functions of alternatives $\{x_i, i = \overline{1,n}\}$ to criteria $K_j, \{j = \overline{1,m}\}$ based on an additive convolution of their local criteria

$$\varphi_{k_j}(x_i) = \sum_{q=1}^{Q} w_{jq}\varphi_{k_{jq}}(x_i)\tag{10}$$

Table 2. Definition of membership functions for criteria to alternatives (fragment)

Alternatives		**Criteria K**						
		K_1			…	K_M		
		k_{11}	…	k_{1Q}	…	k_{M1}	…	k_{Mm}
X	x_1	$\varphi k_{11}(x_1)$	…	$\varphi k_{1Q}(x_1)$	…	$\varphi k_{M1}(x_1)$	…	$\varphi_{Mm}(x_1)$
	…	…	…	…	…	…	…	…
	x_n	$\varphi k_{11}(x_n)$	…	$\varphi k_{1Q}(x_n)$	…	$\varphi k_{M1}(x_n)$	…	$\varphi_{Mm}(x_n)$

Step 4. Taking into account the membership functions obtained $\{\varphi_{K_j}(x_i), j = \overline{1,m}\}$ for all alternatives $x_i, \{i = \overline{1,n}\}$ it is possible to determine a joined membership function for a generalized criterion K:

$$\varphi_K(x_i) = \sum_{j=1}^{m} w_j \varphi_{K_j}(x_i) \tag{11}$$

where $w_j, j = \overline{1,m}$ are coefficients of their relative importance $K_j, j = \overline{1,m}$.

Step 5. Finally, an alternative with a maximum value of membership function for generalized criterion K can be chosen as a target solution:

$$\varphi(x^*) = \max\{\varphi_K(x_i), i = \overline{1,n}\}) \tag{12}$$

Below in Section 6 we present the case-study, which was performed to prove this method, and we discuss the results achieved.

4 Ontological Specifications for ITSMS-EA Integration Framework

As already mentioned above (see Section 2), any ITSMS has to be integrated into an existing EA of a target organization. In our approach this problem (II) has to be resolved for an ITSMS-configuration defined with the method presented in Section 3.

This issue is already discussed intensively in a lot of publications, and their authors consider both its conceptual and technological aspects. E.g., an ITSMS-EA integration based on well-known SOA – framework is presented in [3], and as the important conceptual input for this issue the appropriate meta-model (actually, some kind of a domain ontology) for IT services is designed. In [5] an approach to integration of ITSM-services and business processes in given organization is elaborated, using ontological specifications to formalize the good practice guidance for ITSM. An ontology-based framework to integration of software development and ITSMS-functioning is proposed in [15], thus resulting in enhanced semantic-aware support tools for both processes. Even this brief overview allows us to conclude that exactly an ontology-based approach is a most effective way to solve this problem. That is why, in our opinion, to provide ITSM-EA integration effectively, it is necessary to combine the following information resources (IR), namely:

a) IR related to ITSMS – functionality,
b) IR concerned EA-domain,
c) IR characterized a target organization (TO), which is facing an ITSMS-EA integration problem with.

Let's define these IR (a)-(c) as: *Onto-ITSMS*, *Onto-EA*, and *Onto-TO* respectively. Thus, the IR needed to provide an ITSMS-EA integration should be specified using an appropriate joined ontology, designated as *Onto_ITSMS-EA*.

$$Onto\_ITSM-EA=<Onto-ITSMS,Onto-EA,Onto-TO> \tag{13}$$

Obviously, some already existing ITIL / ITSM ontological specifications can be used for this purpose, e.g.: *Onto-ITIL* ontology elaborated in [5] basing on *OpenCyc* ontology (www.opencyc.org), *Onto-SPEM* (Software Process Engineering Meta-model) ontology [19], and *Onto-WF* (WorkFlow) ontology [20]. Taking these resources into account, we can represent the ontological specification for *Onto_ITSM* in the following way.

$$Onto-ITSMS=<Onto-ITIL,Onto-SPEM,Onto-WF> \tag{14}$$

There are also several ontologies developed to specify EA, and according to one of recent and comprechensive researches in this domain presented in [21], we accept the following 3-level definition for EA-ontology.

$$Onto-EA=<Onto-BT,Onto-AC,Onto-RS> \tag{15}$$

where: *Onto_BT* is a sub-ontology of *Business Terms (BT)*, *Onto-AC* is a sub-ontology of *Architecture Components (AC),* and *Onto-RS* as a sub-ontology of *RelationShips (RS)* among items of AC.

And finally, to define an *Onto-TO* ontology for target organization given in expression (13), its specific features and needs related to ITSMS-usage within existing EA have to be taken into account. As a small excerpt of such domain-specific *Onto-TO*, which is elaborated in our University-ITSMS case-study (see Section 6), the following UML-class diagram in Fig. 2 is shown.

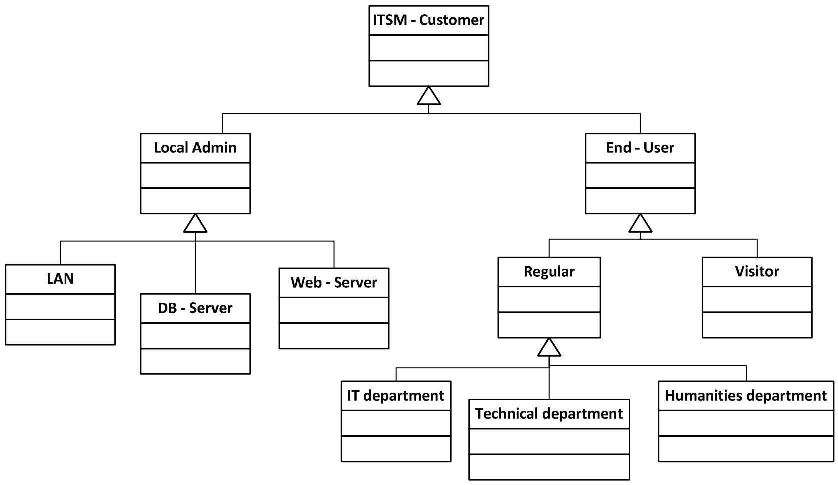


Fig. 2. Taxonomy of customers in a University-ITSMS as a part of a *Onto-TO* ontology

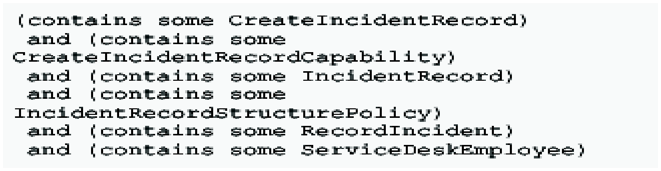
The proposed ontology-based approach for ITSMS-EA integration can also be used to elaborate the solution for the problem (II) from their list completed in Section 2.3.

5 Adaptive Onto-CBR Approach to Advanced Incidents Management in ITSM

In order to solve the problem (III), namely: to provide *an advanced incidents management* in ITSMS, accordingly to our inter-disciplinary vision about the ITSMS-development in general, we propose to amalgamate the following design tasks (i)-(iv) listed below

(i) An *incident management* as a weak-formalized and complex task within the ITSMS-support for its customers can effective be resolved using one of the intelligent decision-support methods, e.g., using *CBR-method*;
(ii) To enhance a CBR-functionality, especially with respect to specific needs in a target organization, an appropriate *domain-ontology* should be elaborated and used combining with CBR;
(iii) Due to permanent changes in an IT-infrastructure of a given organization, and changes arising in its environment as well, such a domain-ontology has to be constructed as an *adaptive ontology*;
(iv) To provide a possibility for knowledge gathering and their reusing in ITSMS, some *e-Learning models and technologies* can be applied.

There are already the approaches elaborated to combine a CBR-method with ontologies [17,18], which allow to provide more efficiently a case-representation, to enhance case-similarity assessment, and to perform case-adaptation process for a new solution. From the other hand, an ontology-centered design for ITSM-services, and especially, for *Incident Management (IM)*, is also discussed in some recent publications in this domain. In particular, the proposed in [22] *Onto-IM* ontology is built according to ISO/IEC20000 for ITIL/ITSM [2], it includes such concepts as *Incident Management, Incident Record, Incident Entity,* etc. specified using OWL (Ontology Web Language) Restrictions for IM-Area, and the small example of this description is shown in Fig.3.



```
(contains some CreateIncidentRecord)
 and (contains some
CreateIncidentRecordCapability)
 and (contains some IncidentRecord)
 and (contains some
IncidentRecordStructurePolicy)
 and (contains some RecordIncident)
 and (contains some ServiceDeskEmployee)
```

Fig. 3. The excerpt of *Onto-IM* Restrictions (Source: [22], p.14)

These results provide a solution for the tasks (i)-(ii), but in our opinion to cover the task (iii) in more efficient way, with respect to permanent changes in IT-infrastructure of a target organization, an appropriate ontology has to be constructed as *adaptive* facility [23]. Thus the *Onto-TO* ontology given in Section 4 should be given as the following tuple.

$$Onto-TO^{(adapt)} =< C, R, P, W^{(C)}, W^{(R)} > \qquad (16)$$

where, additionally to the basic components of any ontology, namely: *C* – set of concepts, *R* – a set of relationships among these concepts, and *P* – a set of axioms (semantic rules), the following ones have to be defined: $W^{(C)}$ is a set of weight coefficients for concepts of *C,* and a $W^{(R)}$ is a set of weight coefficients for relationships of *R* respectively. Usage of these weight coefficients allows us, e.g., to take into account an appropriate importance grade in several types of ITSMS-customers (see Fig. 2) to provide IM - services for them.

In order to get all information resources needed for a completed solution of tasks (i)-(iv), we propose to apply some *e-Learning* models and technologies within an ITSMS, especially, for skills training and experience gathering by ITSMS-staff, designated in the *Onto-IM* ontology as *Incident Manager, ServiceDeskEmployee, Specialist* [22]. For this purpose an e-Learning ontology (*Onto-EL*) can be used, e.g., in [24] the *Onto-EL* is elaborated to build for learners their personal paths in e-learning environment, according to the selected *curriculum* (*Incident Management* in terms of *Onto-IM*), *syllabus* (*Incident Record*) and *subject* (*Incident Entity*).

Summarizing aforementioned issues concerning the tasks listed in (i)-(iv), the conceptual mechanism to provide an *advanced incidents management* (AIM) in ITSMS can be represented at the high-architectural level as the UML-package diagram [25] shown in Fig.4.

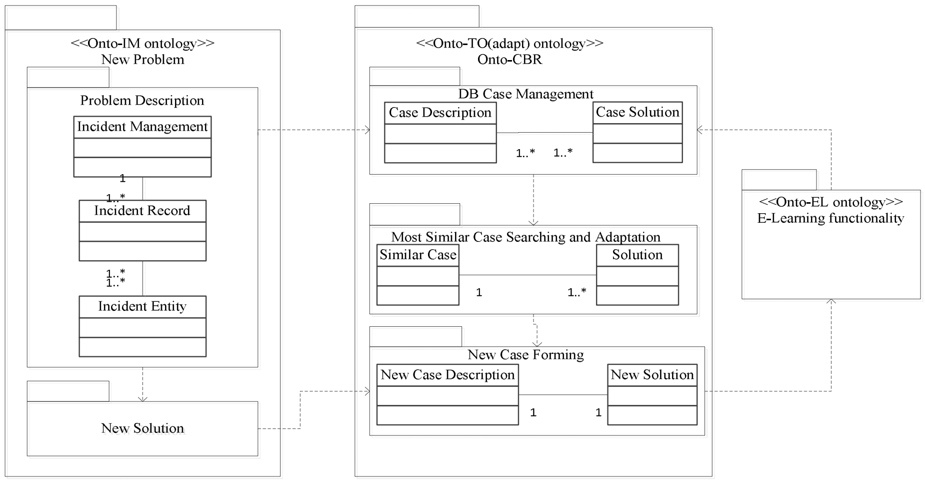


Fig. 4. The AIM – architectural framework (to compare with the scheme given in [26], p. 29)

The package *New_Problem* aggregates all functions needed to record and to process data concerning new problems and incidents using *Onto-IM* ontology as an information base for this purpose. The package *Onto-CBR* includes 3 sub-packages: to support DB case management, to provide most similar case searching and adaptation, and to form new case respectively. All these functions can be implemented using the data defined by $Onto - TO^{(adapt)}$ ontology. The package *E-Learning functionality* is used in this architectural solution as some kind of feedback loop in ITSMS-operating, in order to process and to reuse new positive skills and experience gathered by ITSMS-staff, and by customers of IT-services as well.

Below the approach to resolve the problem (I) from their list given in Section 2 is illustrated using the real case-study within our research and practice activities to apply ITSMS for IT-infrastructure management at the National Technical University "Kharkiv Polytechnic Institute" (www.kharkov.ua) referred in following as NTU "KhPI".

6 First Part in the Proof-of-Concept Project: Effective ITSM-Modules Configuring

It is to mention that exactly university- and /or a campus-domains are considered by many authors as a suitable example of ITSMS-usage (see, e.g., in [27-28]), because intensive research- and educational activities require a modern and well-organized IT-environment. That is why we also proved our approach to effectiveness estimation of alternative configurations of ITSM-modules using the test-case data collected at the NTU "KhPI".

6.1 Application Domain Description: IT-Infrastructure of NTU "KhPI"

The NTU "KhPI" is one of the largest technical universities of Ukraine located in the city of Kharkiv, which is the important industrial and cultural center at the East of the country. The University has about 22000 students, ca. 3500 of faculty members, and accordingly there is the advanced IT-infrastructure to support all educational and research tasks. Its simplified topology is depicted in Fig.5, and some characteristics are given in Table 3.

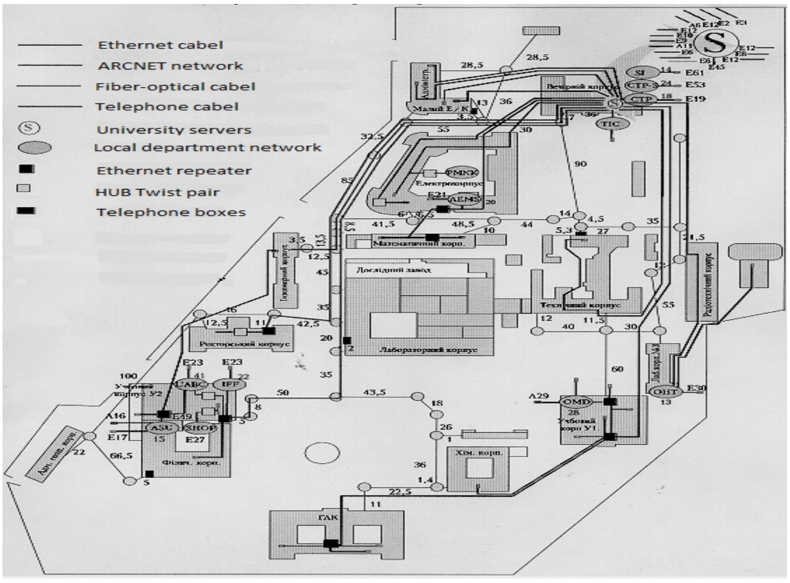


Fig. 5. The simplified scheme of IT-Infrastructure NTU "KhPI"

The daily IT-services provided for faculty members and students included all aspects of IT- technologies, which can be divided into following categories:

- Office automation and desktop applications;
- Wireless network communication;
- University Web-portal;
- E-learning and some others.

In cooperation with the IT-staff at the University IT control office we have analyzed retrospective data about some typical problem situations which have occurred, and about the corresponded incidents, which daily have been resolved within the direct communication with IT-service customers.

Table 3. Some technical data about IT-infrastructure of NTU "KhPI"

Parameters	**Values**
PCs in the network configuration	1525
User's accounts	2725
Buildings	23
Servers	62
Routers	84
Peripheral units (ca.)	6200

In this way the main types of ITSM-incidents and their initial problem situations were identified, and they are described in Table 4.

Table 4. Main types of ITSM-incidents and their related problem situations

1	**Incident type**	**Cause (problem situation)**
1	No Internet-connection at Dept or on local PC	- router was turned off; - network cable breaked or failure on router hardware; - incorrect network setup; - problems with software on local PC
2	High-loading of PC processor with a small number of active user's programs	- computer viruses - high degree of PC hard drive de-fragmentation.
3	Installing problems for new software	- computer viruses - absence of additional (middleware) software needed for installation.
4	Failure to send email	- incorrect setup of local network server (proxy) - problems with central e-mail server.
5	Troubles in the use of third-party software	- lack of specific configuration, - improper use of system services.

Basing on the analysis results obtained, we can apply the elaborated method to estimate alternative ITSMS-module configurations (see Section 3).

6.2 Customizing of the Elaborated Estimation Method: Alternative Configurations and Criteria Definition

According to the *Step 1* of the method presented in Section 3, the list of alternative ITSM-module configurations have to be defined, and in our case they are presented:

- X_1 = *Service Desk* subsystem (SDS) and *Incident Management* Module;
- X_2 = SDS, *Incident Management* Module and *Configuration Management* Module;
- X_3 = SDS, *Incident Management* Module and *Change Management* Module;
- X_4 = SDS, *Incident Management* Module and *Problem Management* Module.

On the next *Step 2,* according to the formulas (6) - (9), we determine the criteria for the quantitative evaluation of the proposed alternatives and their performance indicators, which are shown in Table.5.

Table 5. List of values for global and local criteria

Global and local criteria	**Semantics performance measurement criteria and target values**	**Insecure values**	**Effect. values**	**Possible values**
K_1	Effectiveness of incident management			
k_{11}	Average time incident resolution →minimum (min)	>30 min	20 min.	9999 minutes
k_{12}	Percentage of incidents resolved proactively →maximum (max)	0%	15%	0-100%
k_{13}	Percentage of incidents resolved at the first level of support →max	<65%	85%	100%
k_{14}	Percentage of incidents that have been resolved from the first time →max	<75%	90%	100%
K_2	Effectiveness of problem management			
k_{21}	The total number of incidents →min	60	5	999
k_{22}	The ratio of the number of resolved problems to all problems (%)→max	< 20%	80%	0-100%
k_{23}	% incidents, which can not be associated with any problem →min	>25%	10%	0-100%
K_3	Quality of customer support			
k_{31}	The degree of customer satisfaction →max	<3	4	0-5
k_{32}	The number of violations SLA→min	>30%	15%	0-100%
k_{33}	The number of services that are not covered SLA →min	> 35%	20%	0-100%

These criteria and their indicators (metrics) are taken from [29], and they are recommended to evaluate effectiveness of IT-infrastructure in any business organization.

For example, a value equal 10 for an alternative X_3 to criteria k_{14} (see Table 5) means, that the implementation of *Service Desk* and *Incident Management Module* will help to increase the ratio of incidents, which are resolved successfully, to its effective value of 90%, etc. The obtained in this way results are given in Table 6.

Table 6. The estimated values for the alternatives with respect to the defined criteria

	K_1: Effective incident management → opt			
	k_{11} (opt= 20min)	k_{12} (15%)	k_{13} (85%)	k_{14} (90%)
X_1	5	5	5	6
X_2	6	7	6	6
X_3	5	5	5	6
X_4	7	6	8	7
	K_2 : Effective problems management → opt			
	k_{21} (opt = 5)	k_{22} (80%)	k_{23} (10%)	
X_1	6	5	4	
X_2	7	6	5	
X_3	7	6	5	
X_4	7	7	7	
	K_3: Quality customer support → opt			
	k_{31} (opt =4)	k_{32} (15%)	k_{33} (20%)	
X_2	5	5	5	
X_3	7	8	6	
X_4	6	6	6	
X_2	7	7	6	

In order to implement the elaborated method with customized data introduced above, the special software tool was developed. Some its designing and implementation facets are presented below.

6.3 Software Tool to Support the Elaborated Approach

To design the data processing algorithm of the proposed approach to provide an effective ITSMS-configuration (see in Section 3), we should consider the possibility to use different expert methods as it was already mentioned in Section 2.3. This solution is shown in Fig. 6 as the appropriate UML- activity diagram [25].

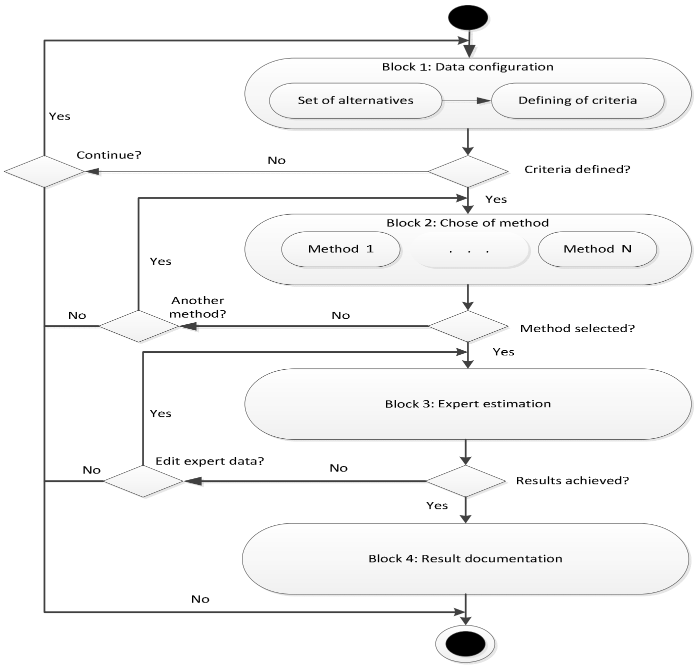


Fig. 6. The algorithm to implement the proposed estimation method

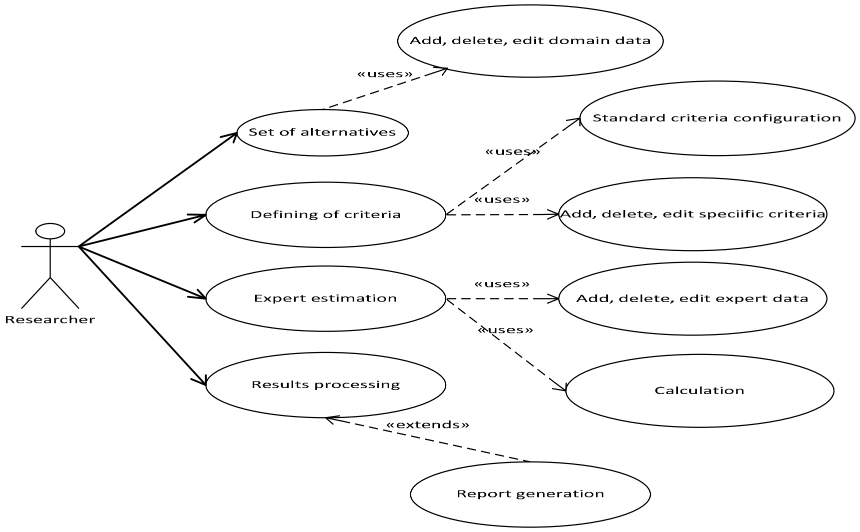


Fig. 7. The basic functionality of the elaborated software tool

Since our application is in the proof-of-concept version, it is implemented to process one expert method only, and its basic functionality is presented in Fig. 7 as UML use-case diagram [25]. According to this there are 4 main use cases called: “Set of alternatives", "Defining of criteria", "Expert estimation", and "Calculation", which can be divided into other sub-cases connected with <<*uses*>> and <<*extend*>> arrows.

To implement the design solutions defined in Fig. 6 and Fig. 7 in a software tool, the well-known MVC (Model-View-Controller) pattern is applied, therefore the following main classes are elaborated in form of UML-class diagram [25], which is shown in Fig. 8.

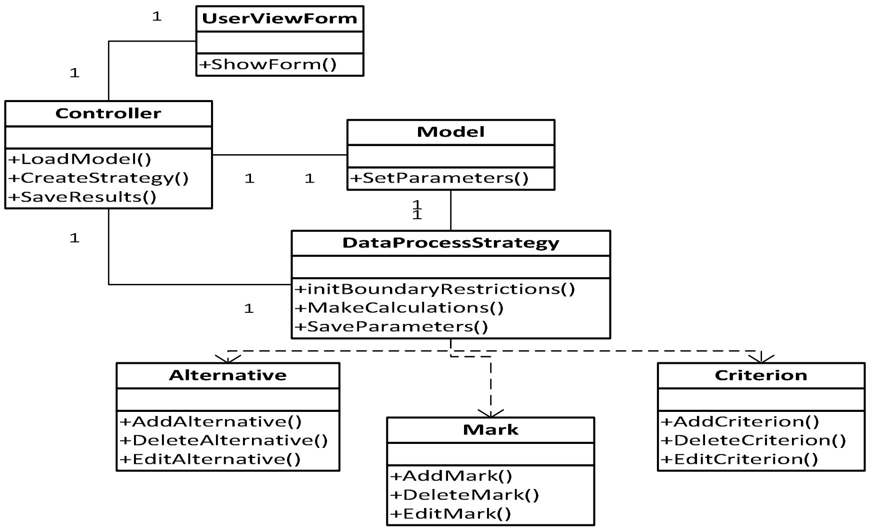


Fig. 7. Main classes in the implemented software tool

- Class *UserViewForm* is an interface that provides direct interaction with users;
- Class *Controller* manages the communication between the user and the system;
- Class *Model* is used for setting and configuring initial parameters;
- Class *DataProcessStrategy* implements all calculations corresponding to the estimation method;
- Class *Alternative* realizes all methods to process data related to alternatives to be estimated;
- Class *Criterion* and class *Mark* implement the same functions for global and local criteria, and for expert estimation values respectively.

The proposed tool is implemented successfully as Web-application using free-license software: DBMS MySQL [30] and PHP-technology [31], and in this way it can be used in future as a new functional module to be integrated with such open-source ITSM-solution like OTRS [9] (see in Section 2) or some others.

6.4 Results of Estimation and Their Analysis

To continue the usage of our method presented in Section 3 (*Step 3* and *Step 4* respectively) using the pair-wise comparison the weight coefficients of relative importance

(WCRI designated as $w(k_{i,j})$) for the local criteria regarding their global ones were determined:

- The WCRI values of the local criteria for the global criterion K_1: $w(k_{11}) =$ 0,239458, $w(k_{12}) = 0{,}239458$, $w(k_{13}) = 0{,}432749$, $w(k_{14}) = 0{,}088335$;
- The WCRI values of the local criteria for the global criterion K_2: $w(k_{21}) =$ 0,68334, $w(k_{22}) = 0{,}19981$, $w(k_{23}) = 0{,}11685$;
- The WCRI values of the local criteria for the global criterion K_3: $w(k_{31}) =$ 0,332516, $w(k_{32}) = 0{,}527836$, $w(k_{33}) = 0{,}139648$;
- The summarized WCRI values for the global criterion K_i: $K_1 = 0{,}527836$, $K_2 = 0{,}332516$, $K_3 = 0{,}139648$.

And finally, according to *Step 5* of this method (see Section 3), and using the multi-criteria ranking formulas (11) - (12), we obtain the following ultimate results of the effectiveness assessment for the considered alternatives (see Table 5), namely

$$X_1 = 0.537, X_2 = 0.671, X_3 = 0.578, X_4 = 0.727 \tag{17}$$

To confirm the reliability of the results given with formula (17), the comparative analysis with some "best practices" in ITSMS implementation was carried out, using the data of IDC-company [32]. In particular, IDC has reviewed approx. 600 organizations worldwide, which used ITSM for over a year, and in this study especially the prioritization issues of different ITSM-modules implementation were analyzed.

In Fig. 8 the result of the performed comparison is shown.

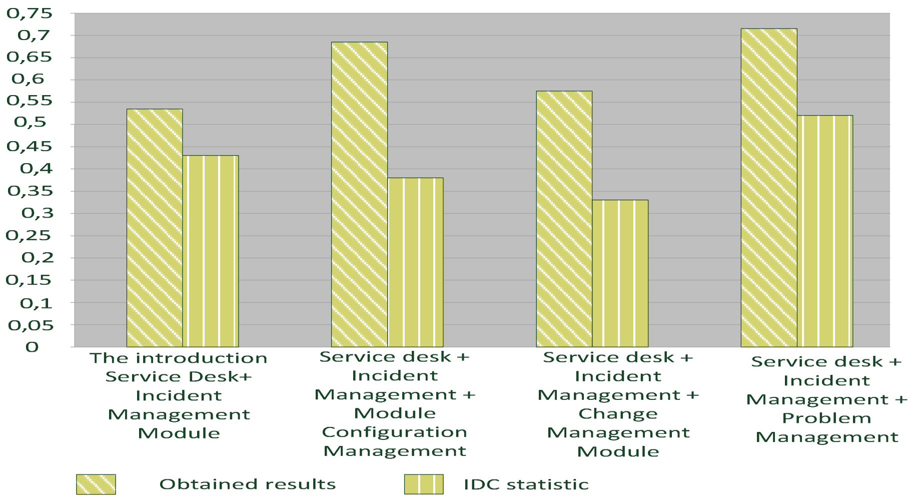


Fig. 8. Graphical representation of the obtained results

As we can see, to provide *Change Management* and *Configuration Management* is necessary to have within an IT-infrastructure database (DB) of IT-configurations, and DB of problem situations as well, these facilities are rather too expensive for the University, and therefore the implementation of these modules is not a priority task. The most effective ITSM-modules configuration for NTU "KhPI" includes an *Incident Management* module and a *Service Desk* subsystem.

7 Conclusions and Future Work

In this paper we have presented the intelligent approach to increase efficiency of ITSMS, which has to resolve 3 interconnected problems for its effective usage in a target organization (in our case: University):

(1) Providing an effective configuration of ITSM-modules according to its specific features and needs;
(2) Elaboration an integration framework for a given ITSMS with existing EA;
(3) Advanced incidents management in ITSMS.

To solve these problems in a comprehensive way the interdisciplinary integrated framework is elaborated, which includes: the expert method for multi-criteria ranking of alternative ITSM-modules configurations, the ontological specifications for ITSMS-EA integration, and an approach to advanced incident management based on the combination of adaptive ontologies and CBR-methodology.

To implement the first part of this approach the appropriate software tool was elaborated, and its applicability was tested successfully using real-word data in the ITSM proof-of-concept project performed in 2011-12 at the National Technical University “Kharkiv Polytechnic Institute”.

In future we are going to elaborate more detailed models and algorithms for the problems (2)-(3) listed above, to implement and to test the appropriate software solutions for them using such technologies as OWL, BPMN, XML / XLST, and Web-services.

References

1. Office of Government Commerce. ITIL Library, London (2003)
2. International Organization for Standardization. ISO/IEC 20000-1,2: Information Technology-Service Management, Part 1, 2. Geneva, Switzerland: ISO/IEC (2005)
3. Braun, C., Winter, R.: Integration of IT Service Management into Enterprise Architeture. In: Proceeding of SAC 2007, Seoul, Korea (2007)
4. Radhakrishnan, R.: ITSM Frameworks and Processes and their Relationship to EA Frameworks: A White Paper. IBM Global Technology Services (2008)
5. Valiente, M.-C., Vicente-Chicote, C., Rodriguez, D.: An Ontology-based and Model-driven Approach for Designing IT Service Management Systems. Int. J. Service Science, Management, Eng. and Techn. 2(2), 65–81 (2011)
6. Valiente, M.-C., Barriocanal-Garcia, E., Sicilia, M.-A.: Applying an Ontology Approach to IT Service Management for Business-IT Integration. J. on Knowledge-Based Systems 28, 76–87 (2012)
7. Official Web-site of Protocol, Ltd. company, `http://www.protocolsoftware.com/hp-openview.php`

8. Official Web-site of BMC Software company, `http://www.bmc.com/products/reme-dy-itsm/solutions-capabilities/it-service-management-suite.html`
9. Official Web-site of OTRS Group company, `http://www.otrs.com`
10. Official Web-site of SourceForge code repository, `http://sourceforge.net`
11. Tkachuk, M.V., Sokol, V.Y.: Some Problems on IT-infrastructure Management in Enterprises: State-of the-Art and Development Perspective. East-European J. on Advanced Technologies 48(6/2), 68–72 (2010) (in Russian)
12. Official Web-site of the Cleverics company, `http://www.cleverics.ru/en`
13. Saaty, T.L.: Fundamentals of the Analytic Hierarchy Process. RWS (2000)
14. Jabrailova, Z.Q.: A Method of Multi-criteria Ranging for Personnel Management Problem Solution. Artificial Intelligent 56(4), 130–137 (2009) (in Russian)
15. Valiente, M.-C., Barriocanal-Garcia, E., Sicilia, M.-A.: Applying Ontology-Based Models for Supporting Integrated Software Development and IT Service Management Processes. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 42(1), 61–74 (2012)
16. Tehrani, A., Mohamed, F.: A CBR-based Approach to ITIL-based Service Desk. J. Emerging Trends in Computing and Information Sciences 2(10), 476–484 (2011)
17. Lopez-Fernandez, H., Fdez-Riverola, L., Reboiro-Jato, M.: Using CBR as Design Methodology for Developing Adaptable Decision Support Systems. Technical report, University of Vigo, Spain (2011)
18. Prentzas, J., Hatzilygeroudis, I.: Combinations of Case-Based Reasoning with Other Intelligent Methods. J. Hybrid Intelligent Systems, 55–58 (2009)
19. Rodríguez-García, D., Barriocanal, E., Alonso, S., Nuzzi, C.: Defining Software Process Model Constraints with Rules Using OWL and SWRL. J. Soft, Eng. Knowl. Eng. 20(4), 533–548 (2010)
20. Prieto, A.E., Lozano-Tello, A.: Use of Ontologies as Representation Support of Workflows. J. Network and Systems Management 17(3), 309–325 (2009)
21. Kang, D., Lee, J., Choi, S., Kim, K.: An Ontology-based Enterprise Architecture. J. Expert Systems with Applications 37(2), 1456–1464 (2010)
22. Pansa, I., Reichle, M., Leist, C., Abeck, S.: A Domain Ontology for Designing Management Services. In: 3rd Int. Conf. on Advanced Service Computing, pp. 11–18 (2011)
23. Litvin, V.: Multi-agent Decision Support Systems Based on Precedents that Use of Adaptive Ontology. J. Artificial Intelligent 54(2), 24–33 (2009) (in Ukrainian)
24. Chung, Í.-S., Kim, J.-M.: Learning Ontology Design for Supporting Adaptive Learning in e-Learning Environment. In: IPCSIT 2012, Singapore, vol. 27, pp. 148–152 (2012)
25. Official Web-site of IBM company, `http://www-1.ibm.com/software/rational/uml/`
26. Suh, H., Lee, J.: Ontology-based Case-Based Reasoning for Engineering Design. Technical report, Design Research Group Manufacturing Engineering Lab (2008)
27. Boursas, L., Hommel, W.: Efficient Technical and Organizational Measures for Privacy-aware Campus Identity Management and Service Integration. In: Lillemaa, T. (ed.) EUNIS 2006, Tartu, Estonia, pp. 425–434 (2006)
28. Knittl, S., Hommel, W.: SERVUS@TUM: User-centric Service Support and Privacy Management. In: Desnos, J.-F., Epelboin, Y. (eds.) EUNIS 2007, Grenoble, France, pp. 142–151 (2007)
29. Brooks, P.: Metrics for IT-service Management. Van Haren Publishing (2006)
30. Official Web-site of MySQL, `http://www.mysql.com/`
31. Official Web-site of PHP, `http://php.net/`
32. Official Web-site of International Data Corporation (IDC), `http://www.idc.com`

Developing and Optimizing Parallel Programs with Algebra-Algorithmic and Term Rewriting Tools

Anatoliy Doroshenko, Kostiantyn Zhereb, and Olena Yatsenko

Institute of Software Systems of National Academy of Sciences of Ukraine, Glushkov prosp. 40, 03187 Kyiv, Ukraine
{doroshenkoanatoliy2,zhereb}@gmail.com, oayat@ukr.net

Abstract. An approach to program design and synthesis using algebra-algorithmic specifications and rewriting rules techniques is proposed. An algebra-algorithmic toolkit based on the approach allows building syntactically correct and easy-to-understand algorithm specifications. The term rewriting system supplements the algebra-algorithmic toolkit with facilities for transformation of the sequential and parallel algorithms, enabling performance improvement. We demonstrate the usage of the proposed tools with a simple example of parallelizing sequential program and improving performance of a parallel program, and also discuss possible applications in larger real-world projects.

Keywords: **A**lgebra of algorithms, code generation, formalized design of programs, parallel computation, term rewriting.

1 Introduction

Nowadays uniprocessor systems are almost fully forced out by multiprocessor ones, as the latter allow getting a considerable increase of productivity of programs. Thus, the need of program parallelization arises [12]. There are libraries, such as pthreads, OpenMP, TBB and others [1], allowing developers to write parallel programs. Using these libraries a programmer manually divides code into independent sections, describes data exchange and synchronization between them. However, such a method has substantial defects, in particular, related to committing errors in program code and a time required for parallelization and debugging. Therefore, the parallelization process has to be automatized as much as possible, and in an ideal, should be carried out fully automatically, without participation of a programmer.

This chapter continues our research on automation of process of designing and development of efficient parallel programs, started in [2], [11], [12], [13]. Our approach is based on Integrated toolkit for Designing and Synthesis of programs (IDS) [2], [28]. The process of algorithm designing in IDS consists in the composition of reusable algorithmic components (language operations, basic operators and predicates), represented in Systems of Algorithmic Algebras (SAA) [2], [11], [28]. We used IDS for generation of sequential and parallel programs in Java and C++ on the basis of high-level algorithm specifications (schemes). To automate the transformations of algorithms and programs we use the term rewriting system Termware [10], [13]. The novelty of this chapter is 1) adjusting IDS to generate parallel code in Cilk++ language, which is an extension to the C and C++ programming languages, designed for multithreaded parallel computing [8] and 2) closer integration between IDS and Termware systems. The approach is illustrated on a recursive sorting algorithm (quick sort) and one-dimensional Brownian motion simulation.

The rest of this chapter is organized as follows. Section 2 outlines our approach by describing our algebra-algorithmic and term rewriting design tools. In Section 3, an example of transformations for parallelizing a simple program is provided. Section 4 provides more detailed example of optimizing transformations aimed at improving parallel performance of an algorithm. Section 5 reports on our preliminary results in applying our approach to real-world applications in meteorology and quantum chemistry domains. In Section 6 we compare our results with other similar approaches reported in literature. The chapter ends with conclusions and directions of future research.

2 Formalized Design of Programs in IDS and Termware

The developed IDS toolkit is based on System of Algorithmic Algebras (SAA), which are used for formalized representation of algorithmic knowledge in a selected subject domain [2], [11], [28]. SAA is the two-based algebra $SAA = <\{U, B\}; \Omega>$, where U is a set of logical conditions (predicates) and B is a set of operators, defined on an informational set; $\Omega = \Omega_1 \cup \Omega_2$ is the signature of operations consisting of the systems Ω_1 and Ω_2 of logical operations and operators respectively (these will be considered below). Operator representations of algorithms in SAA are called regular schemes. The algorithmic language SAA/1 [2] is based on mentioned algebra and is used to describe algorithms in a natural language form. The algorithms, represented in SAA/1, are called SAA schemes.

Operators and predicates can be basic or compound. The basic operator (predicate) is an operator (predicate), which is considered in SAA schemes as primary atomic abstraction. Compound operators are built from elementary ones by means of operations of sequential and parallel execution operators, branching and loops, and synchronizer `WAIT 'condition'` that delays the computation until the value of the `condition` is true (see also Table 1 in next section).

The advantage of using SAA schemes is the ability to describe algorithms in an easy-to-understand form facilitating achievement of demanded quality of programs. The IDS is intended for the interactive designing of schemes of algorithms in SAA and generating programs in target programming languages (Java, C++, Cilk++). In IDS algorithms are designed as syntactically correct programs ensuring the syntactical regularity of schemes. IDS integrates three forms of design-time representation of algorithms: regular schemes, SAA schemes (textual representation of SAA formulae) and flow graphs. For integration with Termware, in this chapter IDS was also adjusted on generation of programs in Termware language.

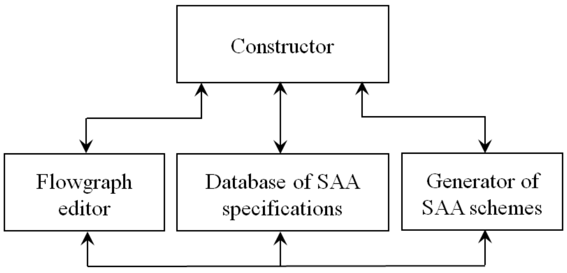


Fig. 1. Architecture of the IDS toolkit

The IDS toolkit consists of the following components (Fig. 1): constructor, intended for dialogue designing of syntactically correct sequential and concurrent algorithm schemes and generation of programs; flow graph editor; generator of SAA schemes on the basis of higher level schemes, called hyper-schemes [28]; and database, containing the description of SAA operations, basic operators and predicates in three mentioned forms, and also their program implementations.

The constructor is intended to unfold designing of algorithm schemes by superposition of SAA language constructs, which a user chooses from a list of reusable components for construction of algorithms. The design process is represented by a tree of an algorithm [2], [28]. On each step of the design process the constructor allows the user to select only those operations, the insertion of which into the algorithm tree does not break the syntactical correctness of the scheme. The tree of algorithm construction is then used for automatic generation of the text of SAA scheme, flow graph and the program code in a target programming language.

Example 1. We illustrate the use of SAA on Quicksort algorithm, which is given below in the form of SAA scheme. The identifiers of basic operators in the SAA scheme are written with double quotes and basic predicates are written with single quotes. Notice that identifiers (the descriptions in quotes) can contain any natural-language text explaining the meaning of operator or predicate. This text is not interpreted: it has to match exactly the specification in the database (however, since constructs are not entered manually, but selected from a list, the misspellings are prevented). The comments and implementations of compound operators and predicates in SAA schemes begin with a string of "=" characters.

```
SCHEME QUICKSORT_SEQUENTIAL ====
"main(n)"
==== Locals (
     "Declare an array (a) of type (int) and size (n)";
     "Declare a variable (i) of type (int)";
     "Declare a variable (end) of type (int)");
     "Fill the array (a) of size (n) with random
```

```
       values";
      "end := a + n";
      "qsort(a, end)";

"qsort(begin, end)"
==== IF NOT('begin = end')
         "Reduce (end) by (1)";
         "Reorder array (a) with range (begin) and (end)
          so that elements less than pivot (end) come
          before it and greater ones come after it; save
          pivot position to variable (middle)";
         "qsort(begin, middle)";
         "Increase (middle) by (1)";
         "Increase (end) by (1)";
         "qsort(middle, end)"
      END IF
END OF SCHEME QUICKSORT_SEQUENTIAL
```

For further usage, the descriptive identifiers are transformed into more concise terms, e.g. `"Declare an array (a) of type (int) and size (n)"` becomes `DeclareArray(a,int,n)`. Such mapping is also stored in the database (see Table 1 in Section 3).

To automate the transformation (e.g. parallelization) of programs we augment capabilities of IDS with rewriting rules technique [10], [13]. At the first step we construct high-level algebraic models of algorithms based on SAA in IDS (see also [2], [11], [28]). After high-level program model is created, we use parallelizing transformations to implement a parallel version of the program on a given platform (multicore in this chapter). Transformations are represented as rewriting rules and therefore can be applied in automated manner. The declarative nature of rewriting technique simplifies adding new transformations. Also transformations are separated from language definitions (unlike approach used in [21]), therefore simplifying addition of new transformations or new languages.

We use the rewriting rules system Termware [10], [13]. Termware is used to describe transformations of *terms*, i.e. expressions in a form $f(t_1,\ldots, t_n)$. Transformations are described as Termware *rules*, i.e. expressions of form `source [condition]-> destination [action]`. Here `source` is a source term (a pattern for match), `condition` is a condition of rule application, `destination` is a transformed term, `action` is additional action that is performed when rule fires. Each of 4 components can contain variables (denoted as `$var`), so that rules are more generally applicable. Components `condition` and `action` are optional. They can execute any procedural code, in particular use the additional data on the program. In particular, the `condition` component allows expressing *conditional rewriting* *rules*. Such rules first match a `source` term, and then check the specified condition for the found term. As an example, consider the following rule:

```
FOR($var,$start,$end,$body) [$end-$start>16] ->
                                   Unroll($var,$start,$end,$body)
```

This rule transforms the loop term into a term corresponding to an unrolled loop only if the number of iterations is greater than 16. The condition expression can use captured variables and basic arithmetic and logical operations. For more complex condition, a procedural code implemented in Java or C# could be called.

Termware could be compared to the functional programming languages such as Haskell. As discussed in [13], the main differences of Termware are as follows:

- Termware is not supposed to be used as a standalone language, but rather as a coordination part of a program written in Java or C#. Therefore, it lacks imperative features, such as I/O. Such features, if needed, can be implemented through actions.
- Unlike many functional languages, Termware is not strictly typed. This simplifies description of rules: if we need to implement a small local change in a complex term describing parse tree of some language, there is no need to build type structure describing all possible constructs of the language. However, absence of strict typing may have negative impact on performance.
- Termware could be described as a domain-specific functional language aimed at transformations of tree-like structures (terms). More generic functional languages also provide such capabilities, along with many others. However, Termware provides a simpler and more focused syntax, concentrating on a single task rather than supporting many different tasks. Therefore it may be easier to learn and use; on the other hand, a situation may arise when Termware capabilities are insufficient for implementing some complex transformation, while more generic functional language would be more suitable. However, we have not encountered such situations so far – Termware language has been expressive enough for our tasks.

3 Parallelizing a Sequential Program with Formal Transformations

IDS system performs generation of program code on the basis of an algorithm tree, received as a result of designing an algorithm in the IDS Constructor (see Section 2), and also code templates – implementations of basic operators and predicates in a target language (Java, C++, Cilk++), that are stored in IDS database. In the process of generation, IDS translates SAA operations into corresponding operators of programming language. Compound operators can be represented as subroutines (methods). IDS database contains various code patterns for generation of parallel programs, namely using WinAPI threads, Message Passing Interface (MPI), and Cilk++ operations [8]. For implementation of parallel version of our illustrative example (Quicksort algorithm), we used Cilk++ as it facilitates programming of recursive parallel programs [8]. Cilk++ is a general-purpose programming language, based on C/C++ and designed for multithreaded parallel computing.

Table 1 gives a list of main SAA operations and templates of their implementation in Termware and Cilk++, which are stored in the IDS database. The implementations contain placeholders like `^condition1^`, `^operator1^` etc., which are replaced with program code during the program generation.

For the purpose of transformation of some algorithm, IDS performs the generation of a corresponding term and developer specifies a set of rules for transformation. Then Termware carries out the actual transformation, the result of which can further be used for code generation in a programming language.

Table 1. The main SAA operations and templates of their implementation in Termware and Cilk++ languages

Text of SAA operation	Termware implementation	Cilk++ implementation
`“operator1”; “operator2”`	`then (^operator1^, ^operator2^)`	`^operator1^; ^operator2^`
`IF ‘condition’ THEN “operator1” ELSE “operator2” END IF`	`IF (^condition1^, ^operator1^, ELSE (^operator2^))`	`if (^condition1^){ ^operator1^ } else {^operator2^}`
`FOR '(var) from (begin) to (end)' LOOP “operator1” END OF LOOP`	`FOR (%1, %2, %3, ^operator1^ )`	`for (%1, %2, %3) { ^operator1^ }`
`(“operator1” PARALLEL “operator2”)`	`Parallel( ^operator1^, ^operator2^)`	`cilk_spawn ^operator1^; ^operator2^`
`WAIT ‘condition’`	`WAIT (^condition1^)`	`cilk_sync;`

In other words, there are two languages that could be used to specify algorithm schemes: IDS language (based on SAA) and Termware language. In our previous papers, we described using only one of these language: either IDS to specify algorithms (in [2], [11], [28]) or Termware to specify transformations (in [10],[12],[13]). In this chapter we report the joint usage of IDS and Termware languages. First, IDS is used to specify the algorithm, using constructor to build syntactically correct algorithm schemes. Then such schemes are transformed into Termware language, and transformation rules are described using Termware. While it may be possible to create a single unified language that would be used throughout the whole program design process, we decided to use two simpler languages and provide an automated translation between them. Such approach was easier to implement; also it allows for future independent evolution of both IDS and Termware languages.

Example 2. We will parallelize the sequential Quicksort algorithm (see Example 1), using IDS and Termware. For the parallelization, function `qsort` has to be transformed, so we generated the term for this function:

```
qsort(Params(begin, end),
     IF (NOT(Equal(begin, end)),
         then (Dec(end, 1),
         then (Partition(a, begin, end, end),
         then (CALL(qsort(begin, middle)),
         then (Inc(middle, 1),
         then (Inc(end, 1),
         CALL (qsort(middle, end))))))))
```

Then the operation of parallel execution of operations has to be added to this term. This is done by applying the following two Termware rules:

```
1. then(CALL($x), $y) -> Parallel (CALL($x), $y)
2. then($x1, Parallel($x2, $x3)) ->
              then($x1, then(Parallel($x2, $x3),
                    WAIT(AllThreadsCompleted(n))))
```

The first rule replaces the operation of sequential execution of operators with parallel execution. The second rule adds a synchronizer `WAIT(AllThreads Completed(n)`, which delays the computation until all threads complete their work. The result of the transformation is given below.

```
qsort(Params(begin, end),
IF(NOT(Equal(begin, end)),
   then (Dec(end, 1),
   then (Partition(a, begin, end, end),
   then (Parallel(
            CALL (qsort(begin, middle)),
            then (Inc(middle, 1),
            then (Inc(end, 1),
            CALL (qsort(middle, end))))),
         WAIT(AllThreadsCompleted(n)))))))
```

Thus, as a result of parallelization, the first operator (thread) of `Parallel` operation executes the operator `qsort(begin, middle)`, and the second one calls two `Inc` operators and `qsort(middle, end)`. Operation `WAIT(AllThreadsCompleted(n))` performs the synchronization of threads. The threads are created recursively; their quantity is specified as an input parameter of function `main`. Notice that these transformations are only valid if two `qsort` calls are independent. The system doesn't check this property: it has to be asserted by a developer. The correctness of the transformations is specified for the complete rule sets (the combinations of rules) and not for the individual rules.

The resulting parallel algorithm scheme Quicksort was used for generation of code in Cilk++ using IDS system. The parallel program was executed on Intel Core 2 Quad CPU, 2.51 GHz, Windows XP machine. Fig. 2 shows the program execution time in seconds. The speedup at execution of program with usage of 2, 3 and 4 processors was 2; 2.9 and 3.8 accordingly, which shows that the program has a good degree of parallelism and is scalable.

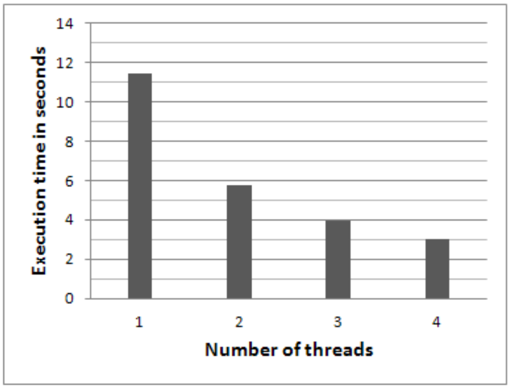


Fig. 2. The execution time of parallel Quicksort program on a quad-core processor; the size of input array is $5{\cdot}10^7$ elements

4 A Set of Transformations for Improving Parallel Performance

In this section we consider more complex transformations aimed at improving parallel performance of a given algorithm. These transformations are demonstrated using an algorithm for simulation of one-dimensional Brownian motion. In this case we start with an algorithm that is already parallel. This algorithm is represented as the following SAA scheme:

```
SCHEME BROWNIAN_PARALLEL ====

" main"
==== "Initialize variables";
     PARALLEL '(ThreadNum) from (0) to(NUM_THREADS - 1))'
     LOOP
         "Move particle (ThreadNum)"
     END LOOP;
     WAIT 'Threads complete (NUM_THREADS)';
```

```
"Move particle (ThreadNum)"
==== "Initialize random number generator";
    FOR '(counter) from (1) to (IterationAmount)'
    LOOP
        "(nextx) := Get new position of particle
        (ThreadNum) from current position (x) with
        probability (ProbabilityLimit) and crystal
        border (CrystalLen - 1)";
        IF NOT ('(nextx) = (x)')
            CriticalSection(Particle_CS)
            (
                 "Decrease (Crystal[x]) by (1)";
                 "Increase (Crystal[nextx]) by (1)";
            )
            "(x) := (nextx)";
        END IF
    END OF LOOP

" Get new position of particle (ThreadNum) from current
position (pos) with probability (limit) and crystal
border (right_border)"
==== "(res) := (pos)";
     "Generate random number (r) from uniform
     distribution on [0..1]";
     IF '(r) >= (limit)'
        AND NOT ('(pos) = (right_border)')
        "(res) := (pos + 1)"
    END IF
    IF '(r) < (limit)' AND NOT('(pos) = (0)')
        "(res) := (pos - 1)"
    END IF

END OF SCHEME BROWNIAN_PARALLEL
```

Notice that this version of algorithm is rather inefficient because of large number of synchronization operations (`CriticalSection(Particle_CS)`). In order to improve its performance, we can apply a number of different transformations to operator "`Move particle (ThreadNum)`". This operator has the following representation in Termware:

```
  MoveParticle( Params(ThreadNum),
       then(InitRandom,
       then(
       FOR(counter, 1, IterationAmount,
          then(Assignment(nextx,
               CalculateNewPosition(minus(CrystalLen, 1),
                        x, ProbabilityLimit, ThreadNum)),
          then(
          IF(logical_not(eq(nextx, x)),
             then(CriticalSection(Particle_CS,
                then(Decrement(ArrayElement(Crystal, x)),
                then(Increment(ArrayElement(Crystal,
nextx))))),
             Assignment(x, nextx))))))))
```

4.1 Transformations of Critical Sections

The simplest applicable transformations modify critical sections. They can be either expanded, to include more elements, or split into multiple critical sections. In the first case, consider the following rule system (`ExpandCS`):

1. `THEN(CriticalSection($x0, $x1), $x2) → CriticalSection($x0, THEN($x1, $x2))`
2. `THEN($x0, CriticalSection($x1, $x2)) → CriticalSection($x1, THEN($x0, $x2))`
3. `IF($x0, CriticalSection($x1, $x2)) → CriticalSection($x1, IF($x0, $x2))`

Rules 1 and 2 describe expanding along sequential operators; rule 3 includes conditional statements. In general case, other control structures could be included by adding rules similar to rule 3. After transformation according to ExpandCS rules, the algorithm has the following structure:

```
MoveParticle( Params(ThreadNum),
     then(InitRandom,
     then(
     FOR(counter, 1, IterationAmount,
        CriticalSection(Particle_CS,
          then(Assignment(nextx,
             CalculateNewPosition(minus(CrystalLen, 1),
                      x, ProbabilityLimit, ThreadNum)),
          then(
          IF(logical_not(eq(nextx, x)),
              then(Decrement(ArrayElement(Crystal, x)),
              then(Increment(ArrayElement(Crystal,
                                         nextx))))),
              Assignment(x, nextx))))))))
```

In transformed scheme, critical section encloses the whole loop body.

Another transformation allows splitting critical section into multiple independent critical sections. It is implemented using the following rules (`SplitCS`):

```
1. CriticalSection($name, then($x, $y)) →
   TODO_CriticalSection(TODO_Name($name, 0),
                              then($x, $y))
2. TODO_CriticalSection(TODO_Name($name, $n),
   then($x, $y)) → then(
      TODO_CriticalSection(TODO_Name($name, $n), $x),
      TODO_CriticalSection(TODO_Name($name, $n+1),$y))
```

Here, the first rule initializes transformation by changing a `CriticalSection(…)` term into a new term `TODO_CriticalSection(…)`. The second rule applies this term to each operator inside original critical section.

Notice that these rules use special sort of terms prefixed with TODO_. Such terms represent model elements that are expected to be further specified during later stages of transformation process. TODO-terms are treated in a special way by Termware engine. First, the list of such terms in current model is maintained, so that developer may specify additional rules that will complete them. Second, when all rules are applied and the list of TODO-terms is still non-empty, the system may either signal an error and fail transformation process, or remove all TODO_ prefixes. In this case, terms `TODO_Name($name, $n)` are transformed automatically into `Name($name, $n)` (this term is interpreted as building a new name by joining all subterms).

In this way, TODO-terms provide capability to specify generic transformations with variable parameters, and then complete these transformations by providing additional rules.

Using TODO-terms is related to the problem of specifying an order in which subterms are visited during rule application. The common approach used in rewriting systems is to specify *strategies* that describe how the rules are applied [10]. Strategies allow separating concerns of tree traversal (encoded in strategies) and node transformations (encoded in rules). There is similarity with functional languages: e.g. in Haskell tree transformation could be described as `maptree func tree`, where `maptree` describes an order in which nodes are visited, and `func` describes transformations of individual nodes.

However, sometimes it may be useful to allow the rules to override the traversal order imposed by the strategy. TODO-terms provide a simple way to implement this feature: some terms are just marked to be excluded from general transformations, and then transformed using more specific rules. The same effect might be achieved by modifying the strategy, but the rule-based approach is simpler, if less universal.

Another usage of TODO-terms is to simplify creation of confluent and terminating rule systems. Consider rules for `SplitCS` transformation without using `TODO_CriticalSection(…)` term:

```
1. CriticalSection($name, then($x, $y)) →
        CriticalSection(TODO_Name($name, 0),
                              then($x, $y))
```

```
2. CriticalSection(TODO_Name($name, $n),
    then($x, $y)) → then(
        CriticalSection(TODO_Name($name, $n), $x),
        CriticalSection(TODO_Name($name, $n+1),$y))
```

This rule system is not confluent, because left parts of rules 1 and 2 intersect. Denote `CriticalSection(…)=CS(…)` and `TODO_Name(…)=TN(…)`. Consider a term `CS(n,a;b;)`. Expected transformation is performed as follows:

```
CS(n,a;b;)
      →1 CS(TN(n,0),a;b;)
            →2 CS(TN(n,0),a); CS(TN(n,1),b)
```

However, this is not the only possible transformation path. After first step, rule 1 could be applied again:

`CS(`***`TN(n,0)`***`,a;b;)` $\to_1$ `CS(`***`TN(TN(n,0),0)`***`,a;b;)`

Such term is not reducible to terms that don't contain nested ***`TN`*** parts, therefore rule system is non-confluent. Furthermore, it is non-terminating, because left part of rule 1 matches its own right part, so it could be applied any number of times.

By providing TODO-prefixed version of term, we are able to break infinite application of rule 1. After first step, there are no more CS() terms, so transformation must proceed by applying rule 2. Also, new terms `TODO_CriticalSection(…)` are reverted back to `CriticalSection(…)` automatically, but it is done after rule system finished its processing, so it does not introduce infinite loops.

Notice that we started with a goal to improve parallel performance of a given algorithm. However, both `ExpandCS` and `SplitCS` transformations actually reduce performance. Expanding critical sections reduces opportunities to run code in parallel. Introducing additional critical sections increases synchronization overhead. Furthermore, `SplitCS` transformation does not lead to equivalent program: while each operator is executed inside its own critical section atomically, multiple operators are no longer executed as one atomic block. In Brownian motion simulation program, this means that there are moments when total number of particles is not preserved, between decrementing previous position and incrementing the next one. However, this inconsistent state is never observed, therefore `SplitCS` transformation could be applied in this case.

The real value of these transformations is that they enable application of additional transformations, and these additional transformations lead to more efficient implementation.

4.2 Using Specialized Synchronization Constructs

Consider `SeparatedLocks` scheme obtained by using `SplitCS` transformation. In this scheme, critical sections contain only single operator of increment or decrement:

```
CriticalSection(Particle_CS0,
                Decrement(ArrayElement(Crystal, x)))
CriticalSection(Particle_CS1,
                Increment(ArrayElement(Crystal, nextx)))
```

Such constructs could be implemented more efficiently by using atomic increment/decrement operators [1]. Such operators eliminate significant portion of overhead due to using critical section constructs. The transformation rules (`AtomicOperators`) are as follows:

```
CriticalSection($name,Increment($x)) ->
                                        IncrementAtomic($x)
CriticalSection($name,Decrement($x)) ->
                                        DecrementAtomic($x)
```

Now consider `BigLocks` scheme obtained by using `ExpandCS` transformation. Here, critical section is used inside loop body:

```
FOR(counter, 1, IterationAmount,
   CriticalSection(Particle_CS, $body))
```

Let's introduce a new synchronization construct: `ThreadSafeFor`. It acts as a regular for loop and guarantees that loop body is executed inside critical section. However, instead of acquiring a critical section for each loop iteration, it runs multiple iterations inside a single critical section. The number of iterations that are executed inside a single critical section is provided as an implementation parameter. The transformation is implemented using the following rule:

```
FOR($var, $start, $end, CriticalSection($name,$body) ->
    ThreadSafeFor($var, $start, $end,
                  $name, TODO_iterations,
                  $body)
```

Notice that this rule makes use of TODO-term to signal that this parameter should be specified during subsequent transformations. It could be specified by a developer based on his knowledge of algorithm and expected data size. Alternatively, it could be selected automatically – the system picks up several values from a given range, runs generated program for each value, measures its performance and selects the best value. For performance measurements, we have used a value of 1000.

Both `AtomicOperators` and `ThreadSafeFor` transformations introduce a new, more specialized synchronization operator instead of generic `CriticalSection`. From the viewpoint of algorithm developer, these new constructs look similar: each one provides more efficient implementation in specific case. However, they differ in their implementation. For `IncrementAtomic` and `DecrementAtomic` operators, many languages provide ready-to-use implementations. On the other hand, `ThreadSafeFor` usually has to be implemented by developer. Such implementation may be in form of library function that takes as parameters loop size, number of iterations in single critical section, and some object representing loop body. Alternatively, it could be implemented by another transformation in Termware:

```
ThreadSafeFor($var,   $start,   $end,   $name,   $iters,
$body) ->
FOR(Name($var,_outer),0,End($start,$end,$iters),
    CriticalSection($name,
        FOR(Name($var,_inner),0,$iters,
            then(RestoreVar($var,$start,$end,$iters)
            then($body)))))
```

Here, a single loop is split into 2 nested loops. Inner loop runs a specified number of iterations inside a single critical section. Outer loop runs this critical section a number of times sufficient to reach original number of iterations in a starting loop (denoted by `End($start,$end,$iters)`). There is an additional operator `RestoreVar($var,$start,$end,$iters)` that restores a value of loop counter from counters of inner and outer loop, and breaks out of loop if this value becomes larger than original upper limit `$end`. By implementing a `ThreadSafeFor` construct as a source-to-source transformation, instead of library call, we can avoid packaging loop body into some language specific object (such as function pointer in C, delegate in C# or anonymous function in languages that support them).

4.3 Algorithm-Specific Transformations

We have already considered two types of transformations. `ExpandCS` and `SplitCS` transformations are most generally applicable: they can be applied to almost any program that uses critical sections. However, they usually don't provide beneficial effects by themselves. On the other hand, `AtomicOperators` and `ThreadSafeFor` transformations provide noticeable speedup, but they require specific combination of operators to be applicable, such as `CriticalSection(Increment)` or `FOR(CriticalSection)`. Notice however that transformations of second type are still generic enough: they could be applied any time when required combination of operators is found. In general, transformations of both types are used together: transformations of first type are tried until they produce conditions that enable transformations of second type.

However, to obtain more significant performance improvements, we need to use transformations that take into account more specific details of a given algorithm. For example, an expert analyzing Brownian motion algorithm might notice that the real reason for low performance lies in unneeded synchronizations. On each iteration, algorithm tries to build a valid and consistent state of crystal that is shared between all threads. This requires large portions of code to run in a synchronized manner, inside critical sections. So far, the transformations that we used tried to make this synchronizations more efficient, either by making critical sections larger and less frequent (`ThreadSafeFor` transformation), or by making critical sections smaller and using more efficient implementations (`AtomicOperators` transformation). However, the real benefit could be obtained by eliminating large portions of synchronizations altogether.

An efficient strategy is to keep local copies of shared data for each thread, and perform infrequent global synchronizations between these copies. To implement such strategy, we need to specify how to initialize local copies from shared data, how to update these local copies on each iteration and finally how to synchronize local copies back into shared data. These details are specific to a given algorithm. However, the general outline of transformation could be implemented in Termware, with variable parts represented by TODO-terms. The rules that implement `LocalCopies` transformation are as follows:

```
1. FOR($var,$start,$end,$body,_MARK_Local) ->
                    then(TODO_InitLocalState,
                    FOR($var,$start,$end,$body))
2. CriticalSection($name,$body, _MARK_Local($var)) ->
        IF($var % TODO_BlockSize == 0,
           then(TODO_SaveLocalState($name,$body),
           TODO_InitLocalState,),
        ELSE(TODO_UpdateLocalState($body)))
```

Rule 1 inserts initialization of local state before the loop that is transformed. Rule 2 removes critical section and replaces it with a check that saves local state after a given number of iterations, otherwise updates local state. Notice that only the fragment of loop body that was initially contained in the critical section was changed: other fragments are assumed to work with local data already, since they didn't require synchronization.

The `LocalCopies` transformation differs from other transformations discussed above: it contains special MARK-terms such as `_MARK_Local`. These terms are similar to TODO-terms, as they are also processed in a special way by the Termware engine. However, their usage is different. TODO-terms are intended to represent variable parts of transformation that are specified *after* this transformation. On the other hand, MARK-terms represent special regions of input scheme that should be affected by transformation, and they are specified *before* the transformation. Therefore, TODO-terms are usually encountered in a *right-hand side* of a rule, while MARK-terms are usually in a *left-hand side*.

The presence of MARK-terms means that the transformation is not generically applicable. Instead, it affects only selected fragments of algorithms. The selection of such fragments can be performed either manually by a developer, or semi-automatically, e.g. when system provides a shortlist of possible locations and a developer selects some of them. (The process of selecting model fragments for transformation is described in more detail in [11]). The special processing for MARK-terms includes 1) ability to insert them from user interface and 2) ability to remove all unused marks after transformation is complete.

To complete the `LocalCopies` transformation, a developer must specify the missing operators represented by TODO-terms. For Brownian motion algorithm, this could be done using the following rules, each of them completing one of TODO-terms:

```
1. TODO_InitLocalState -> Assignment(firstx,x)
2. TODO_UpdateLocalState($body) -> NIL
3. TODO_SaveLocalState($name,
       then(Decrement(ArrayElement(Crystal,x)),
       $next)) ->
    CriticalSection($name,
       then(Decrement(ArrayElement(Crystal,firstx)),
       $next))
4. TODO_BlockSize -> 1000
```

In Brownian motion simulation algorithm, parts of local state that affect shared state are represented by a variable `x` that denotes current position of a particle in a crystal. On each iteration, this variable is used to modify number of particles in a shared array: the number of particles in the previous cell is decreased by 1, and the number of particles in the next cell is incremented by 1. Denote this operator as `Move(prev,next)`. It is easy to see that subsequent moves from position `a` to position `b`, and then from position `b` to position `c` are equivalent to a single move from starting position `a` to end position `c`

```
Move(a,b);Move(b,c) = Move(a;c)
```

Therefore, we need to store in local state only initial position of a particle (variable `firstx`); this is done by rule 1. Updating local state is not needed in this case, so rule 2 replaces `TODO_UpdateLocalState($body)` with a special empty symbol `NIL`. Rule 3 describes the process of updating shared state: we need to decrement the number of particles in the initial cell, and increment the number of particles in the final cell. However, since the second part was already present in original algorithm, it could be just copied from the body of critical section. Finally, in rule 4 we specify a number of iterations that are performed without synchronization. This parameter is similar to the one used in `ThreadSafeFor` transformation: in that case, too, there was a single critical section for each 1000 iterations, avoiding synchronization overhead. However, the critical difference between `ThreadSafeFor` and `LocalCopies` transformation is the size of this single critical section: in `ThreadSafeFor`, everything is executed inside it, therefore the performance cannot exceed that of sequential program. On the other hand, in `LocalCopies` program only a small part of program is executed inside critical section, and its performance can benefit from parallelization.

4.4 Performance Measurements Results

The results of performance measurements on a quad-core processor are provided on Fig. 3. As can be seen from it, most programs are actually less efficient than sequential program: only `LocalCopies` program achieves some speedup.

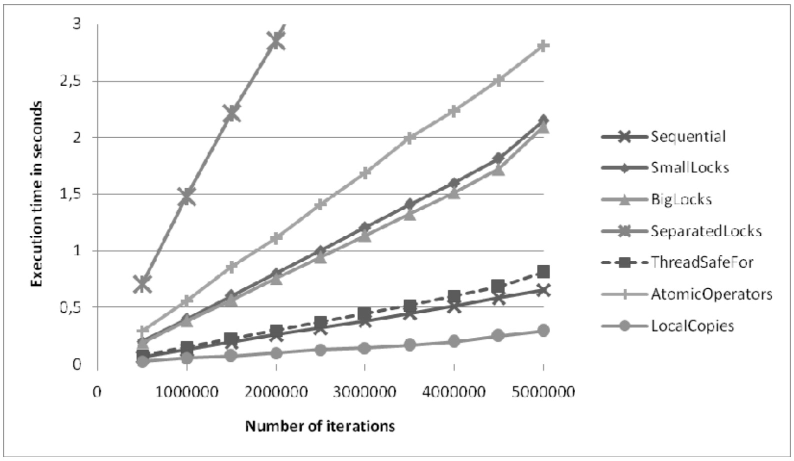


Fig. 3. The execution times of different implementations of Brownian motion program

Comparison of execution times demonstrates that the main reason of poor parallel performance is indeed synchronization overhead. Therefore, the worst program is `SeparatedLocks` that contains two critical sections per iteration. `AtomicOpe-rators` program is better because of more efficient synchronization constructs, but still very inefficient. `SmallLocks` and `BigLocks` program have almost the same performance, because they have the same number of critical sections (1 per iteration). `ThreadSafeFor` program is almost identical to `Sequential` program, because almost all computations are forced to be run on a single thread at a time. Finally, the only efficient parallel program is `LocalCopies`, because it is able to reduce the number of synchronization constructs without losing benefits of parallelism.

5 Real-World Applications

We have started applying our algebra-algorithmic approach in real-world domains of meteorology and quantum chemistry. This is still work in progress, and we are on early stages of these projects. Nevertheless this section briefly reports the results obtained so far, as well as our further expectations.

In subject domain of meteorology, we are working with a number of atmospheric models, including forecasting regional atmospheric processes based on a global model [9] and models of various dimensions based on convection-diffusion equations [24]. We start with numerical methods developed by our partners and already implemented as sequential programs, either in Fortran or in C languages. Our current goals can be described according to 2 main directions: 1) to develop efficient parallel versions of existing sequential programs, and 2) to extract and formalize subject domain knowledge (in form of operators and predicates of SAA) that would allow constructing new algorithms. So far, we have mostly progressed in a direction of parallelizing existing programs: versions for shared-memory systems (OpenMP) and distributed systems (MPI) have been developed, versions for GPUs (CUDA) is under development.

We are working on formalized transformations, including both generic rules such as represented in Sections 3 and 4, and subject domain specific rules. We also hope to implement an automatic solution for fine-tuning a number of implementation parameters. As discussed in Section 4, such auto-tuning system should be able to pick several values for a given parameter from a given range, generate and execute a program for each value, measure its performance and select the most efficient value.

As for extraction of subject-specific operators, we still have much work to do before our system becomes usable by subject domain specialists without programming qualification. So far, most of operators that can be used in the IDS toolkit are more *implementation-oriented* rather than *subject domain-oriented*. As an example, consider operators `ThreadSafeFor` or `IncrementAtomic` described in Section 4. Our current understanding is that such operators represent a lower level of program models compared to subject-domain specific operators such `"Move particle (number)"` as in Brownian motion example. Therefore, it is necessary to define such lower-level operators before they could be used in higher-level constructs. Another viable approach might be to work with algorithm descriptions that are not yet implemented as source code – e.g. represented informally as textual description or diagram or pseudo-code. In essence, such descriptions are already close to SAA schemes, they differ only by representation. Therefore we may first translate such descriptions manually to SAA schemes, and then implement similar facilities in IDS toolkit and provide them to subject domain specialists. Yet another approach might be to develop higher-level and lower-level facilities simultaneously. In this case there would be multiple implementations for a single high-level operator: starting with simple and less efficient, and gradually developing into more sophisticated versions.

In subject domain of quantum chemistry, we are working with a number of models that allow computing evolution of various configurations containing significant number of atoms (starting from thousands). Again, there are implementations in Fortran and our immediate goal is to port them to parallel platforms such as multicore [13], clusters, Grids and GPUs. The distinction of code that we are working with is its legacy nature: many procedures were written decades ago, long abandoned by original authors and subsequently modified by many other developers. Current scientists might not understand the details of some algorithm, but they are sure that its Fortran implementation works as intended. This creates a number of problems when trying to move these implementations to a new platform: we have to either use existing implementations "as is" – losing possible performance improvements and encountering problems on platforms where Fortran is unavailable; or try to reimplement the procedure on a new platform – with a possibility that it will not work in all cases exactly in the same way as the already proven implementation.

In this case, the viable approach seems to be reconstructing algorithm description form source code, by gradually increasing level of abstraction. Again, this may mean that first implementation-specific operators are reconstructed, and only later subject-domain specific constructs could be used. When a formal specification has been recovered from existing source code, it may be tested in following way. A new program, likely in a new language, is generated from the specification. This program is executed (on a new platform) alongside with original Fortran code (running on platform that supports it) on same inputs, and outputs are compared. With such approach, we cannot guarantee that all significant cases have been covered. However, by using static analysis on existing Fortran code, we might be able to discover some special cases (e.g. lower and upper bounds of arrays, different branches of conditionals, etc.)

While working with legacy Fortran code, we have discovered another benefit of high-level algebraic models and semi-automated process of constructing them: a possibility to discover errors in implementations. As an example, while trying to build a high level model of certain procedure, we have encountered an operator that wrote some computation results into a file. This operator was called very frequently because of its position deep within nested loops. We have executed a quickly constructed rewriting rule to find the places where data from this file was read, and didn't find such places. It turned out that this file was used with debugging purposes: original implementation was executed on unreliable hardware, and such frequent file writes were necessary to separate software errors from hardware faults. However, when the code was moved to more reliable platforms, the now-unneeded file writes were not removed. By removing them, we obtained immediate 9x speedup of sequential version – both from eliminating expensive disk writes, and from eliminating some computations that were only used to get data that was written to disk. Furthermore, by removing excessive disk writes, we removed bottleneck that prevented efficient parallelization of a given procedure.

Notice that, as far as we know, such optimization would not be possible to do with currently available completely automated tools: they would not be able to understand the meaning of file write operations and therefore remove it as unnecessary. Strictly speaking, the program obtained by removing write operations was not equivalent to original one: it differed in outputs by not creating one of output files. On the other hand, based on this isolated example, we don't have sufficient evidence to suggest that our semi-automated approach performed better than manual approach would perform. While we know that this bug stayed unfixed for quite a long time, it was probably because there was no organized effort to port the code to new platform, and not because the bug was especially hard to spot by a developer. However, we do have an expectation that our approach helps find such bugs, by focusing developer's attention on a small fragment of code and requiring to understand the meaning of this fragment in order to extract from it a high-level model element. We hope to validate this expectation during further development.

6 Related Work

The problem of automated synthesis of program code from specifications has been studied extensively and many approaches have been proposed [15], [16]. Important aspects of program synthesis include

1. format of inputs (specifications) – how specifications or other inputs for synthesis are provided;
2. methods for supporting concrete subject domains – how the proposed approach is specialized for a given domain, so that domain knowledge can be used to produce more efficient programs and/or to improve developer's productivity;
3. techniques for implementing transformation from specifications to output program code.

These aspects roughly correspond to 3 dimensions of program synthesis discussed in [16]: *user intent* describes specifications, *search space* restricts all possible programs to a manageable subset, and *search technique* describes how transformations are implemented.

For input specification, a popular option is using domain-specific languages (DSLs) that allow capturing requirements of subject domain. In [4], authors describe Pol, a system that transforms specifications in Alloy language [17] into a platform-specific code that targets different object-oriented frameworks. Similar approach is used in [22]: domain-specific models are transformed automatically into mobile applications for Android platform. Domain-specific languages can reduce the size of specifications significantly, because many details are already accounted for in domain-specific language constructs. However, creating a new DSL could require significant effort, and learning it might be hard for domain expert. To simplify DSL development, a new language can be embedded into an existing language that provides facilities required for DSL implementation. Recently, Scala has become popular as such host language [20], [26]. However, it is possible that simplifying DSL development in such way leads to a more complex language, limiting its adoption.

It is possible to make DSL easier to learn by using graphical modeling language. For instance, in [6] a graphical tool is described that operates on state charts and eventually generates web components. Paper [22] also develops graphical language including state charts and models of user interface screens. Such approach works well for small specifications, but becomes inconvenient as the size of models grows and their graphical representations don't fit a single page.

Some more formal approaches are also used, including formal specification languages [21], ontologies [7] and algebraic specifications [3], [18]. In work [21] authors describe a project to translate a requirements specification, expressed in SCR (Software Cost Reduction) notation, into C code. The translation is implemented using the Abstract Program Transformation System (APTS). In paper [7] the schema-based software synthesis system is considered, which uses generic code templates called schemas to represent general knowledge about software generation in a reusable ontological format. Authors of [3] propose algebraic specification for describing program features and provide automatic generations of a program once the needed features are described. In [18], programs implementing data types and their containers are generated using term algebras. Using such formalisms enables analysis and verification of specifications and generated code.

There are also approaches that provide specification not of program or algorithm, but of problem to be solved, in form of functional and nonfunctional constraints [20], [27], examples of input/output pairs [19], [23], or natural language descriptions [16]. In this case there is no simple mapping from specification to implementation; rather, there is a need to search through a space of possible programs to find one that corresponds to given constraints. Such search is usually exponential on size of specification. While there are heuristics that significantly reduce search space, such approaches are still limited to small programs.

Finally, specifications could be provided in form of existing implementation of an equivalent program, but lacking some important property that should be incorporated into a new program that is synthesized. Authors of [5] describe an approach for synthesis of SIMD program based on equivalent program without SIMD constructs. In [25], a non-deterministic parallel program is augmented with synchronization constructs to make it deterministic.

Another crucial aspect of program synthesis is specialization for subject domain. Some approaches are restricted to a single domain, such as statistical data analysis [14] or mobile application development [22]; others provide facilities for changing domain-specific parts, by using ontological descriptions [7], grammars [21], or by providing generic framework that is complemented by domain-specific tools [27].

Finally, an important aspect is transformation from input specification into source code in a target language. A transformation algorithm can be hand-coded [14], but it reduces flexibility of system. Therefore, transformation is often described in a declarative form, such as rewriting rules [21], visualized graph transformations [22], code templates [7]. More complex approaches require searching the space of possible programs [27], possibly using genetic programming or machine learning approaches [16], [23]. In [4], partial synthesis is proposed: generic parts of application are generated, and then completed with specific details manually.

In comparison, our approach uses algebraic specifications, based on Glushkov algebra of algorithms [2], but they can be represented in three equivalent forms: algebraic (formal language), natural-linguistic and graphical, therefore simplifying understanding of specifications and facilitating achievement of demanded program quality. Another advantage of IDS is a method of interactive design of syntactically correct algorithm specifications [2], [27], which eliminates syntax errors during construction of algorithm schemes. Specialization for subject domain is done by describing basic operators and predicates from this domain. Our approach uses code templates to specify implementations for operators and predicates. Program transformations, such as from sequential to parallel algorithm, are implemented as rewriting rules. Such separation simplifies changing subject domain or transformations.

7 Conclusion

We have described our approach of constructing efficient parallel programs using high-level algebra-algorithmic specifications and rewriting rules technique. Algebra-algorithmic toolkit IDS and rewriting rules engine Termware are combined to enable formal, yet easy-to-understand algorithm specifications and automate program synthesis and some stages of parallelization process. The combined development toolkit can be retargeted to various subject domains and implementation languages, as exemplified by Cilk++.

Our future research directions include using the proposed approach for large, real-world programs, as briefly discussed in Section 5. This would mean both improving our tools and working more closely with subject domain experts. Also the developed system could be further extended with automated code analysis facilities based on rewriting technique.

References

1. Akhter, S., Roberts, J.: Multi-Core Programming. Intel Press, Hillsboro (2006)
2. Andon, F.I., Doroshenko, A.Y., Tseytlin, G.O., Yatsenko, O.A.: Algebra-Algorithmic Models and Methods of Parallel Programming. Akademperiodika, Kyiv (2007) (in Russian)
3. Apel, S., et al.: An Algebraic Foundation for Automatic Feature-Based Program Synthesis. Science of Computer Programming 75(11), 1022–1047 (2010)
4. Bagheri, H., Sullivan, K.: Pol: Specification-Driven Synthesis of Architectural Code Frameworks for Platform-Based Applications. In: Proc. 11th Int. Conf on Generative Programming and Component Engineering, pp. 93–102. ACM, New York (2012)
5. Barthe, G., et al.: From relational verification to SIMD loop synthesis. ACM SIGPLAN Notices 48(8), 123–134 (2013)
6. Batory, D.: Program Refactoring, Program Synthesis, and Model-Driven Development. In: Adsul, B., Odersky, M. (eds.) CC 2007. LNCS, vol. 4420, pp. 156–171. Springer, Heidelberg (2007)
7. Bures, T., et al.: The Role of Ontologies in Schema-Based Program Synthesis. In: Proc. Workshop on Ontologies as Software Engineering Artifacts, Vancouver (2004)
8. Cilk Home Page, `http://cilkplus.org/`
9. Doroshenko, A.Y., Prusov, V.A.: Methods of Efficient Modeling and Forecasting Regional Atmospheric Processes. In: Faragó, I., et al. (eds.) Advances in Air Pollution Modeling for Environmental Security, pp. 143–152. Springer, Netherlands (2005)
10. Doroshenko, A., Shevchenko, R.: A Rewriting Framework for Rule-Based Programming Dynamic Applications. Fundamenta Informaticae 72(1-3), 95–108 (2006)
11. Doroshenko, A., Tseytlin, G., Yatsenko, O., Zachariya, L.: A Theory of Clones and Formalized Design of Programs. In: Proc. Int. Workshop on Concurrency, Specification and Programming (CS&P 2006), Wandlitz, Germany, pp. 328–339 (2006)
12. Doroshenko, A.Y., Zhereb, K.A., Yatsenko, Y.A.: On Complexity and Coordination of Computation in Multithreaded Programs. Problems in Programming 2, 41–55 (2007) (in Russian)
13. Doroshenko, A., Zhereb, K.: Parallelizing Legacy Fortran Programs Using Rewriting Rules Technique and Algebraic Program Models. In: Ermolayev, V., Mayr, H.C., Nikitchenko, M., Spivakovsky, A., Zholtkevych, G., et al. (eds.) ICTERI 2012. CCIS, vol. 347, pp. 39–59. Springer, Heidelberg (2013)
14. Fischer, B., Schumann, J.: AutoBayes: a System for Generating Data Analysis Programs from Statistical Models. J. Funct. Program. 13(3), 483–508 (2003)
15. Flener, P.: Achievements and Prospects of Program Synthesis. In: Kakas, A.C., Sadri, F. (eds.) Computational Logic: Logic Programming and Beyond. LNCS (LNAI), vol. 2407, pp. 310–346. Springer, Heidelberg (2002)
16. Gulwani, S.: Dimensions in Program Synthesis. In: 12th Int. ACM SIGPLAN Symposium on Principles and Practice of Declarative Programming, pp. 13–24. ACM, New York (2010)
17. Jackson, D.: Alloy: a lightweight object modelling notation. ACM Transactions on Software Engineering Methodology 11(2), 256–290 (2002)
18. Jacobs, S., Kuncak, V., Suter, P.: Reductions for Synthesis Procedures. In: Giacobazzi, R., Berdine, J., Mastroeni, I. (eds.) VMCAI 2013. LNCS, vol. 7737, pp. 88–107. Springer, Heidelberg (2013)

19. Kitzelmann, E.: Inductive Programming: a Survey of Program Synthesis Techniques. In: Schmid, U., Kitzelmann, E., Plasmeijer, R. (eds.) AAIP 2009. LNCS, vol. 5812, pp. 50–73. Springer, Heidelberg (2010)
20. Kneuss, E., et al.: On Integrating Deductive Synthesis and Verification Systems. Technical Report (2013)
21. Leonard, E.I., Heitmeyer, C.L.: Automatic Program Generation from Formal Specifications using APTS. In: Automatic Program Development. A Tribute to Robert Paige, pp. 93–113. Springer, Dordrecht (2008)
22. Mannadiar, R., Vangheluwe, H.: Modular Synthesis of Mobile Device Applications from Domain-Specific Models. In: Proc. 7th Int. Workshop on Model-Based Methodologies for Pervasive and Embedded Software, pp. 21–28. ACM, New York (2010)
23. Menon, A., et al.: A machine learning framework for programming by example. In: Proceedings of the 30th International Conference on Machine Learning (ICML 2013), pp. 187–195 (2013)
24. Prusov, V.A., Doroshenko, A.Y.: On efficient numerical solution of one-dimensional convection-diffusion equations in modelling atmospheric processes. International Journal of Environment and Pollution 32(2), 231–249 (2008)
25. Raychev, V., Vechev, M., Yahav, E.: Automatic Synthesis of Deterministic Concurrency. In: Logozzo, F., Fähndrich, M. (eds.) Static Analysis. LNCS, vol. 7935, pp. 283–303. Springer, Heidelberg (2013)
26. Rompf, T., Odersky, M.: Lightweight modular staging: a pragmatic approach to runtime code generation and compiled DSLs. Communications of ACM 55(6), 121–130 (2012)
27. Srivastava, S., Gulwani, S., Foster, J.S.: From Program Verification to Program Synthesis. In: Proc. 37th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, pp. 313–326. ACM, New York (2010)
28. Yatsenko, O.: On Parameter-Driven Generation of Algorithm Schemes. In: Proc. Int. Workshop on Concurrency: Specification and Programming (CS&P 2012), Berlin, Germany, pp. 428–438 (2012)

Asymptotical Information Bound of Consecutive Qubit Binary Testing

Anastasiia Varava and Grygoriy Zholtkevych

V.N. Karazin Kharkiv National University
School of Mathematics and Mechanics, 4, Svobody Sqr., 61022, Kharkiv, Ukraine
{nastia.varava,g.zholtkevych}@gmail.com

Abstract. The problem of estimating quantity of information, which can be obtained in the process of binary consecutive measuring a qubit state, is considered in the paper. The studied quantity is expressed as Shannon mutual information between the initial qubit state and the outcomes of a consecutive measurement. It is demonstrated that the maximum of information is reached by projective measurements. The maximum quantity of accessible information is calculated. It is proved that in the case of arbitrary binary test the maximum is reached asymptotically for consecutive measurements.

Keywords: Quantum information theory, quantum communication channels, accessible classical information, Shannon mutual information, pure qubit state, consecutive qubit binary testing.

1 Introduction

The theory of quantum information is a generalization of the classical information theory to the quantum world. Quantum information theory aimes to investigate what happens if information is stored in a state of a quantum system. By state of a physical system one means the mathematical description that provides a complete information on the system.

Talking about information from the point of view of quantum mechanics, we take into account some specific consequences of fundamental quantum postulates. The most important of them are the following: firstly, information, that is held in a state of a quantum system, cannot be read without changing the initial state. This property makes it largely inaccessible. Secondly, an arbitrary quantum state can not be cloned because of the no-cloning theorem. In other words, one can not create a perfect copy of the quantum system without already knowing its state in advance. And finally, quantum system can embody more information than classical, because a qubit can be in a superposition of basis state at the same time.

Industrial applications of quantum communication channels require developing sound methods for constructing them. It is well known that Shannon information theory is a mathematical background for such methods and the notion of Shannon mutual information [1,10] is a basic instrument for the classical theory of communication. Thus, studying informational quantities for quantum communication channels is a very important problem for building quantum information theory.

One of the well-known problems in this context is the problem of obtaining information on a quantum state. Suppose that one experementalist prepares an individual particle in a particular physical state and then sends it to another experementalist, who is not aware of the preparation procedure, but wishes to determine the state of the particle. By making observations on the received particle, the second experementalist can obtain information about the physical state by observing how it interacts with other wellcharacterized systems, such as a measuring instrument. The amount of this information depends strongly on whether the particle is macroscopic or microscopic. In the macroscopic, classical case, one can observe the particle without disturbing it, and determine its state. In quantum mechanics, on the contrary, it is impossible to learn the quantum state of any individual physical system, because each observation will disturb its state. The amount of the achievable information on the quantum system will never be sufficient to determine the given state accurately. This is, in particular, the basis of quantum key distribution for cryptography. However, effective techniques of obtaining information on the quantum state are quite important for different applications. Such techniques can be used for characterizing optical signals, as well as in quantum computing and quantum information theory to estimate the actual states of the qubits.

Thus, the amount of accessible information on the quantum system is an important information-theoretic quantity. To study the possibility of obtaining information it is convenient to express it as Shannon mutual information between the qubit state and the outcomes of a measurement.

One of the first investigated problems with respect to the discussed quantity is the so-called problem of distinguishing quantum states. It consists of the following: suppose that quantum system is prepared in a state described by one of the density operators ρ_i $(i = 1, ..., n)$ with probability p_i. The goal is to determine which state is given using a single measurement of the considered quantum system. Thus, the task is to find measurements providing the maximum of Shannon mutual information. This problem was investigated by Holevo [5], Davies [3] and many others. Particularly, Holevo proved in [5] that obtainable information I is less or equal than the so-called Holevo bound

$$I \leq S\left(\sum_i p_i \rho_i\right) - \sum_i p_i S(\rho_i) ,$$

where $S(\rho) = \mathrm{Tr}(\rho \log \rho)$ is the von Neumann entropy. It follows that the amount of information obtainable by a single measurement never exceeds $\log(\dim \mathcal{H})$, where $\mathcal{H}$ is a state space of a quantum system. This result is fundamental for the above mentioned problem. The work on this question was then continued by many reserchers. In particular, in [2] its extension to sequential measurements based on the properties of quantum conditional and mutual entropies was presented.

A more general question, the problem of estimating the amount of accessible information about quantum states, was also studied by several researchers. In particular, in [11] the brief review of obtained results is given. In the chapter we continue this investigation by considering a particular case. Imagine that we do not have any preliminary information on the possible states of a quantum system. In this case all possible pure states are equiprobable, so, we can not work with a finite set of initial states. Assume also that we have in our disposal a single measuring instrument. How much classical information can we now obtain?

In the paper we consider the simplest case of this problem. Suppose we have an arbitrary pure qubit state and an instrument doing binary test of it. Assume that all possible initial states are equiprobable. Our aim is to obtain classical information on the qubit state using only the given measuring instrument. Certainly, we want to get as much classical information as possible. So, this time we investigate the amount of information accessible in this case.

It is well known that in the case of performing a single measurement the maximum of classical information is obtainable by a projective measurement. We present a simple proof of this fact. In addition, we calculate the maximum value of information and then we show that in the case of arbitrary measuring instrument this value can be attained asymptotically using consecutive testing.

This paper is organized as follows: in Section 2 the problem is set formally and the formula for Shannon mutual information of two random variables is obtained. The first one describes the density of the parameter of the initial state, and the second one corresponds to the measurement result. In Section 3 a special kind of measuring instrument which performs a projective measurement is considered. It is demonstrated that in this case we obtain the maximum of accessible information. Further, in Section 4, it is proved that in general case consecutive measurements provide to attain this value asymptotically.

2 Qubits and Their Binary Testing

In quantum computing, a qubit is the quantum analogue of the classical bit. Like a bit, a qubit has two basis states: $|0\rangle$ and $|1\rangle$. The difference is that a qubit can be in a superposition of the basis states at the same time. This means that a pure qubit state can be represented as a linear combination of $|0\rangle$ and $|1\rangle$:

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle\ ,$$

where a and b are probability amplitudes and can in general both be complex numbers such that $|a|^2 + |b|^2 = 1$.

More precisely, a pure state of a qubit is a unit vector $|\psi\rangle$ in 2-dimensional Hilbert space $\mathcal{H}_2$ over the field of complex number. As usual, a density matrix equalled the ortho-projector on the subspace $\mathbb{C} \cdot |\psi\rangle$ of $\mathcal{H}_2$ is used to represent the pure state described by the vector $|\psi\rangle$. This ortho-projector is denoted by $|\psi\rangle\langle\psi|$. Hence, one can get

$$|\psi\rangle\langle\psi| = (a\,|0\rangle + b\,|1\rangle)(\overline{a}\,\langle 0| + \overline{b}\,\langle 1|) = |a|^2\,|0\rangle\langle 0| + a\overline{b}\,|0\rangle\langle 1| + \overline{a}b\,|1\rangle\langle 0| + |b|^2\,|1\rangle\langle 1|\,.$$

If we denote $|a|^2$ and $|b|^2$ by p and q respectively then we can rewrite the previous formula in the following form

$$|\psi\rangle\langle\psi| = p\,|0\rangle\langle 0| + e^{i\alpha}\sqrt{pq}\,|0\rangle\langle 1| + e^{-i\alpha}\sqrt{pq}\,|1\rangle\langle 0| + q\,|1\rangle\langle 1|\,,$$

where $p + q = 1$, $0 \le \alpha < 2\pi$.
Now using the representation $p = \frac{1}{2} - \theta$, $q = \frac{1}{2} + \theta$ where $-\frac{1}{2} \le \theta \le \frac{1}{2}$ one can identify $|\psi\rangle\langle\psi|$ with the matrix

$$\rho(\theta,\alpha) = \begin{pmatrix} \dfrac{1}{2} - \theta & e^{i\alpha}\sqrt{\dfrac{1}{4} - \theta^2} \\ e^{-i\alpha}\sqrt{\dfrac{1}{4} - \theta^2} & \dfrac{1}{2} + \theta \end{pmatrix}. \tag{1}$$

This matrix is called a density operator of the pure state.

In the general case a density operator for a qubit is a nonnegative definite operator on the space $\mathcal{H}_2$ such that its trace equals unity.

A binary test of a qubit is determined by a pair of operators $\{K(0), K(1)\}$ such that the equality $K(0)^\dagger K(0) + K(1)^\dagger K(1) = \mathbf{1}$ in the following meaning:

$$\begin{aligned} \Pr(0\,|\,\psi) &= \langle\psi|K(0)^\dagger K(0)|\psi\rangle \quad\text{and}\quad \Pr(1\,|\,\psi) = \langle\psi|K(1)^\dagger K(1)|\psi\rangle\,; \\ |\mathrm{Eff}(\psi\,|\,0)\rangle &= \frac{K(0)|\psi\rangle}{\sqrt{\langle\psi|K(0)^\dagger K(0)|\psi\rangle}} \quad\text{and}\quad |\mathrm{Eff}(\psi\,|\,1)\rangle = \frac{K(1)|\psi\rangle}{\sqrt{\langle\psi|K(1)^\dagger K(1)|\psi\rangle}}\,, \end{aligned}$$

where $|\psi\rangle$ describes the qubit state immediately before testing, $\Pr(s\,|\,\psi)$ is equal to obtain the outcome s as testing result, $|\mathrm{Eff}(\psi\,|\,s)\rangle$ describes the qubit state immediately after testing if the testing outcome equals s .

As known from theorem about polar decomposition [9, p. 28, Theorem 2.3] operators $K(0)$ and $K(1)$ can be represented as

$$K(0) = U(0)M(0)^{1/2} \text{ and } K(1) = U(1)M(1)^{1/2}\,, \tag{2}$$

where $M(0) = K(0)^\dagger K(0)$ and $M(1) = K(1)^\dagger K(1)$, $U(0)$ and $U(1)$ are unitary operators.

In this chapter we consider a binary tests subclass only. Its members are characterised by the condition $U(0) = U(1) = \mathbf{1}$. Such a qubit binary test has studied in [6].

Thus, let us introduce the following definition.

Definition 1. *A pair of nonnegative definite operators* $\mathbf{T} = \{M_0, M_1\}$ *on* $\mathcal{H}_2$ *such that* $M_0 + M_1 = \mathbf{1}$ *is called a qubit binary test.*
Probabilities of test outcomes are given by the next formulae

$$\Pr(s\,|\,\psi) = \langle\psi|M(s)|\psi\rangle\,,\ \textit{where } s = 0,1 \textit{ and } |\psi\rangle \textit{ describes the state of a qubit immediately before testing.} \tag{3}$$

The vector described the qubit state immediately after testing are given by the next formulae

$$|\mathrm{Eff}(\psi \mid s)\rangle = \frac{M(s)^{1/2}|\psi\rangle}{\sqrt{\Pr(s \mid \psi)}}, \text{ where } s = 0, 1 \text{ and } |\psi\rangle \text{ describes the state of a qubit immediately before testing.} \tag{4}$$

The mathematical model of a qubit binary test is described and studied in [12].

3 Shannon Mutual Information between the Qubit State and the Outcomes of a Consecutive Measurement

To set the problem formally we firstly need some basic definitions. We use a density operator as a mathematical model of a qubit state and describe the considering pure state of a qubit by the corresponding density operator $\rho(\theta, \alpha)$ represented by the matrix $\rho(\theta, \alpha)$ (see formula (1)).

Let $\varTheta$ be the equiprobability distribution on the segment $\left[-\frac{1}{2}, \frac{1}{2}\right]$. It is used to model uncertainty of our knowledge about parameter θ for an initial state of the qubit.

Let $\mathcal{M}$ be a set of all possible qubit binary tests. Consider $\mathbf{T} \in \mathcal{M}$, $\mathbf{T} = \{M_0, M_1\}$. Let $\{|0\rangle, |1\rangle\}$ be the ortho-normal basis in $\mathcal{H}_2$ such that $|0\rangle$, $|1\rangle$ are eigenvectors of the operator M_0 corresponding to eigenvalues $0 \le m_1 \le m_2 \le 1$ respectively. In this case the operators M_0 and M_1 are represented in a such way:

$$M_0 = \begin{pmatrix} m_1 & 0 \\ 0 & m_2 \end{pmatrix}, \quad M_1 = \begin{pmatrix} 1 - m_1 & 0 \\ 0 & 1 - m_2 \end{pmatrix}.$$

Denote by $\varSigma = \{0, 1\}$ the set of outcomes for the given qubit binary test. The probability distribution on this set is defined by the following formulae:

$$\Pr(0 \mid \rho) = \mathrm{Tr}(\rho \cdot M_0), \quad \Pr(1 \mid \rho) = \mathrm{Tr}(\rho \cdot M_1).$$

As we already mentioned, our aim is to obtain classical information value of the initial state as much as possible, using only the given instrument to measure. For this purpose we perform consecutive qubit binary testing. In other words, we repeat our measurement n times to observe the result of each iteration. After n iterations we obtain a sequence $\mathbf{x^{(n)}} = (x_1, \ldots, x_n)$, where $x_i \in \varSigma (i = 1, \ldots, n)$. As it is shown in [12],

$$\Pr(\mathbf{x^{(n)}} \mid \rho(\theta, \alpha)) = m_1^k (1 - m_1)^{n-k} \left(\frac{1}{2} - \theta\right) + m_2^k (1 - m_2)^{n-k} \left(\frac{1}{2} + \theta\right),$$

where k is a number of '1' in the outcomes sequence $\mathbf{x^{(n)}}$.

In further computations it is suitable to use some properties of Bernstein basis polynomials of degree n [8]:

$$B_{n,k}(z) = \binom{n}{k} z^k (1 - z)^{n-k},$$

so, we rewrite the last equation as follows:

$$\Pr(\mathbf{x^{(n)}} \mid \rho(\theta,\alpha)) = \binom{n}{k}^{-1} \left(\left(\frac{1}{2}-\theta\right) B_{n,k}(m_1) + \left(\frac{1}{2}+\theta\right) B_{n,k}(m_2)\right).$$

Let $\mathbf{X^{(n)}} \in \{0,1\}^n$ be a random variable describing an outcomes sequence. The density of $\mathbf{X^{(n)}}$ with respect to Θ is defined by the following formula

$$p(\mathbf{x^{(n)}}|\theta) = \Pr(\mathbf{x^{(n)}} \mid \rho(\theta,\alpha)).$$

The main objective of this section is to compute Shannon mutual information [1,10] of random variables $\mathbf{X^{(n)}}$ and Θ. It is equal to

$$I(\mathbf{X^{(n)}};\Theta) = H(\mathbf{X^{(n)}}) - H(\mathbf{X^{(n)}}|\Theta), \tag{5}$$

where $H(\mathbf{X^{(n)}})$ is Shannon entropy of $\mathbf{X^{(n)}}$, and $H(\mathbf{X^{(n)}}|\Theta)$ is conditional entropy [1,10]:

$$H(\mathbf{X^{(n)}}|\Theta) = \int_{-\frac{1}{2}}^{\frac{1}{2}} H(\mathbf{X^{(n)}}|\Theta=\theta)d\theta\ .$$

Let us compute the marginal distribution of the random variable $\mathbf{X^{(n)}}$, which corresponds to the joint density function $p(\mathbf{x^{(n)}},\theta)$. The latter is described in a such way:

$$p(\mathbf{x^{(n)}},\theta) = \binom{n}{k}^{-1} \left(\left(\frac{1}{2}-\theta\right) B_{n,k}(m_1) + \left(\frac{1}{2}+\theta\right) B_{n,k}(m_2)\right)\ .$$

The marginal distribution $p(\mathbf{x^{(n)}})$ is defined as follows:

$$p(\mathbf{x^{(n)}}) = \int_{-\frac{1}{2}}^{\frac{1}{2}} p(\mathbf{x^{(n)}},\theta)d\theta = \binom{n}{k}^{-1} \left(\frac{B_{n,k}(m_1)+B_{n,k}(m_2)}{2}\int_{-\frac{1}{2}}^{\frac{1}{2}} d\theta - \right.$$
$$\left.(B_{n,k}(m_1)-B_{n,k}(m_2))\int_{-\frac{1}{2}}^{\frac{1}{2}} \theta d\theta\right) = \binom{n}{k}^{-1} \frac{B_{n,k}(m_1)+B_{n,k}(m_2)}{2}\ .$$

Firstly, let us compute entropy of $\mathbf{X^{(n)}}$:

$$H(\mathbf{X^{(n)}}) = -\sum_{\mathbf{x^{(n)}}\in\{0,1\}^n} p(\mathbf{x^{(n)}})\log p(\mathbf{x^{(n)}}) = -\sum_{k=0}^{n}\binom{n}{k} p(\mathbf{x^{(n)}})\log p(\mathbf{x^{(n)}})\ .$$

Substituting the expression of marginal distribution, we obtain

$$H(\mathbf{X^{(n)}}) = \sum_{k=0}^{n}\left(\frac{B_{n,k}(m_1)+B_{n,k}(m_2)}{2}\right)\log\left(\binom{n}{k}\right) -$$
$$\sum_{k=0}^{n}\left(\frac{B_{n,k}(m_1)+B_{n,k}(m_2)}{2}\right)\cdot\log\left(\frac{B_{n,k}(m_1)+B_{n,k}(m_2)}{2}\right)\ . \tag{6}$$

Secondly, we can compute $H(\mathbf{X^{(n)}}|\Theta)$:

$$H(\mathbf{X^{(n)}}|\Theta) = -\int_{-\frac{1}{2}}^{\frac{1}{2}} H(\mathbf{X^{(n)}}|\Theta = \theta)d\theta = -\int_{-\frac{1}{2}}^{\frac{1}{2}} \sum_{k=0}^{n} \binom{n}{k} p(\mathbf{x^{(n)}}|\theta) \log p(\mathbf{x^{(n)}}|\theta) d\theta \ .$$

By direct calculations it is easy to check that

$$\begin{aligned} H(\mathbf{X^{(n)}}|\Theta) = \sum_{k=0}^{n} \left(\frac{B_{n,k}(m_1) + B_{n,k}(m_2)}{2}\right) \log\left(\binom{n}{k}\right) - \\ \sum_{k=0}^{n} \int_{-\frac{1}{2}}^{\frac{1}{2}} \left(\left(\frac{1}{2} - \theta\right) B_{n,k}(m_1) + \left(\frac{1}{2} + \theta\right) B_{n,k}(m_2)\right) \cdot \\ \log\left(\left(\frac{1}{2} - \theta\right) B_{n,k}(m_1) + \left(\frac{1}{2} + \theta\right) B_{n,k}(m_2)\right) d\theta \ . \end{aligned} \quad (7)$$

Now, using (5), (6) and (7), one can obtain the expression for Shannon mutual information

$$\begin{aligned} I(\mathbf{X^{(n)}}; \Theta) = \sum_{k=0}^{n} \Bigg(\int_{-\frac{1}{2}}^{\frac{1}{2}} \left(\left(\frac{1}{2} - \theta\right) B_{n,k}(m_1) + \left(\frac{1}{2} + \theta\right) B_{n,k}(m_2)\right) \cdot \\ \log\left(\left(\frac{1}{2} - \theta\right) B_{n,k}(m_1) + \left(\frac{1}{2} + \theta\right) B_{n,k}(m_2)\right) d\theta - \\ \frac{B_{n,k}(m_1) + B_{n,k}(m_2)}{2} \cdot \log\left(\frac{B_{n,k}(m_1) + B_{n,k}(m_2)}{2}\right)\Bigg) \ . \end{aligned}$$

It is easy to see that if k is such that $B_{n,k}(m_1) = B_{n,k}(m_2)$, then the corresponding summand is equal to zero:

$$\int_{-\frac{1}{2}}^{\frac{1}{2}} \left(B_{n,k}(m_1) \cdot \log B_{n,k}(m_1)\right) d\theta - B_{n,k}(m_1) \cdot \log B_{n,k}(m_1) = 0 \ .$$

Note that in the case of equality $m_1 = m_2$ the polynomials $B_{n,k}(m_1)$ and $B_{n,k}(m_2)$ are equal for each k, so, we can not obtain any information about the initial state. Thus we futher consider the case $m_1 < m_2$.

Let $\mathbf{A_n}$ be a set of non-negative integers defined as follows:

$$\mathbf{A_n} = \{k \in \{0, \ ..., \ n\} | B_{n,k}(m_1) \neq B_{n,k}(m_2)\} \ .$$

Denote by $\overline{\mathbf{A_n}}$ the complement of the set $\mathbf{A_n}$, i.e. $\overline{\mathbf{A_n}} = \{0, \ ..., \ n\} \backslash \mathbf{A_n}$. Now we can consider only values of k from this set, but we keep on working with all summands. We demonstrate further that it is more suitable. Instead of omitting zero summands, let us present them in the following way:

$$0 = \left(B_{n,k}(m_1) - \frac{B_{n,k}(m_1)}{2\ln(2)}\right) - \frac{2\ln 2 - 1}{2\ln 2} \cdot B_{n,k}(m_1) \ .$$

Thus, we have

$$I(\mathbf{X^{(n)}};\Theta) = \sum_{k \in \mathbf{A_n}} \left(\int_{-\frac{1}{2}}^{\frac{1}{2}} \left(\left(\frac{1}{2} - \theta\right) B_{n,k}(m_1) + \left(\frac{1}{2} + \theta\right) B_{n,k}(m_2) \right) \cdot \right.$$
$$\log \left(\left(\frac{1}{2} - \theta\right) B_{n,k}(m_1) + \left(\frac{1}{2} + \theta\right) B_{n,k}(m_2) \right) d\theta -$$
$$\left. \frac{B_{n,k}(m_1) + B_{n,k}(m_2)}{2} \cdot \log \left(\frac{B_{n,k}(m_1) + B_{n,k}(m_2)}{2} \right) \right) +$$
$$\sum_{k \in \overline{\mathbf{A_n}}} \left(\left(B_{n,k}(m_1) - \frac{B_{n,k}(m_1)}{2\ln(2)} \right) - \frac{2\ln 2 - 1}{2\ln 2} \cdot B_{n,k}(m_1) \right) \ .$$

After integration we obtain

$$I(\mathbf{X^{(n)}};\Theta) = \sum_{k \in \mathbf{A_n}} \left(- \frac{B_{n,k}(m_1) + B_{n,k}(m_2)}{4\ln 2} + \right.$$
$$\frac{B_{n,k}(m_2)^2 \log B_{n,k}(m_2) - B_{n,k}(m_1)^2 \log B_{n,k}(m_1)}{2(B_{n,k}(m_2) - B_{n,k}(m_1))} -$$
$$\left. \frac{B_{n,k}(m_1) + B_{n,k}(m_2)}{2} \cdot \log \left(\frac{B_{n,k}(m_1) + B_{n,k}(m_2)}{2} \right) \right) +$$
$$\sum_{k \in \overline{\mathbf{A_n}}} \left(\left(B_{n,k}(m_1) - \frac{B_{n,k}(m_1)}{2\ln(2)} \right) - \frac{2\ln 2 - 1}{2\ln 2} \cdot B_{n,k}(m_1) \right) \ .$$

Taking into account that for k from the set $\overline{\mathbf{A_n}}$ the equality $B_{n,k}(m_1) = B_{n,k}(m_2)$ is held, we can rewrite this formula as follows:

$$I(\mathbf{X^{(n)}};\Theta) = \sum_{k=0}^{n} \frac{2\ln 2 - 1}{4\ln 2} (B_{n,k}(m_1) + B_{n,k}(m_2)) - \sum_{k \in \overline{\mathbf{A_n}}} \frac{2\ln 2 - 1}{2\ln 2} B_{n,k}(m_1) +$$
$$\sum_{k \in \mathbf{A_n}} \left(\frac{B_{n,k}(m_2)^2 \log B_{n,k}(m_2) - B_{n,k}(m_1)^2 \log B_{n,k}(m_1)}{2(B_{n,k}(m_2) - B_{n,k}(m_1))} - \right.$$
$$\left. \frac{(B_{n,k}(m_1) + B_{n,k}(m_2)) \cdot \log \left(B_{n,k}(m_1) + B_{n,k}(m_2)\right)}{2} \right) \ .$$

Simplifying this expression and using the evident equality, $\sum_{k=0}^{n} B_{n,k}(z) = 1$, we get the following expression for mutual information:

$$I(\mathbf{X}^{(\mathbf{n})}; \Theta) = \frac{2\ln 2 - 1}{2\ln 2} - \sum_{k \in \overline{\mathbf{A_n}}} \frac{2\ln 2 - 1}{2\ln 2} B_{n,k}(m_1) + \frac{1}{2} \cdot \sum_{k \in \mathbf{A_n}} \left(\frac{B_{n,k}(m_2)^2 \log B_{n,k}(m_2) - B_{n,k}(m_1)^2 \log B_{n,k}(m_1)}{(B_{n,k}(m_2) - B_{n,k}(m_1))} - \frac{(B_{n,k}(m_2)^2 - B_{n,k}(m_1)^2) \cdot \log\left(B_{n,k}(m_1) + B_{n,k}(m_2)\right)}{(B_{n,k}(m_2) - B_{n,k}(m_1))} \right) . \quad (8)$$

4 Extremal Property of Projective Measurements

In this section we consider a special kind of measurement. Let $m_1 = 0$ and $m_2 = 1$. In this case the qubit binary test $\mathbf{T} = \{M_0, M_1\}$ looks as follows:

$$M_0 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} , \quad M_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} .$$

The first interesting property of this measurement is its repeatability. Actually, note that $M_i \cdot M_j = \delta_{i,j} \cdot M_i$. So, M_0 and M_1 are orthoprojectors, and $\mathbf{T}$ is a projective measurement. Thus, in this case the repeated measurements give the same result. We demonstrate further that due to this property consecutive testing do not provide any extra information. The second and the most remarkable property is that we can obtain the maximum of the accessible information if and only if we use a projective measurement.

First of all let us compute the amount of information obtainable by the considering measurement. In the considered case we have only two values of k such that the corresponding polymomials $B_{n,k}(m_1)$ and $B_{n,k}(m_2)$ are not equal. So, $\mathbf{A_n} = \{0, n\}$, and

$$\forall k \in \overline{\mathbf{A_n}} : B_{n,k}(m_1) = B_{n,k}(m_2) = 0 .$$

Now using (8) it is easy to see that considering $m_1 = 0$ and $m_2 = 1$ we obtain

$$I(\mathbf{X}^{(\mathbf{n})}; \Theta) = \frac{2\ln 2 - 1}{2\ln 2} .$$

Our next goal is to show that the obtained amount of information can not be reached using any other qubit binary test. For this purpose we investigate the function describing the accessible information (8).

At first let us rewrite this function in a more suitable way. Consider the general case, in which $0 < m_1 < m_2 < 1$. We will demonstrate futher that the exceptional cases, when $0 = m_1 < m_2 < 1$ or $0 < m_1 < m_2 = 1$, are similar to the latter.

Taking into account that in the general case $\forall k \in \{0, \ ..., \ n\}$ $B_{n,k}(m_1) > 0$, let us denote the ratio between $B_{n,k}(m_2)$ and $B_{n,k}(m_1)$ by a new function:

$$t_{n,k}(m_1, m_2) = \frac{B_{n,k}(m_2)}{B_{n,k}(m_1)} \ .$$

Thus, by direct calculations we obtain the following formula for mutual information:

$$I(\mathbf{X^{(n)}}; \Theta) = \frac{2\ln 2 - 1}{2\ln 2} - \frac{2\ln 2 - 1}{2\ln 2} \sum_{k \in \overline{\mathbf{A_n}}} B_{n,k}(m_1) - \frac{1}{2} \sum_{k \in \mathbf{A_n}} B_{n,k}(m_1) \cdot$$
$$\left(\frac{t_{n,k}(m_1, m_2)^2 \cdot \log\left(\dfrac{t_{n,k}(m_1, m_2)}{t_{n,k}(m_1, m_2) + 1}\right) + \log\left(t_{n,k}(m_1, m_2) + 1\right)}{1 - t_{n,k}(m_1, m_2)} \right) . \tag{9}$$

Let $f(t)$ be a function defined as follows:

$$f(t) = \frac{t^2 \cdot \log\left(\dfrac{t}{t+1}\right) + \log(t+1)}{1 - t} \ .$$

Now it is easy to see that the formula (9) can be written as

$$I(\mathbf{X^{(n)}}; \Theta) = \frac{2\ln 2 - 1}{2\ln 2} - \left(\frac{2\ln 2 - 1}{2\ln 2} \sum_{k \in \overline{\mathbf{A_n}}} B_{n,k}(m_1) + \right.$$
$$\left. \frac{1}{2} \sum_{k \in \mathbf{A_n}} B_{n,k}(m_1) \cdot f(t_{n,k}(m_1, m_2)) \right) . \tag{10}$$

We claim that in the considered case $(0 < m_1 < m_2 < 1)$ mutual information is less than

$$\frac{2\ln 2 - 1}{2\ln 2} \ .$$

To prove this fact it is enough to notice that the variable part of expression (10) is always negative. Actually, we know that $B_{n,k}(m_1) > 0$ and $t_{n,k}(m_1, m_2)$ is a ratio of two positive values, so, it is also positive. In addition, it is easy to show in the classical way that for all $t \in (0, 1) \cup (1, \infty)$ function $f(t)$ is greater than zero.

Now we need to consider the special case, in which $0 < m_1 < 1$, $m_2 = 1$. It is evident that the case when $m_1 = 0$, $0 < m_2 < 1$ is similar to the latter.

Suppose that $0 < m_1 < 1, m_2 = 1$. Thus, on the one hand, for all k we have $B_{n,k}(m_1) > 0$. On the other hand, for all $k < n$ $B_{n,k}(m_2) = 0$, and only for $k = n$ $B_{n,k}(m_2) = 1$. It is easy to see that in this case we obtain

$$I(\mathbf{X}^{(\mathbf{n})};\Theta) = \frac{2\ln 2 - 1}{2\ln 2} + \frac{1}{2}\cdot\left(\frac{B_{n,n}(m_2)^2 \log B_{n,n}(m_2) - (B_{n,n}(m_2)^2 - 1)\cdot\log(1 + B_{n,n}(m_2))}{B_{n,n}(m_2) - 1}\right) = \frac{2\ln 2 - 1}{2\ln 2} - \frac{1}{2}\cdot f(B_{n,n}(m_2)) < \frac{2\ln 2 - 1}{2\ln 2} .$$

Thus, now we know that

$$I(\mathbf{X}^{(\mathbf{n})};\Theta) \le \frac{2\ln 2 - 1}{2\ln 2} ,$$

and, in particular, the equality is held if and only if the considered binary test is projective.

5 Asymptotic Properties of Consecutive Measurements

In the previous section we have considered the case when the given qubit binary test is a projective measurement. We have proved that only this type of measurement allows to achieve the maximum of information about the initial state. As far as the measurement is projective, repeating of the measuring procedure does not provide any extra information. In addition, we have found the maximum value of the accessible information:

$$\max_{\mathbf{T}\in\mathcal{M},\ n\in\mathbb{N}}\{I(\mathbf{X}^{(\mathbf{n})};\Theta)\} = \frac{2\ln 2 - 1}{2\ln 2} .$$

In this section we return to considering of the general view of a qubit test, and we work with consecutive qubit testing. So, this time we investigate the dependence of the amount of information on n – the number of iterations. The objective of this section is to prove that the maximum of accessible information can be reached asymptotically by performing consecutive measurements using an arbitrary qubit binary test.

More strictly, our aim is to prove the next theorem:

Theorem 1. *Suppose we have a pure qubit state and we perform consecutive qubit binary testing using the given test $\mathbf{T} = \{M_1, M_2\}$. Then for arbitrary $\varepsilon > 0$ there exists a corresponding number of iterations $n(\varepsilon)$ such that for all subsequent iterations ($n > n(\varepsilon)$) the following inequality is held:*

$$\max_{\mathbf{T}\in\mathcal{M},\ m\in\mathbb{N}}\{I(\mathbf{X}^{(\mathbf{m})};\Theta)\} - I(\mathbf{X}^{(\mathbf{n})};\Theta) < \varepsilon .$$

In other words, as far as the mutual information can be written as

$$I(\mathbf{X}^{(\mathbf{n})};\Theta) = \frac{2\ln 2 - 1}{2\ln 2} - \left(\frac{2\ln 2 - 1}{2\ln 2}\sum_{k\in\overline{\mathbf{A_n}}} B_{n,k}(m_1) + \frac{1}{2}\sum_{k\in\mathbf{A_n}} B_{n,k}(m_1)\cdot f(t_{n,k}(m_1, m_2))\right) ,$$

we need to find $n_1(\varepsilon)$ such that for all $n > n_1(\varepsilon)$

$$\sum_{k\in\mathbf{A_n}} B_{n,k}(m_1)\cdot f(t_{n,k}(m_1,m_2)) < \frac{\varepsilon}{2}\ , \tag{11}$$

and $n_2(\varepsilon)$ such that for all $n > n_2(\varepsilon)$

$$\sum_{k\in\overline{\mathbf{A_n}}} B_{n,k}(m_1) < \frac{\varepsilon}{2}\ . \tag{12}$$

Therefore, for $n > n(\varepsilon) = \max\{n_1(\varepsilon), n_2(\varepsilon)\}$ both of these inequalities are held.

Let us fix a certain positive value of ε. At first we consider the left side of inequality (11). Let us divide the set $\mathbf{A_n}$ into two non-intersecting subsets:

$$\mathbf{\Gamma_n}(m_1) = \{k\in\mathbf{A_n}\} :\ \left|\frac{k}{n} - m_1\right| < \tilde{\delta}(m_1,m_2)\ ,$$

$$\mathbf{\Delta_n}(m_1) = \{k\in\mathbf{A_n}\} :\ \left|\frac{k}{n} - m_1\right| \geq \tilde{\delta}(m_1,m_2)\ ,$$

where $\tilde{\delta}(m_1,m_2)$ is a certain positive function.

It was demonstrated in [8], that for $0 < m_1 < 1$ and $\delta > 0$:

$$\sum_{k\in\mathbf{\Delta_n}(m_1)} B_{n,k}(m_1) \leq \frac{1}{4n\delta^2}\ . \tag{13}$$

On the one hand, it is easy to see that $f(t)$ is a bounded function. Suppose that for all $t\in(0,1)\cup(1,\infty)$ $f(t) < C$, where C is a certain positive constant. Thus we have

$$\sum_{k\in\mathbf{\Delta_n}(m_1)} B_{n,k}(m_1)\cdot f(t_{n,k}(m_1,m_2)) \leq \frac{C}{4n\tilde{\delta}^2(m_1,m_2)}\ .$$

So, we can choose a value $n_{1,1}(\varepsilon)$ such that for all $n > n_{1,1}(\varepsilon)$

$$\sum_{k\in\mathbf{\Delta_n}(m_1)} B_{n,k}(m_1)\cdot f(t_{n,k}(m_1,m_2)) < \frac{\varepsilon}{4}\ .$$

On the other hand, we can see that when k is close to $n\cdot m_1$, the value of $t_{n,k}(m_1,m_2)$ goes to zero as n goes to infinity. As far as $\lim\limits_{t\to0} f(t) = 0$, there exists a value $n_{1,2}(\varepsilon)$ such that for all $n > n_{1,2}(\varepsilon)$

$$\sum_{k\in\mathbf{\Gamma_n}(m_1)} B_{n,k}(m_1)\cdot f(t_{n,k}(m_1,m_2)) < \frac{\varepsilon}{4}\ .$$

Now let $n_1(\varepsilon)$ be a maximum of values $n_{1,1}(\varepsilon)$ and $n_{1,2}(\varepsilon)$. Thus, for all $n > n_1(\varepsilon)$ inequality (11) is held.

Finally, let us consider inequality (12). Note that in the case of inequality $m_1 < m_2$ the set $\overline{\mathbf{A_n}}$ contains at most one element. Actually, by construction

$k \in \overline{\mathbf{A_n}}$ if and only if $B_{n,k}(m_1) = B_{n,k}(m_2)$. Solving this equation for the variable k, we have

$$k_0 = n \cdot \frac{\ln\left(\dfrac{1-m_1}{1-m_2}\right)}{\ln\left(\dfrac{(1-m_1)\cdot m_2}{(1-m_2)\cdot m_1}\right)}$$

If n, m_1 and m_2 are such that k_0 is an integer then $\overline{\mathbf{A_n}} = \{k_0\}$. If not, the set $\overline{\mathbf{A_n}}$ is empty. It is easy to show that $\lim\limits_{n\to\infty} B_{n,k}(m_1) = 0$, so, we can easily find $n_2(\varepsilon)$ such that for all $n > n_2(\varepsilon)$ inequality (12) is held.

Now let us build a rigorous proof of the considering statement using this heuristic consideration. To do it, we firstly need several trivial propositions.

Proposition 1. *The following statements are correct:*

1. *for all $x \in (0,1)$ the inequality*

$$x > \ln(x+1)$$

is true;

2. *for all $x > 0$ the inequality*

$$\ln\left(\frac{x}{x+1}\right) < -\frac{1}{x+1}$$

is true too.

Proof. Consider the first statement. One can directly obtain it using the Tailor series expansion of $\ln(x+1)$:

$$\ln(x+1) = \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n} x^n = x + \sum_{n=1}^{\infty} x^{2n}\left(\frac{x}{2n+1} - \frac{1}{2n}\right) <$$

$$x + \sum_{n=1}^{\infty} x^{2n}\left(\frac{1}{2n+1} - \frac{1}{2n}\right) < x$$

Now let us prove the second statement. By using Euler's transform on Tailor series expansion shown above, we have

$$\ln\frac{\tilde{x}}{\tilde{x}-1} = \sum_{n=1}^{\infty} \frac{1}{n\tilde{x}^n} > \frac{1}{\tilde{x}}$$

for every $|\tilde{x}| > 1$.

Let $\tilde{x} = x + 1$, then $|\tilde{x}| > 1$ and

$$\ln\frac{x+1}{x} > \frac{1}{x+1},$$

so,

$$\ln\frac{x}{x+1} < -\frac{1}{x+1}.$$

□

Proposition 2. *Let $x, y \in (0,1)$ and $x \neq y$ then the following inequality is held:*

$$\left(\frac{x}{y}\right)^{y}\left(\frac{1-x}{1-y}\right)^{(1-y)} < 1 \, .$$

Proof. Let $g(x) = x^{c} \cdot (1-x)^{(1-c)}$ be a function of $x \in (0,1)$, where $c \in (0,1)$ is a certain constant. It is easy to prove in a standart way that

$$\max_{x \in (0,1)} g(x) = g(c)$$

Let $c = y$. Then for $x \neq y$:

$$x^{y} \cdot (1-x)^{(1-y)} < y^{y} \cdot (1-y)^{(1-y)},$$

so,

$$\left(\frac{x}{y}\right)^{y}\left(\frac{1-x}{1-y}\right)^{(1-y)} < 1.$$

□

Now we can prove the above formulated theorem.

Proof (of Theorem 1). As we already know, the mutual information can be presented as

$$I(\mathbf{X}^{(\mathbf{n})}; \Theta) = \frac{2\ln 2 - 1}{2\ln 2} - \left(\frac{2\ln 2 - 1}{2\ln 2} \sum_{k \in \overline{\mathbf{A_n}}} B_{n,k}(m_1) + \frac{1}{2} \sum_{k \in \mathbf{A_n}} B_{n,k}(m_1) \cdot f(t_{n,k}(m_1, m_2)) \right) \, .$$

We also know that

$$\max_{\mathbf{T} \in \mathcal{M},\ n \in \mathbb{N}} \{ I(\mathbf{X}^{(\mathbf{n})}; \Theta) \} = \frac{2\ln 2 - 1}{2\ln 2} \, .$$

To prove the theorem it is enough to show that there exists $n(\varepsilon)$ such that for all $n > n(\varepsilon)$

$$\sum_{k \in \mathbf{A_n}} B_{n,k}(m_1) \cdot f(t_{n,k}(m_1, m_2)) + \sum_{k \in \overline{\mathbf{A_n}}} B_{n,k}(m_1) < \varepsilon \, .$$

Consider an arbitrary $\varepsilon > 0$. Let us divide the set $\mathbf{A_n}$ into subsets $\mathbf{\Gamma_n}(m_1)$ and $\mathbf{\Delta_n}(m_1)$ in the following way:

$$\mathbf{\Gamma_n}(m_1) = \{ k \in \mathbf{A_n} \} : \ \left| \frac{k}{n} - m_1 \right| < \tilde{\delta}(m_1, m_2) \, ,$$

$$\mathbf{\Delta_n}(m_1) = \{ k \in \mathbf{A_n} \} : \ \left| \frac{k}{n} - m_1 \right| \geq \tilde{\delta}(m_1, m_2) \, ,$$

where $\tilde{\delta}(m_1, m_2)$ is a certain positive function of m_1 and m_2 defined further. Our aim is to prove that there exists such $n(\varepsilon)$ that for all $n > n(\varepsilon)$:

$$\sum_{k \in \mathbf{\Delta_n}(m_1)} B_{n,k}(m_1) \cdot f(t_{n,k}(m_1, m_2)) < \frac{\varepsilon}{4}\,, \tag{14}$$

$$\sum_{k \in \mathbf{\Gamma_n}(m_1)} B_{n,k}(m_1) \cdot f(t_{n,k}(m_1, m_2)) < \frac{\varepsilon}{4}\,, \tag{15}$$

$$\sum_{k \in \overline{\mathbf{A_n}}} B_{n,k}(m_1) < \frac{\varepsilon}{2}\,. \tag{16}$$

Firstly, let us consider inequality (14). We had already mentioned that for all $t \in (0,1) \cup (1, \infty)$ $f(t) \geq 0$, and it is easy to see that for considered values of t $f(t) < 2$. So, using the above mentioned property of Bernstein basis polynomials (13), we have:

$$\sum_{k \in \mathbf{\Delta_n}(m_1)} B_{n,k}(m_1) \cdot f(t_{n,k}(m_1, m_2)) \leq \frac{1}{2n\tilde{\delta}^2(m_1, m_2)}\,.$$

Let $n_{1,1}(\varepsilon)$ be sufficiently great. Then for all $n > n_{1,1}(\varepsilon)$ the considering inequality (14) is held.

Secondly, let us consider inequality (15). On the one hand, it is not hard to find such $\delta(\varepsilon)$ that

$$\forall t \in (0, \delta(\varepsilon)): \;\; f(t) = \frac{t^2 \cdot \log\left(\dfrac{t}{t+1}\right) + \log(t+1)}{1-t} < \frac{\varepsilon}{4}\,.$$

On the other hand, for sufficiently great values of n for all $k \in \mathbf{\Gamma_n}(m_1)$ we have $t_{n,k}(m_1, m_2) < \delta(\varepsilon)$. It follows that for great values of n we obtain

$$\sum_{k \in \mathbf{\Gamma_n}(m_1)} B_{n,k}(m_1) \cdot f(t_{n,k}(m_1, m_2)) < \frac{\varepsilon}{4} \cdot \sum_{k=0}^{n} B_{n,k}(m_1) = \frac{\varepsilon}{4}\,.$$

Let $\delta(\varepsilon) = \min\left(\frac{\sqrt{1+(\varepsilon/2\cdot\ln(2))^2}-1}{\varepsilon/2\cdot\ln(2)}; \frac{1}{2}\right)$. Then for $t \in (0,\ \delta(\varepsilon))$ we have

$$t < \frac{\varepsilon \cdot \ln(2)}{4}(1-t^2).$$

As far as $0 < t \leq \frac{1}{2} < 1$, we obtain

$$\frac{t - \frac{t^2}{t+1}}{(1-t)\cdot\ln(2)} = \frac{t}{(1-t^2)\cdot\ln(2)} < \frac{\varepsilon}{4}\,.$$

If we combine this with Proposition 1, then for all $t \in (0, \delta(\varepsilon))$ we have

$$f(t) = \frac{t^2 \cdot \ln\left(\dfrac{t}{t+1}\right) + \ln(t+1)}{(1-t)\cdot\ln(2)} < \frac{t + t^2 \cdot \left(-\frac{1}{t+1}\right)}{(1-t)\cdot\ln(2)} = \frac{t - \frac{t^2}{t+1}}{(1-t)\cdot\ln(2)} < \frac{\varepsilon}{4} \,.$$

Now we need to find such $n_{1,2}(\varepsilon)$ that for all $k \in \mathbf{\Gamma_n}(m_1) : t_{n,k}(m_1, m_2) < \delta(\varepsilon)$. By definition,

$$t_{n,k}(m_1, m_2) = \left(\frac{m_2}{m_1}\right)^k \cdot \left(\frac{1-m_2}{1-m_1}\right)^{n-k} \,.$$

As far as $\dfrac{m_2}{m_1} > 1$, $t_{n,k}(m_1, m_2)$ strictly increases with respect to k. So, if $\left|\dfrac{k}{n} - m_1\right| < \tilde{\delta}(m_1, m_2)$ then

$$t_{n,k}(m_1, m_2) < \left(\left(\frac{m_2}{m_1}\right)^{m_1 + \tilde{\delta}(m_1,m_2)} \cdot \left(\frac{1-m_2}{1-m_1}\right)^{1-m_1-\tilde{\delta}(m_1,m_2)}\right)^n \,.$$

Consider the right side of this inequality. Note that

$$\left(\frac{m_2}{m_1}\right)^{m_1 + \tilde{\delta}(m_1,m_2)} \cdot \left(\frac{1-m_2}{1-m_1}\right)^{1-m_1-\tilde{\delta}(m_1,m_2)}$$

srtictly increases with respect to $\tilde{\delta}(m_1, m_2)$. It is equal to 1 when $\tilde{\delta}(m_1, m_2) = \tilde{\delta}^*(m_1, m_2)$, where

$$\tilde{\delta}^*(m_1, m_2) = \left(\frac{\ln\left(\dfrac{1-m_2}{1-m_1}\right)}{\ln\left(\dfrac{(1-m_2)\cdot m_1}{(1-m_1)\cdot m_2}\right)} - m_1\right) \,.$$

According to Proposition 2,

$$\left(\frac{m_2}{m_1}\right)^{m_1} \cdot \left(\frac{1-m_2}{1-m_1}\right)^{1-m_1} < 1 \,,$$

we see that for $\tilde{\delta}(m_1, m_2) \in (0, \tilde{\delta}^*(m_1, m_2))$ we have

$$\left(\frac{m_2}{m_1}\right)^{m_1 + \tilde{\delta}(m_1,m_2)} \cdot \left(\frac{1-m_2}{1-m_1}\right)^{1-m_1-\tilde{\delta}(m_1,m_2)} < 1 \,.$$

Let

$$\tilde{\delta}(m_1, m_2) = \frac{\tilde{\delta}^*(m_1, m_2)}{2}.$$

Now it is easy to put $n_{1,2}(\varepsilon)$ such that for all $n > n_{1,2}(\varepsilon)$ inequality (15) is held.

Finally, let us find such $n_2(\varepsilon)$ that for $n > n_2(\varepsilon)$ condition (16) is satisfied. We have already seen that $|\overline{\mathbf{A_n}}| \leq 1$, and the equality is held when

$$k_0 = n \cdot \frac{\ln\left(\dfrac{1-m_1}{1-m_2}\right)}{\ln\left(\dfrac{(1-m_1)\cdot m_2}{(1-m_2)\cdot m_1}\right)}$$

is an integer. Let us denote the right side as $n \cdot c(m_1, m_2)$ and write k_0 as $k_0 = n \cdot c(m_1, m_2)$.

Referring to the standard way of proving the Stirling's formula, we can write the following inequality:

$$\sqrt{2\pi n} \cdot \left(\frac{n}{e}\right)^n \leq n! \leq e \cdot \sqrt{n} \cdot \left(\frac{n}{e}\right)^n .$$

It follows that

$$\binom{n}{k} \leq \frac{e}{2\pi}\sqrt{\frac{n}{k(n-k)}} \cdot \left(\frac{n}{k}\right)^k \cdot \left(\frac{n}{n-k}\right)^{n-k} .$$

So, it is now easy to see that

$$B_{n,k}(m_1) \leq \frac{e}{2\pi}\sqrt{\frac{n}{k(n-k)}} \cdot \left(\frac{n \cdot m_1}{k}\right)^k \cdot \left(\frac{n \cdot (1-m_1)}{n-k}\right)^{n-k} .$$

Substituting $k_0 = n \cdot c(m_1, m_2)$ for k in the last inequality, we get

$$B_{n,k_0}(m_1) \leq \frac{e}{2\pi}\sqrt{\frac{1}{n \cdot c(m_1,m_2)(1-c(m_1,m_2))}} \cdot \left(\left(\frac{m_1}{c(m_1,m_2)}\right)^{c(m_1,m_2)} \cdot \left(\frac{1-m_1}{1-c(m_1,m_2)}\right)^{1-c(m_1,m_2)}\right)^n . \quad (17)$$

It follows from Proposition 2 that

$$\left(\frac{m_1}{c(m_1,m_2)}\right)^{c(m_1,m_2)} \cdot \left(\frac{1-m_1}{1-c(m_1,m_2)}\right)^{1-c(m_1,m_2)} < 1 .$$

Now using (17) it is not hard to put $n_2(\varepsilon)$ such great that for $n > n_2(\varepsilon)$ inequality (16) is held.

Finally, let $n(\varepsilon) = \max\{n_{1,1}(\varepsilon), n_{1,2}(\varepsilon), n_2(\varepsilon)\}$. Now for all $n > n(\varepsilon)$

$$\max_{\mathbf{T}\in\mathcal{M},\ m\in\mathbb{N}}\{I(\mathbf{X^{(m)}};\varTheta)\} - I(\mathbf{X^{(n)}};\varTheta) < \varepsilon .$$

The theorem is proved. □

6 Conclusions

In the paper the problem of obtaining classical information about the pure qubit state using a single qubit binary test has been considered. It has been demonstrated that the maximum of information is reached if and only if the using measurement is projective. The maximum value of information has been calculated:

$$\max_{\mathbf{T}\in\mathcal{M},\, n\in\mathbb{N}} \left\{ I(\mathbf{X}^{(\mathbf{n})};\Theta) \right\} = \frac{2\ln 2 - 1}{2\ln 2}\,.$$

It follows, in particular, that to distinguish two arbitrary pure qubit states using a single binary test it is necessary to have at least four pairs of qubits prepared in the considered states.

It has been shown that the maximum of reachable information can be attained asymptotically using an arbitrary consecutive qubit binary test. Thus, if we have a certain measuring instrument performing a qubit binary test, we can obtain an amount of information arbitrary close to the maximum.

As known [4,7], Yu. Manin and R. Feynman proposed to use quantum systems to simulate others quantum systems. The results obtained in the paper show that this idea should be refined: one should take into account all dependences between an input data, a behaviour of a simulating system, and a structure of an output data.

Our further research will deal with generalizing results of the paper for the case of an n-level quantum system and a measurement with m outcomes.

References

1. Cover, T.M., Thomas, J.A.: Elements of Information Theory. John Wiley & Sons, Inc. (1991)
2. Cerf, N.J., Adami, C.: Accessible information in quantum measurement Caltech. Kellogg Lab & Santa Barbara. KITP & UC, Santa Barbara (1996)
3. Davies, E.B.: Information and Quantum Measurement. IEEE Trans. Inf. Theory IT-24, 596–599 (1978)
4. Feynman, R.P.: Simulating Physics with Computer. Int. J. Theor. Phys. 21, 467–488 (1982)
5. Holevo, A.S.: Bounds for the Quantity of Information Transmitted by a Quantum Communication Channel. Problemy Peredachi Informatsii 9(3), 3–11 (1973) (in Russian)
6. Holevo, A.S.: Statistical Structure of Quantum Theory. Springer, Berlin (2001)
7. Manin, Y.I.: Mathematics as metaphor: selected essays of Yuri I. Manin. AMS (2007)
8. Natanson, I.P.: Constructive function theory, vol. 1. Ungar, New York (1964)
9. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information, 10th Anniversary edn. Cambridge University Press, Cambridge (2010)
10. Shannon, C.E., Weaver, W.: The Mathematical Theory of Communication. University of Illinois Press, Urbana (1949)

11. Suzuki, J., Assad, S.M., Englert, B.-G.: Mathematics of Quantum Computation and Quantum Technology. In: Chen, G., Lomonaco, S.J., Kauffman, L. (eds.) Accessible information about quantum states: An open optimization problem. ch. 11. Chapman & Hall/CRC, Boca Raton (2007)
12. Thawi, M., Zholtkevych, G.M.: About One Model of Consecutive Qubit Binary Testing. Bull. Khark. Nat. Univ. 890(13), 71–81 (2010)

Availability Assessment of Computer Systems Described by Stiff Markov Chains: Case Study

Vyacheslav Kharchenko[1,2], Oleg Odarushchenko[2], Valentina Odarushchenko[1], and Peter Popov[3]

[1] National Aerospace University "KhAI", Kharkov, Ukraine
{v.kharchenko,v.odarushchenko}@khai.edu
[2] RPC "Radiy", Kirovograd, Ukraine
skifs2005@mail.ru
[3] Centre for Software Reliability, City University London, United Kingdom
ptp@csr.city.ac.uk

Abstract. Markov models are widely used in dependability assessment of complex computer-based systems. Model stiffness poses a serious problem in terms of both computational difficulties and accuracy of the assessment. Selecting an appropriate method and software package for solving stiff Markov models proved to be a non-trivial task. In this paper we provide an empirical comparison of two approaches– stiffness avoidance and stiffness-tolerance. The study includes several well-known techniques and software tools used for solving differential equations derived from stiff Markov models. In the comparison, we used realistic case studies developed by others in the past. The results indicate that the accuracy of the known methods is significantly affected by the stiffness of the Markov models, which led us to developing an algorithm for selecting the optimal method and tool to solve a given stiff Markov model. The proposed algorithm was used to assess the availability of a redundant FPGA-based safety controller.

Keywords: Markov chains, stiffness, stiffness-avoidance, stiffness-tolerance, availability, multi-fragmentation, FPGA-based safety controller.

1 Introduction

Nowadays complex computer systems are widely used in the fields of life-critical, safety-critical and business-critical applications. Part of those systems can be referred to as high availability systems (HAS) which are required to provide continuous operation during a defined period, despite the failures of hardware and software components and possible cyber attacks. Such high availability can be ensured by components redundancy, failure diagnostic and preventive means, dynamic reconfiguration, "hot" swap, etc.

Dependability assessment of complex computer systems is an essential part of the development process as it either allows for demonstrating that relevant regulations have been met (e.g. as in safety critical applications) and/or for making informed decisions about the risks due to automation (e.g. in applications when poor dependability may lead to huge financial losses). Achieving these goals, however, requires an *accurate* assessment. Assessment errors may lead to wrong or suboptimal decisions.

Dependability of computer systems is assessed using probabilistic models in which reliability and availability are typically used as measures of interest. Markov chains (MC) [2, 23] are often preferred to reliability block diagrams and fault trees as MC can handle well complex situations such as failure/repair dependencies and shared repair resources [3].

System modellers are often interested in *transient measures*, which provide more useful information than steady-state measures. The main computational difficulty when MC are used is the size of the models (i.e. their largeness), which is known to affect the accuracy of the transient numerical analysis. It is not unusual for modern complex systems to have a very large state space: often it may consist of tens of thousands of states. Additional difficulty in solving such models is the model *stiffness* [5], which is the focus of this paper. In practice stiffness in models of computer systems is caused by: i) in case of repairable systems the rates of failure and repair differ by several orders of magnitude [4]; ii) fault-tolerant computer systems (CS) use redundancy. The rates of simultaneous failure of redundant components are typically significantly lower than the rates of the individual components [4]; iii) in models of reliability of modular software the modules' failure rates are significantly lower than the rates of passing the control from a module to a module [4].

In practice it is useful to detect the model stiffness as early as possible. If the model is stiff, using a small integration step is usually a necessary step for obtaining an accurate solution. On the other hand, models with moderate stiffness may allow for obtaining accurate solutions without using a small integration step, thus saving computational resources. With some numerical methods decreasing the step of integration may be even counterproductive as it may simply not improve the solution accuracy.

During last three decades, a number of software tools have been developed and applied to solving models of complex systems. In general, they can be split into three main groups: specialised tools, e.g. developed by researchers to implement new solution methods, off-the-shelf mathematical packages and own (user) developed utilities (ODU). SHARP [24], Save [8], Reno [25], λPredict [25], Möbius [26] and etc. can be referred to as specialised tools. An example of ODU is the EXPMETH utility, developed by some of the authors of this paper, more than 15 years ago. It is based on the special *modified exponential numerical method* [21] and has been validated extensively on a range of models [18 - 20]. However it should be noted that any special utility developed for the exact purpose and used over the range of models can be used as ODU tool. A number of off-the-shelf mathematical software packages exist which can be used for solving Markov models, e.g. Maple (Maplesoft), Mathematica (Wolfram Research) and MATLAB (Mathworks) and MathCad (PTU), which use standard methods for solving differential equations. These math packages enjoy high reputation among the respective customers earned over several decades by providing a wide range of solutions and good support with regular updates.

Such variety of tools is extremely helpful in the process of system modeling but poses also a difficulty when it comes to choosing the most appropriate one for a specific assessment. The assessment tools must provide high confidence in the assessment results. In many cases various regulation bodies would require the tools used in the development to be certified to meet stringent quality requirements. The stiffness of an MC can make it difficult to meet this requirement. As every tool is limited in its properties and area of use a careful selection of the tools used to solve accurately and efficiently stiff MCs is needed. Stiffness usually requires that the modeler focuses on the mathematical details of the underplaying MC model. This view goes against the recommendations in one of the leading standards in the safety area IEC 61508-6 [27]. This standard asserts that methods for solving Markov models have been developed long ago and there is no point in trying to improve these methods[1]. We disagree with this assertion in the standard and demonstrate in this paper that solving a large and/or stiff Markov model requires a careful selection of the solution method/tool. Using an inappropriate method/tool for the solution of a non-trivial MC may lead to significant errors. In this paper we offer an algorithm to help with the selection of the most appropriate tool and technique which takes into account the model properties such as the model size and its stiffness.

Many approaches have been developed to deal efficiently with MC stiffness [4, 5, 6, 7]. They can be split into two groups - “stiffness-tolerance” (STA) and “stiffness-avoidance” approaches (SAA) [5]. With STA *specialised numerical methods* are used which provide highly accurate results despite stiffness. The limitations of STA are: i) STA cannot deal effectively with large models, and ii) computational efficiency is difficult to achieve when highly accurate solutions are sought. The SAA solution, on the other hand, is based on an *approximation algorithm* which converts a stiff MC to a non-stiff chain first, which typically has a significantly smaller state space [4]. An advantage of this approach is that it can deal effectively with large stiff MCs. Achieving high accuracy with SAA, however, may be problematic.

In this paper we present an empirical study using two systems: i) a computer system with hardware redundancy and diverse software under the assumptions that the rate of failure of software may vary over time, and ii) a queuing system with a server break-down and repair [4]. The solution of the first system is based on the principle of multi-fragmentation [9], one of the efficient methods of solving an MC in case the model parameters change over time. The main idea of this principle is that the MC is represented as a set of fragments with *identical structure*, but which differ in the values of one or more parameters. Each of the systems included in the study is described by a stiff MC. The main difference between the two systems is that the ratio between the stiffness indices (to be defined below) of the first and the second system is 10^3. In other words we chose them to be quite different in terms of their stiffness index so that we could study if the stiffness index impacts the accuracy of the different methods for solving the respective models. Both systems were solved using STA. The second system was also solved using SAA. Thus we could compare the accuracy of

[1] IEC 61508 – Part 6 states on p. 58: "Efficient algorithms have been developed and implemented in software packages a long time ago in order to solve above equations. Then, when using this approach, the analyst can focus only on the building of the models and not on the underlying mathematics..."

STA and SAA when applied to solving the same (the second) system and how these compare with the exact solution, which for the second system is available in [10], [11].

We report that indeed the stiffness index impacts significantly the accuracy of the solution methods. We also offer a novel selection procedure which allows one to choose (among the many available for solving stiff MCs) the solution method that provides the best accuracy given the value of the stiffness index of the MC to be solved and takes into account some additional attributes (usability, possibilities of verification). We provide also a justification for our recommendations based on the comparison of the different methods when applied to the chosen two systems described by stiff MCs. As a case study, we present availability assessment of an FPGA-based industrial safety controller manufactured by RPC Radiy [28], which consists of three parallel channels in a “2-out-of-3” configuration.

1.1 Related Works Analysis

The numerical transient analysis of MCs is faced with two computational difficulties – the model stiffness and largeness, which can affect the accuracy of the solutions obtained. Several numerical methods are widely used that address these difficulties, among them the Rosenbrock method [13], [14], the TR-BDF2 [5], [7], [14], the Jensen’s method (uniformization) [5], the implicit Runge-Kutta method [5], [6], [12], and the modified Gir method [6].

The Rosenbrock method is the one-step numerical method that has the advantage of being relatively simple to understand and implement. For moderate accuracy (tolerances of order 10^{-4} – 10^{-5}) and systems of moderate-size ($N \lesssim 10$) the method allows for obtaining solutions which in terms of achieved accuracy are comparable with the more complex algorithms. If a low accuracy is acceptable, then this method is attractive. When larger systems are solved the Rosenbrock method becomes less accurate and reliable [13].

TR-BDF2 is a second order accurate A-stable and L-stable single step composite method that uses one step of trapezoidal rule (TR) and one step of BDF2 (second order backward difference formula) [7]. [14] demonstrated that TR-BDF2 deals well with increased stiffness and only requires little extra computations as the parameter values or the mission time are increased. TR-BDF2 is also recommended for use if low accuracy is acceptable [5].

The Jensen method (also known as *uniformization* or randomization) [15] involves the computation of Poisson probabilities. It was extensively modified [5], [14], [16] to deal with the stiffness problem. It achieves greater accuracy than TR-BDF2 but still deals poorly with stiffness in extreme cases. [14] recommends that the Jensen method be used only in cases of moderately stiff models.

The implicit Runge-Kutta method is a single step numerical method that deals with the problem of stiffness and is one of the most computationally efficient methods for achieving high accuracy [6].

Also an aggregation/disaggregation technique for transient solution of stiff MCs was developed by K. S. Trivedi and A. Bobbio [4]. The technique can be applied to any MC, for which the transition rates can be grouped into *two separated sets of values*: one of the sets would include the "slow" states and the second set would include the "fast" states [4]. After aggregating the fast transition rates the MC is reduced to a smaller non stiff MC, which can be solved efficiently using a standard numerical technique [4].

In the rest of the paper the method by Trivedi and Bobbio [4] is referred to as an SAA while the other methods surveyed above are referred to as an STA.

The focus of this study is a comparison of the accuracy of the solution obtained with different methods when applied to the same system. Of particular interest is how the solutions are affected by the stiffness of the system under study.

The rest of the paper is organized as follow: in the section 2 we describe formally the stiffness problem and the stiffness index introducing informally the idea of how stiffness index may impact the accuracy of the numerical methods used in solving the systems. In section 3 we present the comparison results. In section 4 we present a procedure for selecting the optimal solution method and tool based on the stiffness index of an MC. In section 5 we present an industrial case study and use it to demonstrate the usefulness of the proposed procedure. In the section 6 we present the conclusions and outline the problems left for future research.

2 Comparative Analysis of Evaluation Techniques

2.1 The Stiffness Problem

Stiffness is an undesirable property of many practical MCs as it poses difficulties in finding transient solutions. There is no commonly adopted definition of "stiffness" but a few of the most widely used ones are summarized below.

The Cauchy problem $\frac{du}{dx} = F(x,u)$ is said to be stiff on the interval *[x0,X]* if for *x* from this interval the following condition is fulfilled:

$$s(x) = \frac{\max\limits_{i=1,n} |\operatorname{Re}(\lambda_i)|}{\min\limits_{i=1,n} |\operatorname{Re}(\lambda_i)|} >> 1, \tag{1}$$

where the *s(x)* – denotes the index of stiffness (stiffness index) and λ_i – are the eigenvalues of a Jacobian matrix($\operatorname{Re}\lambda_i < 0, i = 1,2,...,n$)[6].

Also the index of stiffness of an MC was defined in [5], [14] as the product of the *largest total exit rate* from a state and the *length of solution interval(=$\lambda_i t$),* where λ_i are the eigenvalues of the Jacobian matrix. A system of differential equations (DE) is said to be stiff on the interval *[0,t)* if there exists a solution component of the system that has variation on that interval that is large compared to *1/t.* Thus, the length of the solution interval also becomes a measure of stiffness [14], [17]. We use the index of stiffness in the empirical evaluations that follow as a measure of discrimination between the MCs with different indices of stiffness: high-stiffness ($s(x) \geq 10^4$), moderate-stiffness ($10^2 < s(x) < 10^4$) and low-stiffness ($s(x) \leq 10^2$).

Here we provide an illustration of how the index of stiffness can affect the accuracy achievable by numerical methods of solving a system of the DEs, which describes a stiff MC.

The most common type of a stiff linear system of DEs is the system in which the eigenvalues can be divided into two groups, based on the difference in their modulus values. The eigenvalues of the first group with large modulus values determine the solution behaviour in the boundary layer. Their corresponding components are rapidly decreasing. The eigenvalues of the second group with small modulus values determine the solution behaviour out of the boundary layer. The index of stiffness (1) is the ratio between the maximum value from the first group and the minimum value from the second one. To describe in detail the influence of this separation on the stability of the numerical methods let us consider a DE matrix with constant coefficients,

$$y' = Ay, A = (a_{ij}), i, j = 1,..., M \tag{2}$$

$$y(0) = y_0, y_i = (y_0^i), i = 1,..., M \tag{3}$$

Matrix A is a simple-structured matrix, which means that it has M linearly independent eigenvectors and λ_i, e_i – are the eigenvalues and the corresponding eigenvectors of matrix A, respectively. The solution of the Cauchy problem (2) with initial conditions (3) is presented in (4):

$$y(x) = \sum_{i=1}^{M} C_i e^{\lambda_i x} e_i \tag{4}$$

Each component $C_i e^{\lambda_i x} e_i$ present in the solution (4) is proportional to one of the eigenvectors and is integrated independently of the other components.

If matrix A has large absolute negative eigenvalues a very small step h would be required on the whole integration interval. With a large integration interval using a small integration step would cause an increase of the local round-off errors, which in turn would become a serious problem if high overall accuracy is sought.

As an example, let us consider a system of two differential equations. Without loss of generality let us assume that $|\lambda_1| >> |\lambda_2|$.

The exact solution (4) will take the following form:

$$y(x) = C_1 e^{\lambda_1 x} e_1 + C_2 e^{\lambda_2 x} e_2 \tag{5}$$

The first component of the solution will decrease rapidly on the interval $\tau = 1/|\lambda_1|$ and after that will become extremely small. In this interval this component influences the solution *y(x)*. The second component changes on the interval $\tau = 1/|\lambda_2|$. The second interval is much wider than the first one. In this interval the second component influences the solution *y(x)*. In the first interval the rate of solution change is high and it is dominated by the change of the first component. In the second interval, the rate of change is small and is dominated by the change of the second component. As we can see the values of the given coefficients affect the behaviour of the transient solution. On the first interval, the so called *boundary layer*, in order to achieve an accurate solution in the presence of rapid changes, the step-size must satisfy the condition $h << 1/|\lambda_1|$. In the second interval the same condition on the step is required, because the corresponding component of the DE must decrease.

The example despite its simplicity provides a clear illustration of how different eigenvalues may lead to the need for changing the step-size in different integration intervals so that accurate results may be obtained. The value of the stiffness index (1) clearly affects the accuracy achievable with a given solution method. The higher the stiffness index the stricter the requirements imposed on the stability of the chosen numerical method. We study in detail the impact of the stiffness index on the achievable accuracy with different numerical methods using two stiff MCs with substantially different stiffness indices, s(x). The first stiff MC is a model of a computer system with hardware redundancy and diverse software (S_{22}) [21]. The second system is a queuing system with server break-down and repair (M/M/1/k) [5].

Table 1. Experimental results

Method	**Parameter**	**t_1=[0,1000]**	
		S_{22}	**M/1/M/m**
Rosenbrock	NSS	93	172
	FE	2141	4302
	NLS	279	516
TR-BDF2	NSS	114	210
	FE	269	506
	NLS	361	692
Backward differentiation	NSS	74	150
	FE	111	203
	NLS	89	179

The first system is of moderate-stiffness, with $s(x)=4*10^2$(1), where max $|Re(\lambda_i)|$ = 0.2 and min$|Re(\lambda_i)|$ = 0.0005. The MC of the second system is of high-stiffness, with s(x) = $3.0001*10^4$, where max $|Re(\lambda_i)|$= 3.0001 and min$|Re(\lambda_i)|$=0.0001. Both MCs are of equal size – 20 states. The study was conducted using the functions for solving stiff DEs implemented in the mathematical package MATLAB – *ode23s, ode23tb, ode15s,* which implement the Rosenbrock method, the TR-BDF2 and the method of backward differentiation, respectively.

Table 1 summarizes the parameters obtained with the different methods for solving stiff DE: the number of successful steps (NSS), the function evaluations (FE) and the number of solutions of the linear systems (NLS). The time interval is t_1=[0;1000]. The Table shows that even when the model largeness is the same the solution for a system of high-stiffness, M/1/M/m, requires nearly twice as many steps, function evaluations and linear system solutions.

2.2 Stiffness-Tolerance and Stiffness-Avoidance Approaches

Stiffness-Tolerance Approach. The main idea of this approach is using methods that are stable for solving stiff models. These can be split broadly into two classes: "classical" numerical methods for solution of stiff DEs and "modified" numerical methods used for finding a solution in special cases.

(a) The classical (non-modified) numerical methods for solving stiff DEs use special single-step and multi-step integration methods. Examples of such methods are the implicit Runge-Kutta, the TR-BDF2, the Rosenbrock method, the exponential method, the implicit Gir method described in [5], [6], [7], [13], respectively. The implicit Runge-Kutta, TR-BDF2 and Rosenbrock method are implemented by several mathematical off-the-shelf software packages and are usually considered the most accurate methods for solving stiff ODEs.

(b) An example of the modified numerical methods is the exponential modified method. The original algorithm was presented in [6] and is based on the evaluation of the matrix exponent. In [6] this method is recommended as one of the most effective algorithms for solving the class of ODE systems with a high value of the Lipchitz constant, and as a special part of a stiff ODE. As a modification part, an automated adaptive step of integration can be implemented [6]. As the method has a multi-step algorithm the given modification can increase the accuracy of the solution. The amount of computations and the machine time needed for the solution of stiff DEs can be reduced, too, [6].

The solution provided by using any numerical method is expected to be accurate. However, typically the result obtained with numerical method include errors coming from different sources, such as: *problem statement error* - an inherent error, due to various simplifications introduced in order to make the problem (analytically) tractable; *truncation error* - the error related to truncating the infinite series after a finite number of terms are computed; *round-off error* - the type of error that arises in every arithmetic operation carried out on a computer; *initial error* - the error related to the presence of approximate parameters in the mathematical formulae. Ideally we would like to control each of these components of the computational error.

Stiffness-Avoidance Approach. The basic idea of this approach is a model transformation by identifying and eliminating the stiffness from the model, which would bring two benefits: i) a reduction of the largeness of the initial MC, and ii) efficiency in solving a non-stiff model using standard numerical methods. The approach was named an aggregation/disaggregation technique for transient solution of stiff MCs. The technique, developed by K. S. Trivedi, A. Bobbio and A. Reibmann [4], [11], can be applied to any MC with transition rates that can be grouped into two separate sets of values – the set of *slow* and the set of *fast* states [4].

While the transformation of the initial stiff MC brings benefits in terms of efficiency, to the best of our knowledge, no systematic study has been undertaken of the impact of the transformation (from a stiff to a non-stiff MC on the accuracy of the solution. In addition, since the method is not supported by standard off-the-shelf tools, the scope for human error in applying it is non-negligible.

3 Examples and Results

3.1 Example Systems Used in the Studies

In this subsection we provide a brief description of the systems that were solved using STA and SAA. The first system was solved using both approaches, while the second one – using only the SAA. The solution of the second system using STA was presented in [11, 12].

System 1: A Fault-Tolerant Computer System. The first system, Fig. 1, used in the study is a fault-tolerant computer system with two hardware channels, on which diverse control software is run. In this case increasing system reliability is achieved via redundant hardware-software components with identical hardware structure that supports "hot backup" [21]. We assume that software run of the hardware channels is diverse [18], i.e. non-identical but functionally equivalent software copies are deployed on the hardware channels. The architecture thus offers protection against software design faults. Two channel configurations are very widely used in many safety-critical application, e.g. in instrumentation of nuclear plants, for instance Quad 3000 SIS critical control and safety application. Also similar architectures are used in many business-critical applications, such as the fault-tolerant servers (Blade Server NS50000c, IBM z10, Sun SPARC Enterprise M9000 [21]).

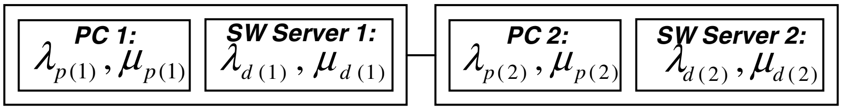

PC 1: $\lambda_{p(1)}, \mu_{p(1)}$
SW Server 1: $\lambda_{d(1)}, \mu_{d(1)}$
PC 2: $\lambda_{p(2)}, \mu_{p(2)}$
SW Server 2: $\lambda_{d(2)}, \mu_{d(2)}$

Fig. 1. Reliability block diagram of the chosen fault-tolerant system

Informally, the operation of the system is as follows. Initially the system is working correctly – both hardware and software channels deliver the service as expected. If during operation one of the hardware channels has failed, the processing will be failed over to the second channel until the first channel is "repaired". Similarly, a software component may fail, in which case a failover will take place to the other channel, etc. In addition, we assume that the rates of failure and repair of software will vary over time, e.g. as a result of executing the software in partitions as discussed in [20]. We implement this assumption based on the research work [21]. This assumption captures a plausible phenomenon – variation of software failure rates - which is well accepted in practice: various software 'aging effects' are indeed modelled by an increased rate of software failure.

Model Parameters. The model parameters are as follows: *i)* $\lambda_{p(1)}$, $\mu_{p(1)}$ and $\lambda_{p(2)}$, $\mu_{p(2)}$– hardware failure and repair rates of the first and second hardware channels, respectively; *ii)* $\lambda_{d(1)}$, $\lambda_{d(2)}$ – the initial software failure rate of the 1st and 2nd software versions; *iii)* $\Delta\lambda_d$ – the step of failure rate decrease after the software recovers from a failure; *iv)* $\mu_{d(1)}$ and $\mu_{d(2)}$ – the initial software repair rate of the 1st and the 2nd software versions; *v)*$\Delta\mu_d$ – the step of software repair rate decrease after the software recovers from a failure. We also assume that the values of system failure and repair rates of both the hardware components and of the software versions are equal: $\lambda_{p(1)} = \lambda_{p(2)}$, $\mu_{p(1)} = \mu_{p(2)}$, $\lambda_{d(1)} = \lambda_{d(2)}$, $\mu_{d(1)} = \mu_{d(2)}$ [21].

The Markov transition graph for the system presented in Fig. 1 is shown in Fig. 2. The model is built using the principle of multi-fragmentation [9]. Using this principle the model can be divided into *N* fragments that are with the same structure but may differ in one or more parameter values [21]. The number of fragments *N* in the MC depends on the number of expected undetected software faults N_d, the value of which can be estimated using probabilistic prediction models (6) [21]:

$$N=N_d+1 \tag{6}$$

The sum of the failure rates of both software versions is defined as (7):

$$\Lambda_d = \lambda_{d(1)} + \lambda_{d(2)} \tag{7}$$

The MC of System 1 consists of the following states: $SF_1=\{S_1, S_4, \ldots, S_{n+1}\}$ – the set of states when both the hardware and software on both channels are working correctly, $SF_2=\{S_2, S_5, \ldots, S_{3n+2}\}$ – the set of states in which one of the hardware channel has failed, $SF_3=\{S_3, S_6, \ldots, S_{3n+3}\}$ – the set of states in which one of the software versions has failed.

The system's operation is described next. At time t_0 the system operates correctly in S_1. At random moment, t_n, a hardware or a software component failure occurs. In case of a hardware failure the system moves to state S_2. The rate of this transition is $2\lambda_p$ and recovers from this state back to S_1 with rate μ_p. In case of software failure the system moves to state S_3 and recovers from this failure by moving to state S_4 with rate μ_d. The states S_1, S_2 and S_3 form the first fragment of the model, S_4, S_5 and S_6 – form fragment 2, etc. We assume that the internal fragments rates μ_d and Λ_d decrease by $\Delta\mu_d$ and $\Delta\lambda_d$, respectively, as the system moves between fragments from left to right.

From the (Fig. 2) we derive the matrix of the system transition rates (8), where *i=(1,..,n)* is the number of system fragments:

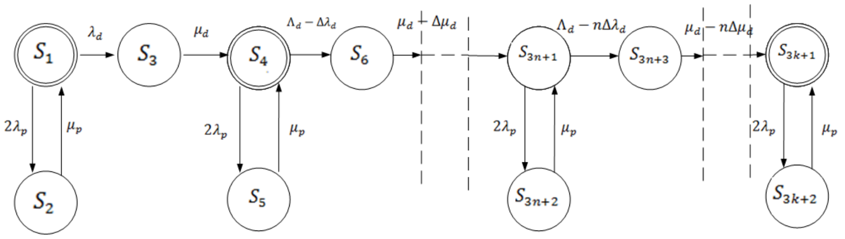


Fig. 2. Model of the system to be studied

$$\begin{pmatrix}
-(2\lambda_p+\lambda_d) & \mu_p & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 & 0 & 0 \\
2\lambda_p & -\mu_p & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 & 0 & 0 \\
\lambda_d & 0 & -\mu_d & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \mu_d & -(\Lambda_d-i\Delta\lambda_d+2\lambda_p) & \mu_p & 0 & \ldots & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 2\lambda_p & -\mu_p & 0 & \ldots & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \Lambda_d-i\Delta\lambda_d & 0 & -(\mu_d-i\Delta\mu_d) & \ldots & 0 & 0 & 0 & 0 & 0 & 0 \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
0 & 0 & 0 & 0 & 0 & 0 & \ldots & -(\mu_d-n\Delta\mu_d) & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \ldots & \mu_d-n\Delta\mu_d & -(\Lambda_d-n\Delta\lambda_d+2\lambda_p) & \mu_p & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \ldots & 0 & 2\lambda_p & -\mu_p & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \ldots & 0 & \Lambda_d-n\Delta\lambda_d & 0 & -(\mu_d-n\Delta\mu_d) & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & \mu_d-n\Delta\mu_d & -2\lambda_p & \mu_p \\
0 & 0 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 & 2\lambda_p & -\mu_p
\end{pmatrix} \quad (8)$$

System 2: A Queuing System with Server Breakdown and Repair. The second system was described in details by K.S. Trivedi and A. Bobbio in [4, 10]. The authors consider an M/M/1/k queuing system where m is the system capacity. The first “M” denotes a Poisson arrival process, the second “M” – denoted an exponentially distributed service times. The system has 1 server and k buffers to hold customers waiting for a service. The resulting model operates in 22 states [11].

Fig. 3 shows the Markov transition graph for the system. The (9) presents the matrix of system transition rates.

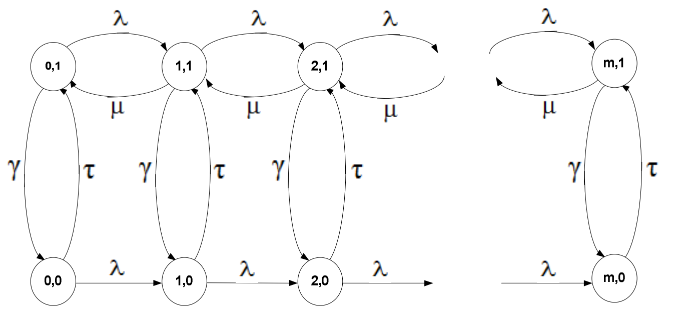


Fig. 3. The state diagram of the M/M/1/m queuing system with a server breakdown and repair

$$\begin{pmatrix}
-(\lambda+\gamma) & \tau & \mu & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\
\gamma & -(\lambda+\tau) & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\
\lambda & 0 & -(\lambda+\gamma+\mu) & \tau & \mu & 0 & \ldots & 0 & 0 & 0 & 0 \\
0 & \lambda & \gamma & -(\lambda+\tau) & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\
0 & 0 & \lambda & 0 & -(\lambda+\gamma+\mu) & \tau & \ldots & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \lambda & \gamma & -(\lambda+\tau) & \ldots & 0 & 0 & 0 & 0 \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
0 & 0 & 0 & 0 & 0 & 0 & \ldots & -(\lambda+\gamma+\mu) & \tau & \mu & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \ldots & \gamma & -(\lambda+\tau) & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \ldots & \lambda & 0 & -(\gamma+\mu) & \tau \\
0 & 0 & 0 & 0 & 0 & 0 & \ldots & 0 & \lambda & \gamma & -\tau
\end{pmatrix} \quad (9)$$

Model Parameters. The λ and μ are the arrival and the service rates, respectively, while γ and τ denote the server failure and failure repair rates. The main assumption is that the rates λ and μ are fast while γ and τ are slow [4]. Using the parameters presented in [11] we computed the index of stiffness for this system to be $s(x) = 3 \cdot 10^4$. Under the terms of this paper this system as classified as a *high-stiffness system*. The system was solved on the same interval [0, 1000) [11].

3.2 Experimental Results

In this Subsection we present the results obtained using the approaches described in Section 2. The solutions for the MC of high-stiffness (the queuing system) is referred to as *Experiment 1.* The solution for the MC of moderate-stiffness (the fault-tolerant computer system) with changing parameters, would be referred to as *Experiment 2.* Lastly, we define an MC of low-stiffness: this is a variant of the fault-tolerant computer system with two hardware channels in which the model parameters have been changed so that the stiffness index has become low ($s(x)=50$). This solution will be referred to as *Experiment 3*.

Experiment 1. A solution of the *queuing system with sever breakdown and repair* using SAA was presented in [4], [11]. In [4] as a measure of interest the authors used the expected number of customers in the queue, *E[N]*. In [11], in addition, the approximate result (calculated using SAA) and the exact values of $P_{22}(t)$ - the probability of having all buffers full and the server down – were plotted together.

Now we turn our attention to solving the MC of high-stiff using the STA to solve this model. We used two methods that fall into the STA category: i) using the EXPMETH utility; ii) using the functions for solving stiff DEs implemented in the mathematical package Mathematica.

We used EXPMETH with initial data as follows: the matrix of coefficients was set as defined in [11]; the mission time was set to $t=1000$; the accuracy of the solution sought was set to 10^{-6}; the number of steps was set to $n=20$. As Table 2 illustrates we obtained a negative result, where the probabilities were calculated for every 50 hours of the mission time, $S=\{S_1,S_2,S_3,\ldots,S_{20}\}$. The results from using the functions for solving stiff DEs implemented in Mathematica are summarised in Fig. 4 in which the exact solution for $P_{22}(t)$ for λ_1=1.0, given in [11], is compared with the results obtained with Mathematica.

Based on the empirical evidence with the STA methods applied to *Experiment 1* we conclude that STA cannot provide highly accurate solution for MCs of high-stiffness.

Table 2. Results obtained for Experiment 1 with ODU (EXPMETH)

S_n t	S_1	…	S_{20}
50	1.76231646111739250e+0100-4.15365304345735515e+0100	…	1.06048815978458797e+0095-3.22465594380700072e+0094
…	…	…	…
1000	1.88981041657022775e+2054-4.45414711917994213e+2054	…	1.13720867698699733e+2049-3.45794216161554014e+2048

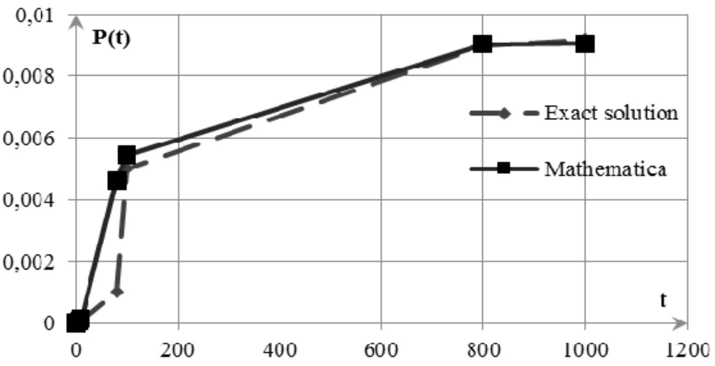


Fig. 4. Results of solving Experiment 1 model using Mathematica

Experiment 2. To solve a moderately-stiff MC (Experiment 2) using SAA we used the algorithm described in [4] and the uniformization method. STA solutions are also obtained using EXPMETH and the mathematical package Mathematica, respectively.

The mission time was set again to $t=1000$, $s(x)=4*10^2$ (1), where $max\ |Re(\lambda_i)|= 0.2$ is the value of μ_d, the software repair rate in the initial model fragment and $min|Re(\lambda_i)|=0.0005$ is the value of μ_d for the software repair rate in the final model fragment.

A comparison between the solutions is shown in Table 3 for values of the probabilities that the system is working in each of the states: $\{S_1, S_4, S_7, S_{10}, S_{13}, S_{16}, S_{19}\}$ at $t=500$. This set represents the states without failure (operational states, i.e. both channels work correctly without hardware or software failure).

Table 3. Results comparison. System 1

State/t=500	**SAA**	**STA**	
		EXPMETH	Mathematica
S_1(t)	0,536010	0,537050	0,537052
S_4(t)	0,323950	0,321630	0,321634
S_7(t)	0,078610	0,078540	0,078542
S_{10}(t)	0,010180	0,009810	0,009814
S_{13}(t)	0,000640	0,000620	0,000618
S_{16}(t)	0,000021	0,000020	0,000023
S_{19}(t)	0	0	0

Fig. 5 shows the results for system availability obtained with EXPMETH and Mathematica. For this system (Experiment 2) we used the result obtained with EXPMETH as an *exact solution* [18]. A discussion of the discrepancies between the solutions obtained with various packages is available in [18].

Based on this experiment we can conclude that for MCs of moderate - stiffness both the SAA and STA can be used. We note that the STA methods would produce an accurate solution *faster* than SAA when applied to small to moderate systems.

We also note that the particular mathematical package, Mathematica, detects automatically the stiffness of the DE's using a built in "StiffnessTest". In addition the user of this package can use "StiffnessSwitching", the basic idea as the name suggests being that the package will switch automatically between stiff and non-stiff solvers depending on the outcome of the stiffness test. The non-stiff solver uses the "ExplicitModifiedMidpoint" base method, while the stiff solver uses the "LinearyImplicitEuler" base method [22]. Such special methods can be useful in case of solving an MC of small to moderate size. Finally, we note that the mathematical package offers convenience, but at the same time significant effort is required to construct the necessary functions in case of large models, which introduces scope for human errors, e.g. while entering the required initial data.

EXPMETH can be more effective in terms of usability. With *Experiment 2* Mathematica required a function with 7 arguments, while EXPMETH only required 4 arguments and a matrix of coefficients of the DEs.

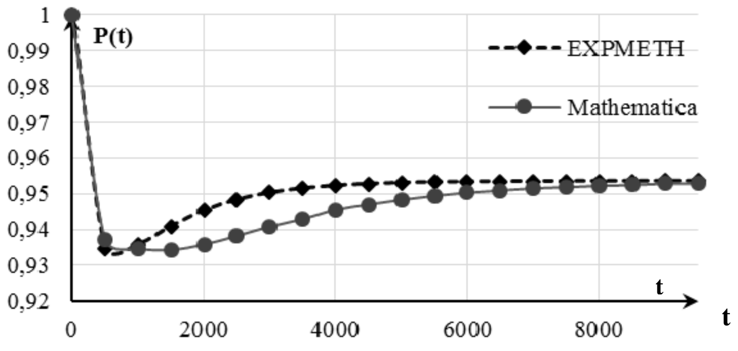


Fig. 5. Comparison of STA methods applied to Experiment 2

Experiment 3. We solved the model of a system with low-stiffness (*Experiment 3*) using the STA only. Based on the justification in Subsection 2.1 we concluded that SAA are the best for solving MCs of high-stiffness and large MCs of moderate-stiffness.

In case of MCs of low-stiffness the use of STA can take less time and still provides an accurate solution. As in the previous experiment we consider a mission time $t=1000$, the $s(x)=50$ (1), where $max\ |Re(\lambda_i)|= 0.2$ – the value of μ_d software repair rate in the initial model fragment and $min|Re(\lambda_i)|=0.004$ – the value of μ_d software repair rate in the final model fragment.

Fig. 6 shows the results of the comparison of system availability, A(t), obtained with EXPMETH and Mathematica. The results are practically indistinguishable. Based on this experiment we can conclude that for MCs of low-stiffness the STA can provide an accurate result.

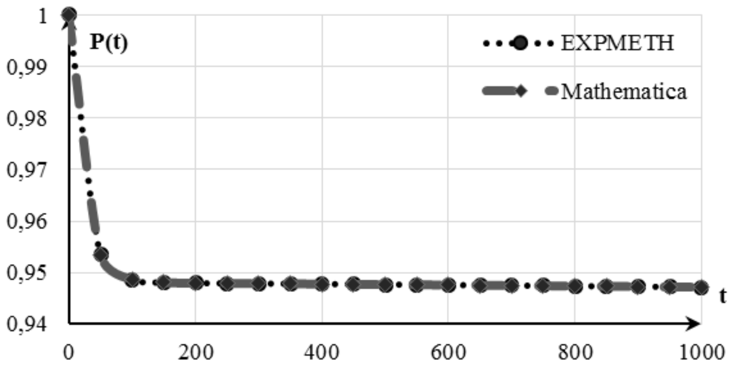


Fig. 6. Comparison of results with STA methods applied to Experiment 3

4 Selection of a Solution Method and of a Software Tool

Based on the results presented in the previous Subsection we propose the following novel selection procedure (Fig. 7), which takes into account the index of stiffness of the MC under consideration. As an example of ODU we use EXPMETH utility.

An MC of High-Stiffness. If in the system under consideration the parameters change (the top decision node) over time the principle of multi-fragmentation could be used and a multi-fragmentation model (MFM) should be build. Otherwise the assessor should build the MC. If the value of *s(x)* is greater than or equal to 10^4 the system model is defined as an MC (or MFM) of high-stiffness. We move to the left branch of the algorithm. Based on the results from *Experiment 1*, we propose that SAA be used. In this case using specialized software will provide the most accurate solution. The use of STA in this case can produce a solution of low accuracy, likely to be unacceptable.

The second layer of the algorithm takes the index of stiffness, *s(x)* given by (1), as an initial separator (the middle of the diagram). The system in question is assigned to one of the three classes: high-stiffness, moderate-stiffness or low-stiffness.

An MC of Moderate-Stiffness. As in the previous case, we propose that MFM can be built if in the system under consideration the parameters vary over time. If the parameters do not change, the procedure suggests building an MC. If the index of stiffness is in the interval [10^2, 10^4] we move to the branch in the middle of the diagram. On the third layer we propose that the largeness be used as an additional separator. As a theoretical separator the number of system states *n=1000* can be used. If the number of model states is greater than *1000* – the model is considered large, otherwise the model is not large. To provide effective results in case of a moderately-stiff large MC we propose the use of SAA and specialized tools. Indeed, one of the main features of SAA is the reduction of the system state space. For moderately-stiff MCs which are not large, based on the results of *Experiment 2*, we propose that STA be used with either ODU or other mathematical tools.

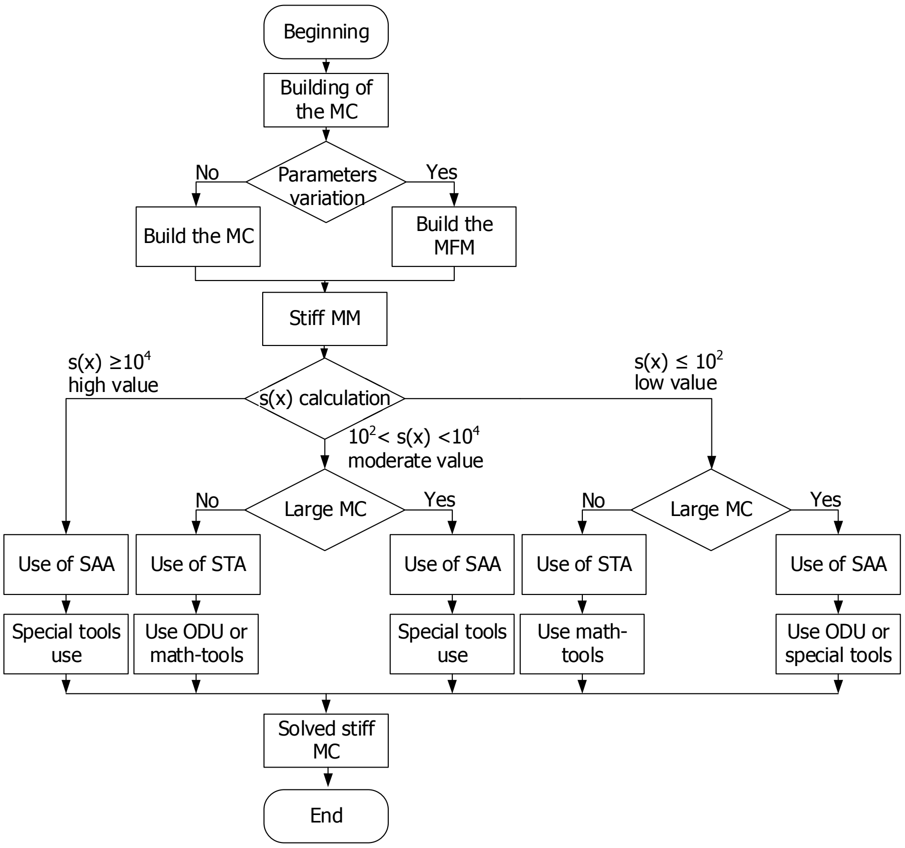


Fig. 7. An algorithm of selecting an optimal method for solving MCs based on the stiffness index and the size (largeness) of an MC

An MC of Low-Stiffness. If the index of stiffness is low, $s(x) \leq 10^2$, then the branch of the algorithm on the right is used, in which the system largeness is used as a decision node. In case of a large MC (of low-stiffness) we propose that STA be used; as a tool we would recommend either ODU (EXPMETH) or another specialized tool. These specialized tools would offer convenient data representation and satisfy the requirements of high accuracy. In the case of an MC of low-stiffness, which is "not large", we propose the use of STA. It should also be noted that in the process of selection from several mathematical packages and an ODU the usability becomes an important concern.

5 An Industrial Case Study

Here we illustrate the use of the proposed algorithm for assessment of a typical NPP Instrumentation and Control system (I&C) based on the digital platform by RPC Radiy [28]. The platform consists of 7 module types (Fig. 8): analog and digital signals input modules, a control module, a diagnostic module, analog and digital signal output modules and an actuators control module. All modules are based on Field Programmable Gates Arrays (FPGA) chips, nowadays widely used for safety-critical applications [1].

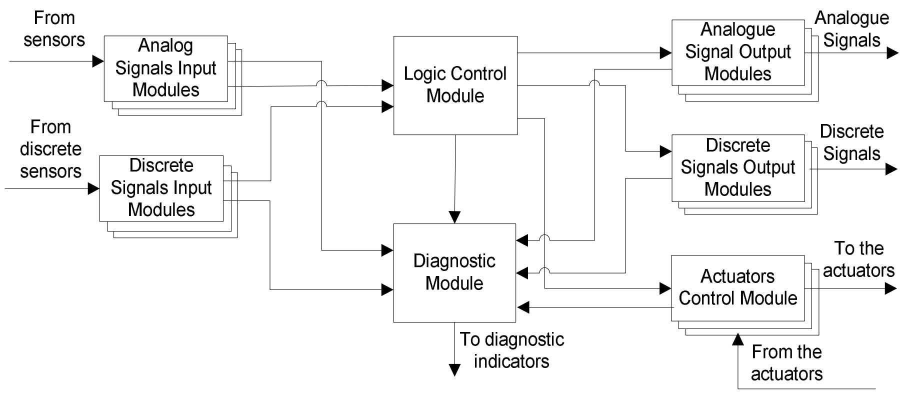


Fig. 8. Block diagram of one channel of I&C

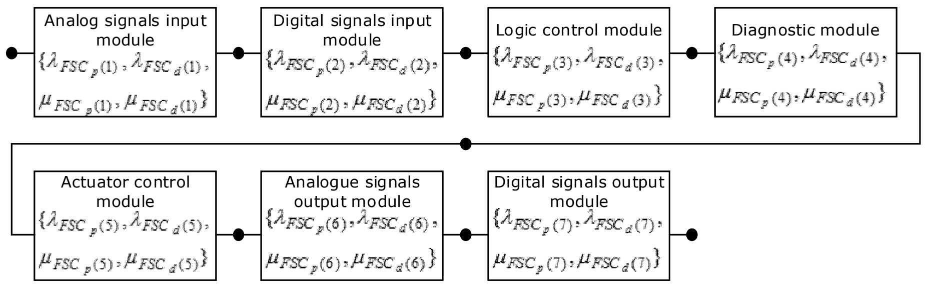

Analog signals input module
$\{\lambda_{FSC_p}(1), \lambda_{FSC_d}(1), \mu_{FSC_p}(1), \mu_{FSC_d}(1)\}$
Digital signals input module
$\{\lambda_{FSC_p}(2), \lambda_{FSC_d}(2), \mu_{FSC_p}(2), \mu_{FSC_d}(2)\}$
Logic control module
$\{\lambda_{FSC_p}(3), \lambda_{FSC_d}(3), \mu_{FSC_p}(3), \mu_{FSC_d}(3)\}$
Diagnostic module
$\{\lambda_{FSC_p}(4), \lambda_{FSC_d}(4), \mu_{FSC_p}(4), \mu_{FSC_d}(4)\}$
Actuator control module
$\{\lambda_{FSC_p}(5), \lambda_{FSC_d}(5), \mu_{FSC_p}(5), \mu_{FSC_d}(5)\}$
Analogue signals output module
$\{\lambda_{FSC_p}(6), \lambda_{FSC_d}(6), \mu_{FSC_p}(6), \mu_{FSC_d}(6)\}$
Digital signals output module
$\{\lambda_{FSC_p}(7), \lambda_{FSC_d}(7), \mu_{FSC_p}(7), \mu_{FSC_d}(7)\}$

Fig. 9. Reliability block diagram of one channel of I&C

The reliability block diagram of a *channel* is presented in (Fig. 9). We assume that a failure of any of the modules will lead to a failure of the channel, hence the reliability diagram shows the channel as a serial system of the modules.

An FPGA-based safety controller (FSC) is built using three parallel channels (Fig. 10) in a "2-out-of-3" configuration, i.e. majority voting is applied to the output signals.

It is also assumed that the corresponding components of the channels are identical, i.e. digital input signals module in the 1st channel is identical to the same module in the 2nd and the 3rd channel, etc. Similarly to the previous example with diverse fault-tolerant system, we assume variation of failure rate [21].

Informally, the system operates as follows. Initially, all components of the FSC are working correctly and deliver the service as expected. If one of the channels has failed, by majority voting the failure of the failed channel is detected and while the failed channel is being repaired, the controller will continue with the second and third channels only. If before the first channel has been repaired another channel fails, the majority voting component will detect the output disagreement and the FSC will stop. Clearly, the reliability of the majority component affects significantly the system reliability. Failure of this component leads to FSC failure.

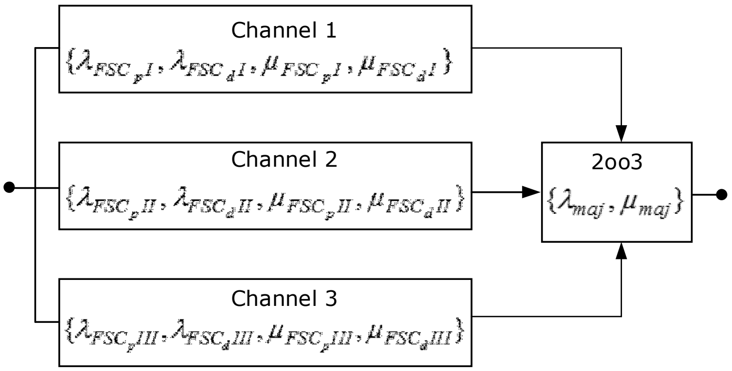

Channel 1
$\{\lambda_{FSC_pI}, \lambda_{FSC_dI}, \mu_{FSC_pI}, \mu_{FSC_dI}\}$
Channel 2
$\{\lambda_{FSC_pII}, \lambda_{FSC_dII}, \mu_{FSC_pII}, \mu_{FSC_dII}\}$
Channel 3
$\{\lambda_{FSC_pIII}, \lambda_{FSC_dIII}, \mu_{FSC_pIII}, \mu_{FSC_dIII}\}$
2oo3
$\{\lambda_{maj}, \mu_{maj}\}$

Fig. 10. Reliability block diagram of FSC with tree parallel channels on voting logic “2oo3”

5.1 Model Parameters

The model parameters are as follows:

(i) $\lambda_{FSC_p(i)}, \mu_{FSC_p(i)}, i=\overline{1,7}$ – the failure and repair rates for the failures caused by physical faults in the seven modules of each channel;

(ii) $\lambda_{FSC_d(i)}, \mu_{FSC_d(i)}, i=\overline{1,7}$,- the failure and repair rates for the failures caused by design faults in the seven modules of each channel;

We assume that the values of modules failure and repair rates for the failures caused by physical and design faults are equal for each module in each channel [21].

(iii) $\lambda_{FSC_pI}, \lambda_{FSC_pII}, \lambda_{FSC_pIII}, \mu_{FSC_pI}, \mu_{FSC_pII}, \mu_{FSC_pIII}$ – failure rates of the 1st, 2nd and 3rd channels (10) caused by physical faults, respectively:

$$\lambda_{FSC_pI} = \lambda_{FSC_pII} = \lambda_{FSC_pIII} = \sum_{i=1}^{7} \lambda_{FSC_p(i)} , \tag{10}$$

Based on the assumption that failure/repair rates of each modelare equal for each model, the failure repair rate of the entire channel can be determined as (11) [2]:

$$\mu_{FSC_pJ} = \lambda_{FSC_pJ} /(\sum_{i=1}^{7} \frac{\lambda_{FSC_p(i)}}{\mu_{FSC_p(i)}}) \Rightarrow \mu_{FSC_pJ} = \mu_{FSC_p(i)}, i=\overline{1,7},\ j=\overline{1,3} \tag{11}$$

(iv) $\lambda_{FSC_dI}, \lambda_{FSC_dII}, \lambda_{FSC_dIII}, \mu_{FSC_dI}, \mu_{FSC_dII}, \mu_{FSC_dIII}$ – the failure (12) and repair rates (13) of the 1st, 2nd and 3rd channels caused by design faults. $\Delta\lambda_{FSC_d}$ - is the step of failure rate decrease after the channel recovers from a failure; here we apply the same modelling assumption as with the example with software failure rates. Calculation of failure repair rate in this case is similar to (11):

$$\lambda_{FSC_dI} = \lambda_{FSC_dII} = \lambda_{FSC_dIII} = \sum_{i=1}^{7} \lambda_{FSC_d(i)} \tag{12}$$

$$\mu_{FSC_dJ} = \mu_{FSC_d(i)}, i=\overline{1,7},\ j=\overline{1,3} \tag{13}$$

(v) λ_{maj}, μ_{maj} - the failure and repair rates for the majority component.

As failure rates of 1st, 2nd and 3rd channels are equal we will further refer to them as follows: λ_{FSC_p}, μ_{FSC_p} - failure and repair rates of failures caused by physical faults and λ_{FSC_d}, μ_{FSC_d} - failure and repair rates of failures caused by design faults.

Since the failure rates change over time (as a result of channel repair) according to the selection algorithm an MFM is needed. The MFM describing the behaviour of the FSC is presented on Fig. 11. The number of fragments is calculated using (6).

The MFM consists of the following operating states: $SF_{3FSC}=\{S_1, S_8, S_{15}, S_{22}\}$-states in which all channels operate correctly; $SF_{2FSC}=\{S_2, S_9, S_{16}, S_{23}\}$-states in which two channels operate correctly. The system not operating states: $SF_{maj} = \{S_4, S_5, S_{11}, S_{12}, S_{18}, S_{19}\}$ – states in which the majority component fails; $SF_{1FSC}=\{S_3, S_{10}, S_{17}, S_{24}\}$ – states in which failure occurs in two channels; $SF_d=\{S_6, S_7, S_{11}, S_{12}, S_{20}, S_{21}\}$ – states in which design failure occurs.

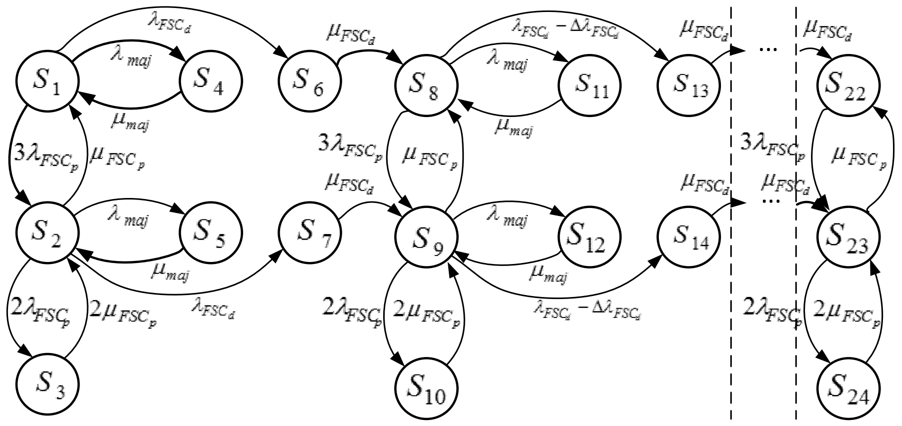


Fig. 11. Markov chain of the 2oo3 FSC

The system operation process is described next. At time t_0 all FSC components operates correctly in S_1. At random moment t_n, a failure caused by design or physical fault occurs and is detected by the majority voting component. In case of a failure caused by physical fault, the system moves to state S_2 with the rate $3\lambda_{FSC_p}$ and repairs to the state S_1 with rate μ_{FSC_p}. If during the first channel repair one of the remaining two operational channels also fails the system moves to S_3 with rate $2\lambda_{FSC_p}$ and repairs back to S_2 with rate $2\mu_{FSC_p}$. In case of failure caused by design faults, the system moves from state S_1 to state S_6 with rate λ_{FSC_d} and recovers from this failure by moving to S_8 with rate μ_{FSC_d}. If the failure caused by design faults occurs in system state S_2 the system moves to S_7 with rate λ_{FSC_d} and recovers from failure by moving to state S_9 with rate μ_{FSC_d}. If during the correct operation of all three channels (state *S₁*) or two channels (state *S₂*) a failure of the majority component occurs, the system moves to states *S₄* or *S₅*, respectively, with failure rate λ_{maj}. The states *{S₁, S₂ , S₃, S₄, S₅, S₆, S₇}* are form the first fragment of the model and *{S₈, S₉ , S₁₀, S₁₁, S₁₂, S₁₃, S₁₄}* – form fragment 2, etc.

From (Fig. 11) we derive the system of DE that describes the MFM. Here we provide DE for the initial (14), the first internal (15) and the final fragment (16).

DE for the initial fragment:

$$\begin{aligned}
&dP_1(t)/dt = -(\lambda_{FSC_d} + \lambda_{maj} + 3\lambda_{FSC_p})P_1(t) + \mu_{FSC_p}P_2(t) + \mu_{maj}P_4(t);\\
&dP_2(t)/dt = 3\lambda_{FSC_p}P_1(t) - (\mu_{FSC_p} + \lambda_{maj} + \lambda_{FSC_d} + 2\lambda_{FSC_p})P_2(t) +\\
&+ 2\mu_{FSC_p}P_3(t) + \mu_{maj}P_5(t);\\
&dP_3(t)/dt = 2\lambda_{FSC_p}P_2(t) - 2\mu_{FSC_p}P_3(t);\\
&dP_4(t)/dt = \lambda_{maj}P_1(t) - \mu_{maj}P_4(t);\\
&dP_5(t)/dt = \lambda_{maj}P_2(t) - \mu_{maj}P_5(t);\\
&dP_6(t)/dt = \lambda_{FSC_d}P_1(t) - \mu_{FSC_d}P_6(t);\\
&dP_7(t)/dt = \lambda_{FSC_d}P_2(t) - \mu_{FSC_d}P_7(t).
\end{aligned} \tag{14}$$

DE for the first internal fragment:

$$\begin{aligned}
&dP_8(t)/dt = \mu_{FSC_d}P_6(t) - ((\lambda_{FSC} - \Delta\lambda_{FSC_d}) + \lambda_{maj} + 3\lambda_{FSC_p})P_8(t) +\\
&+ \mu_{maj}P_{11}(t) + \mu_{FSC_p}P_9(t);\\
&dP_9(t)/dt = \mu_{FSC_d}P_7(t) + 3\lambda_{FSC_p}P_8(t) - (\mu_{FSC_p} + \lambda_{maj} + (\lambda_{FSC} - \Delta\lambda_{FSC_d}) +\\
&+ 2\lambda_{FSC_p})P_9(t) + 2\mu_{FSC_p}P_{10}(t) + \mu_{maj}P_{12}(t);\\
&dP_{10}(t)/dt = 2\lambda_{FSC_p}P_9(t) - 2\mu_{FSC_p}P_{10}(t);\\
&dP_{11}(t)/dt = \lambda_{maj}P_8(t) - \mu_{maj}P_{11}(t);\\
&dP_{12}(t)/dt = \lambda_{maj}P_9(t) - \mu_{maj}P_{12}(t);\\
&dP_{13}(t)/dt = (\lambda_{FSC} - \Delta\lambda_{FSC_d})P_8(t) - \mu_{FSC_d}P_{13}(t);\\
&dP_{14}(t)/dt = (\lambda_{FSC} - \Delta\lambda_{FSC_d})P_9(t) - \mu_{FSC_d}P_{14}(t).
\end{aligned} \tag{15}$$

DE for the final fragment:

$$\begin{aligned}
&dP_{22}(t)/dt = \mu_{FSC_d}P_{20}(t) - 3\lambda_{FSC_p}P_{22}(t) + \mu_{FSC_p}P_{23}(t);\\
&dP_{23}(t)/dt = 3\lambda_{FSC_p}P_{22}(t) - (\mu_{FSC_p} + 2\lambda_{FSC_p})P_{23}(t) +\\
&+ \mu_{FSC_d}P_{21}(t) + 2\mu_{FSC_p}(t)P_{24}(t);\\
&dP_{24}(t)/dt = 2\lambda_{FSC_p}P_{23}(t) - 2\mu_{FSC_p}P_{24}(t).
\end{aligned} \tag{16}$$

With the following initial conditions: $P_1(0) = 1,\ P_i(0) = 0, i \in 2,3,...,24$.

5.2 An Application of the Selection Algorithm

At the second stage of the proposed algorithm the index of system stiffness has to be determined (1).

We solve the MFM (Fig. 11) for the values of $\lambda_{FSC_p} = 1 \cdot 10^{-4}$, $\mu_{FSC_p} = 1$, $\lambda_{FSC_d} = 5 \cdot 10^{-5}$, $\mu_{FSC_d} = 1$, $\Delta\lambda_{FSC_d} = 1.6 \cdot 10^{-5}$, $\lambda_{maj} = 1 \cdot 10^{-5}$, $\mu_{maj} = 1$ and an accuracy requirement of 10^{-6}. The model thus become one of *moderate-stiffness*, with $s(x) = 6 \cdot 10^3$, where $max\ |Re(\lambda_i)| = 2$ and $min|Re(\lambda_i)| = 0.0003$, so we move to the branch in the middle of diagram. As we work in the state space that contain 24 states, the system MFM can be generally classified as not large MC. For moderately-stiff MCs, which are not large, the STA was proposed with either ODU (for example, EXPMETH) or other mathematical package.

We provided the assessment of MFM availability function $A(t)$using EXPMETH and Mathematica with “StiffnessTest” and “LinearyImplicitEuler” functions. The solution was computed on the time interval *[0; 10 000]* hours with the time-step *h=200* hours. The results are presented on Fig. 12. Table 4 presents the values of $A(t)$ with step 200 hours on the interval *[0; 5000]* hours and |ω| difference between them.

Also, we obtained solution of the initial MFM using SAA (aggregation/disaggregation technique [4]). Table 5, presents the probabilities of the initial, the first internal and the final fragments of the MFM states that were obtained using SAA in comparison to the results obtained with STA and the average difference |ω | between them.

The availability function $A(t)$ is defined as the sum of states probabilities $P_{FSC_i}(t)$ (17):

$$A(t) = \sum_{i=1}^{k} P_{FSC_i}(t) \tag{17}$$

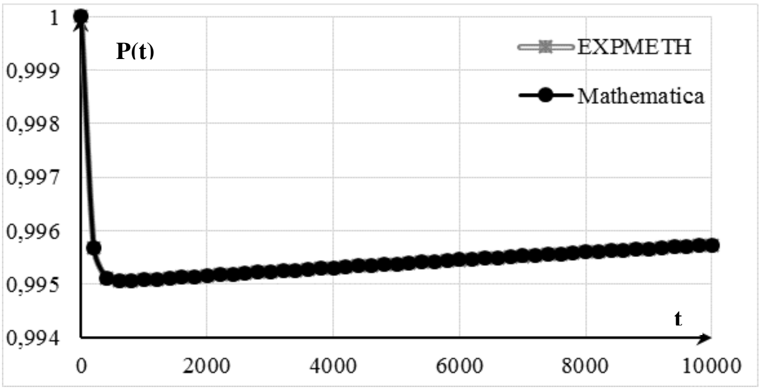


Fig. 12. Comparison of the results obtained with STA on the t = [0;10 000]

Table 4. Comparison of results obtained with STA on the t = [0; 5000]

t	A(t) EXPM.	A(t) Mathemat.	|ω|	t	A(t) EXPM.	A(t) Mathemat.	|ω|
0	1	1	0	2600	0.9952	0.995201	$1\cdot10^{-6}$
200	0.995686	0.995686	0	2800	0.995217	0.995216	$1\cdot10^{-6}$
400	0.99512	0.995121	$1\cdot10^{-6}$	3000	0.995232	0.995232	0
600	0.995058	0.995059	$1\cdot10^{-6}$	3200	0.995248	0.995247	$1\cdot10^{-6}$
800	0.995064	0.995064	0	3400	0.995262	0.995262	0
1000	0.995078	0.995078	0	3600	0.995277	0.995277	0
1200	0.995093	0.995093	0	3800	0.995291	0.995292	$1\cdot10^{-6}$
1400	0.995108	0.995109	$1\cdot10^{-6}$	4000	0.995307	0.995307	0
1600	0.995124	0.995124	0	4200	0.995322	0.995322	0
1800	0.995141	0.99514	$1\cdot10^{-6}$	4400	0.995337	0.995336	$1\cdot10^{-6}$
2000	0.995155	0.995155	0	4600	0.99535	0.995351	$1\cdot10^{-6}$
2200	0.995171	0.995171	0	4800	0.995366	0.995366	0
2400	0.995186	0.995186	0	5000	0.995381	0.995381	0

Table 5. Comparison of SAA and STA (EXPMETH) results

	S_1	S_2	S_8	S_9	S_{22}	S_{23}
SAA / t=200	0,989752	0,000297	0,009913	0	0	0
STA (EXPM.)	0,989743	0,000297	0,005639	0,000002	0	0
SAA / t=1000	0,950943	0,000285	0,047930	0,000005	0,000004	$1.5\cdot10^{-9}$
STA (EXPM.)	0,950935	0,000285	0,043298	0,000013	0,000002	0
SAA / t=2000	0,904565	0,000271	0,091919	0,000025	0,000038	$1.17\cdot10^{-8}$
STA (EXPM.)	0,904558	0,000271	0,087670	0,000026	0,000025	0
SAA / t=3000	0,860449	0,000258	0,132215	0,000037	0,000126	$3.81\cdot10^{-8}$
STA (EXPM.)	0,860443	0,000258	0,128325	0,000038	0,000095	0
SAA / t=4000	0,818485	0,000245	0,165049	0,000048	0,000291	$8.76\cdot10^{-8}$
STA (EXPM.)	0,818479	0,000246	0,165496	0,000049	0,000236	0
SAA / t=5000	0,778567	0,000233	0,202639	0,000058	0,000554	$1.665\cdot10^{-7}$
STA (EXPM.)	0,778561	0,000234	0,199401	0,000060	0,000472	0
SAA / t=6000	0,740595	0,000222	0,233193	0,000068	0,000931	$2.796\cdot10^{-7}$
STA (EXPM.)	0,740591	0,000222	0,230250	0,000069	0,000819	0
SAA / t=7000	0,704476	0,000211	0,260905	0,000076	0,001439	$4.32\cdot10^{-7}$
STA (EXPM.)	0,704472	0,000211	0,258238	0,000077	0,001297	0
SAA/ t= 8000	0,670118	0,000201	0,285958	0,000084	0,002091	$6.276\cdot10^{-7}$
STA (EXPM.)	0,670115	0,000201	0,283551	0,000085	0,001919	$1\cdot10^{-6}$
SAA/ t= 9000	0,637436	0,000191	0,308527	0,000092	0,002896	$8.691\cdot10^{-7}$
STA (EXPM.)	0,637433	0,000191	0,306361	0,000092	0,002699	$1\cdot10^{-6}$
SAA/ t=10000	0,606349	0,000182	0,328775	0,000098	0,003865	$1.1601\cdot10^{-6}$
STA (EXPM.)	0,606346	0,000182	0,326834	0,000098	0,003648	$1\cdot10^{-6}$
Average |ω|	$5.36\cdot10^{-6}$	$1.82\cdot10^{-7}$	$2.9\cdot10^{-3}$	$1.1\cdot10^{-6}$	$9.3\cdot10^{-5}$	$3.33\cdot10^{-7}$

Based on the assessment results we can conclude that both ODU and the mathematical packages provide almost equally accurate solutions for a moderately-stiff, not large MC. It should be noted that with the particular example, the use of ODU can be seen as more convenient and effective since ODU does not require any complex data and model preparation. Mathematica required a system of DE in a symbolic form, a matrix of coefficients, and a function with 7 arguments, while EXPMETH only required 4 arguments and a matrix of coefficients of the DEs.Comparing the quantitative values of the probabilities obtained using STA and SAA (Table 5) we can conclude that SAA is also effective in case of moderately-stiff not large system solution, but requires additional effort and time to prepare the model and data for computation.

6 Conclusion

As a result of conducted empirical studies we established that the value of stiffness index and the size of an MCs can affect the accuracy of the solutions achievable using different methods. One of the interesting results that we report is that we can effectively use the SAA to solve a large moderately-stiff MC when the parameters vary, which was the focus in our previous work [18].

In our future work we intend to extend the algorithm presented in this paper and seek to identify the most effective approach for dealing with large MCs: largeness-tolerant and largeness-avoidance approaches. We are hoping that eventually we will be able to make recommendation about the most suitable method dealing with both "largeness-stiffness" of systems with variable parameters.

References

1. Review Guidelines for Field-Programmable Gate Arrays in Nuclear Power Plant Safety Systems. NUREG/CR-7006, U.S. Nuclear Regulatory Commission, Washington, D.C., USA (2010)
2. Ventsel', E., Ovcharov, L.: Probability Theory and its Applications in Engineering. Nauka, Moscow (2000) (in Russian)
3. Archana, S., Srinivasan, R., Trivedi, K.S.: Availability Models in Practice. In: Proc. Int. Workshop on Fault-Tolerant Control and Computing (FTCC-1), Seoul, Korea, May 22-23 (2000)
4. Bobbio, A., Trivedi, K.S.: A Aggregation Technique for Transient Analysis of Stiff Markov Chains. IEEE Transactions on Computers C-35, 803–814 (1986)
5. Malhotra, M., Muppala, J.K., Trivedi, K.S.: Stiffness-Tolerant Methods for Transient Analysis of Stiff Markov Chains. Microelectronic Reliability 34(11), 1825–1841 (1994)
6. Arushanyan, O., Zaletkin, S.: Numerical Solution of Ordinary Differential Equations using FORTRAN. Moscow State University, Moscow (1990) (in Russian)
7. Bank, R.E., et al.: Transient Simulation of Silicon Devices and Circuits. IEEE Transactions on Electron Devices 32(10), 1992–2007 (1985)
8. Geist, R., Trivedi, K.S.: Reliability Estimation of Fault-Tolerant Systems: Tools and Techniques. Computer 23, 52–61 (1990)

9. Kharchenko, V., Timonkin, G., Sychev, V.: Fundamentals of Design and Constructions the Automated Systems for Aircraft Complexes Technical State Control. Study Guide. KhHMCIC, Kharkov (1992) (in Russian)
10. Nicola, V.F.: Markovian Models of Transactional System Supported by Check Pointing and Recovery Strategies, Part 1: a Model with State-Dependent Parameters. Eindhoven Univ. Technol., Eindhoven, The Netherlands, EUT Rep. 82-E-128 (1982)
11. Reibman, A., Trivedi, K.S., Kumar, S., Ciardo, G.: Analysis of Stiff Markov Chains. ORSA Journal on Computing 1(2), 126–133 (1989)
12. Hayrer, E., Vanner, G.: Solution of Ordinary Differential Equations. Stiff and Differential-Algebraic Poblems. Mir, Moscow (1999) (in Russian)
13. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes. The Art of Scientific Computing, 3rd edn. Cambridge University Press (2007)
14. Reibman, A., Trivedi, K.S.: Numerical Transient Analysis of Markov models. Comput. Opns. Res. 15(1), 19–36 (1988)
15. Jensen, A.: Markoff Chains as an Aid in the Study of Markoff Processes. Skand. Aktuarietidskrift 36, 87–91 (1953)
16. Fox, B.L., Glynn, P.W.: Computing Poisson Probabilities. Commun. ACM 31(4), 440–445 (1985)
17. Miranker, L.: Numerical Methods for Stiff Equations and Singular Perturbation Problems, Dordrecht, Holland (1981)
18. Kharchenko, V., Popov, P., Odarushchenko, O., Zhadan, V.: Empirical Evaluation of Accuracy of Mathematical Software Used for Availability Assessment of Fault-Tolerant Computer Systems. RT&A #03(26), 7, 85–97 (2012)
19. Littlewood, B., Popov, P., Strigini, L.: Modelling Software Design Diversity - a Review. ACM Computing Surveys 33(1), 177–208 (2001)
20. Popov, P., Manno, G.: The Effect of Correlated Failure Rates on Reliability of Continuous Time 1-Out-of-2 Software. In: Flammini, F., Bologna, S., Vittorini, V. (eds.) SAFECOMP 2011. LNCS, vol. 6894, pp. 1–14. Springer, Heidelberg (2011)
21. Kharchenko, V., Odarushchenko, O., Ponochovny, Y., Odarushchenko, E., Kharibin, O., Odarushchenko, V.: High Availability Systems and Technologies. In: Kharchenko, V. (ed.) Lectures, National Aerospace University "KhAI" (2012)
22. Wolfram Mathematica 9 Documentation Center, `http://reference.wolfram.com/mathematica/tutorial/NDSolveStiffnessSwitching.html`
23. Trivedi, K.S., Ciardo, G., Dasarathy, B., Grottke, M., Matias, R., Rindos, A., Varshaw, B.: Achieving and Assuring High Availability. In: IEEE International Symposium, IPDPS 2008, pp. 1–7 (2008)
24. Trivedi, K.S., Sahner, R.: SHARPE at the Age of Twenty Two. ACM SIGMETRICS Performance Evaluation Review 36(4), 52–57 (2009)
25. Reliability Software, Training, Consulting and Related Reliability Engineering Analysis Services from ReliaSoft Corporation, `http://www.reliasoft.com/index.html`
26. Clark, G., Courtney, T., Daly, D., Deavours, D., Derisavi, S., Doyle, M.J., Sanders, W.H., Webster, P.: The Möbius Modeling Tool. In: Proc. 9 Int. Workshop on Petri Nets and Performance Models, Aachen, Germany, pp. 241–250 (2001)
27. IEC 61508 (6 part), Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems (2010)
28. Kharchenko, V., Sklyar, V., Volkoviy, A.: Development and Verification of Dependable Multi-Version Systems on the Basic of IP-Cores. In: Proc. Int. Conf. Dependability of Computer Systems (2008)

Quantifying Ontology Fitness in OntoElect Using Saturation- and Vote-Based Metrics

Olga Tatarintseva[1,2], Vadim Ermolayev[1], Brita Keller[3], and Wolf-Ekkehard Matzke[3]

[1] Department of IT, Zaporozhye National University,
66 Zhukovskogo st., 69063, Zaporozhye, Ukraine
tatarintseva@znu.edu.ua, vadim@ermolayev.com
[2] Satelliz, 158 Lenina st., P.O. Box 317, 69057, Zaporozhye, Ukraine
[3] MINRES Technologies GmbH, Keltenhof 2, 85579, Neubiberg, Germany
{brita,wolf}@minres.com

Abstract. This paper presents the details of the OntoElect methodology for ontology engineering. These details comprise: (i) the presentation of the objectives with the emphasis on the problems arising when the domain knowledge stakeholder requirements to the developed ontology are elicited; (ii) the elaboration of the ontology engineering workflow and software tools; (iii) the proposal of the formal metrics for the representativeness of the used document corpus based on saturation and the fitness of different ontology elements to those requirements based on the computation of the stakeholder votes. The paper also reports on the set-up and results of our experiment with the document corpus of the ICTERI conference series papers and the ICTERI scope ontology. The available results of this ongoing experiment confirm that the methodological approach of OntoElect is valid.

Keywords: OntoElect, ontology engineering methodology, term extraction, term conceptualization, saturation, vote, ontology fitness.

1 Introduction

Developing an ontology, in its lifecycle, that fits well to the requirements of the domain knowledge stakeholders is a complicated task which does not have a satisfactory solution so far. The problem is to a large extent in devising a rigorous methodology for ontology engineering that enables a complete and timely account for those requirements and maps them to the updated revision of the ontology thus improving its fitness. One complication is that it is very hard to get the stakeholders possessing the knowledge and expertise in the domain who are committed to provide their inputs in an explicit form. Those inputs are however required as they need to be processed, measured, and applied in a harmonized way to ensure the consistency and validity of the result. We propose a way to remedy this complication by extracting the required inputs from the available products of the domain knowledge stakeholders – their professional papers.

This paper presents our OntoElect methodology for iterative and incremental ontology engineering. This work is based on, extends, and refines our previous results [1, 2, 3]. The approach has been proposed in [1] using the allusion of elections in which different "ontology offerings" compete for the commitment of the pool of the relevant domain knowledge stakeholders being the "electorate". This approach has been basically validated in an experiment reported in [2] where the measurable objectives were initially specified and used in the experimental study using a social semantic annotation approach. The approach has been further extended in [3] which reported about using term extraction from the relevant document corpus for learning the votes of the domain knowledge stakeholders. In this paper we (i) put OntoElect in line with the other well known ontology engineering methodologies and describe its workflow; (ii) offer a formalism to measure all the necessary aspects in this workflow, based on the metaphors of document corpus saturation and domain knowledge stakeholder votes; and (iii) evaluate the methodology by applying it to the corpus of ICTERI conference series papers and ICTERI scope ontology.

OntoElect is positioned as a complementary add-on to the existing ontology engineering methodologies focused on the proper: (i) elicitation of the requirements by the domain knowledge stakeholders inferred indirectly and without involving the subjects in the elicitation process; (ii) account for those requirements in the subsequent conceptualization of the requirements through the use of mappings of several types and the computation of the domain stakeholder votes based on those mappings and confidence factors; (iii) way to analyze the fitness of the ontology and provide recommendations about the most valuable improvements to it.

The remainder of the paper is structured as follows. Section 2 briefly reviews the related work in relevant fields. Section 3 presents the OntoElect methodology. Section 4 describes the environment for our evaluation experiment and sets the experiment up. Section 5 presents and discusses the results of our experiment. The paper is further concluded, and our plans for the future work are outlined.

2 Related Work

One of the possible ways to check if a conceptualization of a domain is correct and complete, is to evaluate the model against the interpretation of the meaning of the representative set of relevant documents. The document corpus will be relevant and representative if it covers the majority of the views by the domain knowledge stakeholders. Their interpretations may be collected and further analysed for refining the ontology using different techniques from several areas of research and development. In this section we briefly outline the related work in the relevant fields and refer to our previous publications [1, 4] for a more in-depth and detailed coverage.

One of the popular relevant research areas studying how interpretations are collected is collaborative or social tagging and annotation. A good survey of the field is [5] where the use of tags for different purposes and associated shortcomings are analysed. Semantic annotation and tagging approaches further refine social tagging techniques by offering the collections of terms that are taken from taxonomies, folksonomies, or thesauri

[6]. Hybrid approaches for collaborative tagging and annotation aiming at the enrichment of seed knowledge representations by a user community are reported for example in [7]. One of the knowledge representation systems for those lightweight ontologies, gaining considerable attention in practices, is SKOS [8]. The aspects of collaborative work, using Wiki, are covered by a Web 2.0 based approach for ontology maturing [9], which is particularly relevant in its spirit to OntoElect.

One of the promising approaches focused, besides collecting interpretations, on motivating more people to take part in developing or refining ontologies is offering a game with a purpose to intended users. In this approach, ontology development can be implicitly embedded in a game software, where ontology elements are created, updated, and validated in the background [10]. The gaming approach has also been tried for evaluating how well ontological specifications fit to the interpretations of random users (FACTory Game, http://game.cyc.com/). Several game scenarios have been elaborated [7] covering different aspects of ontology development and content annotation. Those are similar to OntoElect in the sense that they offer the possibilities to identify if users start to agree on or commit to certain ontological items.

Social and gaming approaches that involve the direct participation of human stakeholders are complemented by the plethora of research results in automated knowledge extraction or ontology learning. This strand of research involves the stakeholders indirectly – through making use of their professional outputs, like authored texts. A comprehensive survey of the techniques used to learn ontologies from texts is [11]. One collection of research contributions in ontology learning and population, complementary to this review, is [12]. In the experiment presented in this paper only term extraction using the TerMine tool [13] has been performed.

Yet one more important aspect in developing or refining ontologies is the re-use of the other ontologies or their parts most relevant to the developed ontology. In this context the ontology meaning summarization approach [14] may help an ontology engineer choose the parts for re-use. The approach is based on detecting the "key concepts" of an ontology under analysis which best characterize its meaning. The key concepts are determined using a combination of criteria from lexical statistics, taxonomy graph analysis, and popularity based on a number of hits. Especially in using the popularity and coverage metrics, this approach coincides with OntoElect. OntoElect is however used not for summarizing but refining an ontology based, among other things, on assessing the coverage and popularity of the ontology elements. In addition, the metrics and the computational mechanisms are different. On the other hand OntoElect does not yet consider ontology re-use as an important mechanism for refinement. Hence, combining some features of [14] in OntoElect may enrich it.

OntoElect uses the notion of ontology fitness, which is central for evaluating the quality of an ontology, in a meaning similar to that of [15], comprising: (i) fitness to the goals of the domain knowledge stakeholders; and (ii) fitness to an existing knowledge repository. OntoElect extends this notion by providing the formal mechanism to measure fitness, based on the metaphor of the stakeholder votes.

Finally, OntoElect is an ontology engineering methodology and therefore has to be compared to the existing mainstream methodologies in the field. This comparison is given in Section 3.

3 OntoElect Methodology

OntoElect is a focused working pattern for some activities in ontology engineering (see Fig. 1). It is complementary to the existing ontology engineering methodologies[1] – for example [16, 17, 18, 19], that embrace all the phases and activities in the lifecycle [20].

The contribution of OntoElect is in maximizing the fitness of the developed ontology to what the domain knowledge stakeholders think about the domain. Fitness is measured as the stakeholders' votes – a metric that allows assessing the stakeholders' commitment to the ontology under development. The more votes are collected – the higher the commitment is expected to be. The votes are acquired both directly – through semantic tagging, or indirectly – by extracting the tacit knowledge or interpretation of the domain from the artifacts authored by the stakeholders.

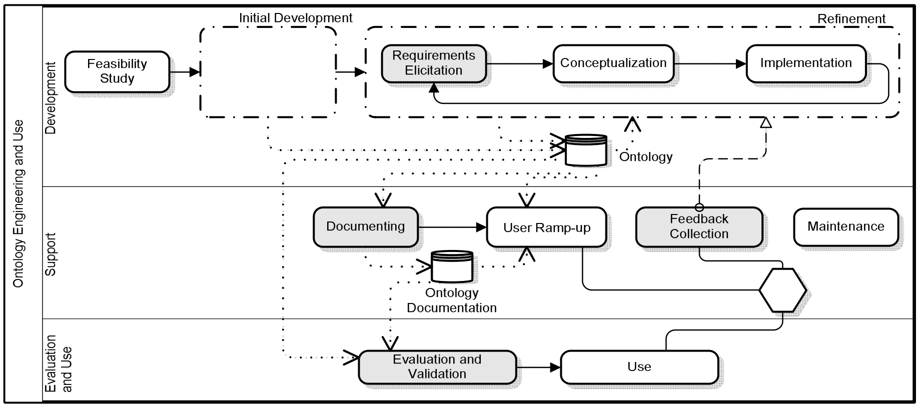


Fig. 1. The focus of OntoElect in ontology engineering lifecycle. Lifecycle phases and activities are summarized from [16, 17, 18, 19]. The Initial Development phase contains the same activities as the Refinement Phase. OntoElect focuses on the grey-shaded activities.

Our results in the direct acquisition of votes and the outcomes of the corresponding experiments have been reported in [4, 3]. In this paper we focus on how the votes could be extracted indirectly – from the professional texts written by the subject experts in a domain.

In this section we first describe the OntoElect approach at the level of objectives and with the emphasis on the problems arising when the knowledge stakeholders' requirements to the domain ontology are elicited. We then elaborate the ontology engineering workflow of OntoElect and describe the used software tools. Finally, we present the metrics that we found reasonable to exploit in the process and also for the analysis of the resulting knowledge representation artifact.

[1] In ontology engineering a methodology is generally understood as a suggested working pattern – i.e. "a body of methods, rules, and postulates employed by a discipline**:** a particular procedure or set of procedures" (Merriam-Webster Dictionary, `http://www.merriam-webster.com/dictionary/methodology`).

3.1 OntoElect and Domain Knowledge Elicitation

Any ontology development process takes as an input the requirements by the subject experts in the domain of interest and produces as its output the ontology – a formal descriptive theory covering those requirements correctly and to the maximal possible extent. Straightforwardly, the set of methods shaping out this process need to comprise the mechanisms for:

- Eliciting the (change[2]) requirements from the domain knowledge stakeholders as fully as possible
- Measuring how completely the requirements were captured
- Transforming the elicited requirements to the (changes in the) ontology
- Measuring how well the result fits to the intentions of the domain knowledge stakeholders

If a methodology fails to do any of the above sufficiently well then the commitment of the knowledge stakeholders to the output ontology will be low. Devising those mechanisms for a methodology of ontology engineering is however a complicated task (see also Fig. 2).

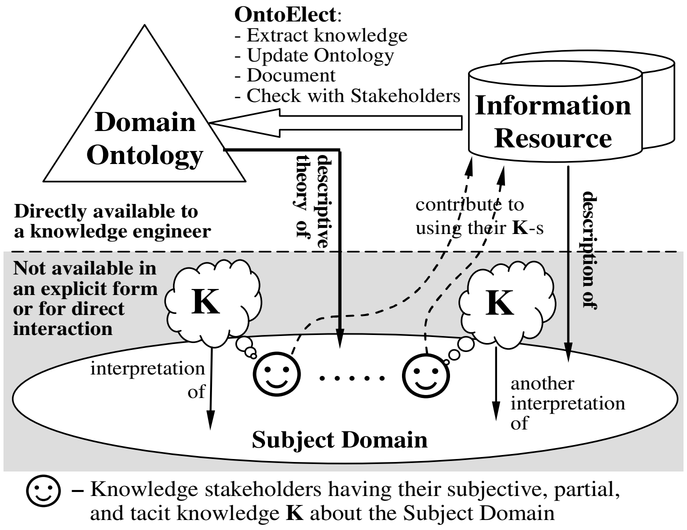

OntoElect:
- Extract knowledge
- Update Ontology
- Document
- Check with Stakeholders
Domain Ontology
Information Resource
descriptive theory of
contribute to using their **K**-s
description of
Directly available to a knowledge engineer
Not available in an explicit form or for direct interaction
K
K
interpretation of
another interpretation of
Subject Domain
– Knowledge stakeholders having their subjective, partial, and tacit knowledge **K** about the Subject Domain

Fig. 2. The challenges of domain knowledge elicitation

A domain ontology $\mathbf{O_D}$ could be regarded as a harmonized formal, and explicit representation of the union of the interpretations (**K**) by the knowledge stakeholders $s_i \in \mathbf{S}$ of the subject domain **D**. So, naively, we may elicit all the **K**-s and build the ontology of those as:

[2] Change requirements are elicited in the ontology Refinement phase. In the phase of Initial Development (Fig. 1) initial requirements are collected.

$$\mathbf{O_D} = hrm(\bigcup_{\mathbf{S}} unf_{f_j}(\mathbf{K}_{s_i})) , \tag{1}$$

where hrm is a harmonization function and unf is the transformation that maps a knowledge interpretation represented in the form f_j to the knowledge representation formalism used by the knowledge engineer (unification). Even if so, harmonization and unification functions are not easy to perform. For example, a formalism f_j for K_{s_i} could be more expressive than the ontology specification language used for coding $\mathbf{O_D}$; K_{s_m} and K_{s_n} could be mutually contradictory in some parts; etc.

Reality introduces more complications (also illustrated in Fig. 2):

- **K**-s are **subjective.** The stakeholders interpret their domain based on their individual background knowledge and experience. Hence, such interpretations, though appropriate in a particular context of the subject expert's activity and experience, may be objectively false beyond this narrow context.
- **K**-s are **tacit.** The views on the domain by the subject experts are often not fully stated explicitly, quite apart from the fact that they are materialized using different f_j. On the contrary, some parts of those **K**-s are assumed, taken as evident or default, having in mind that all the professional community knows/regards those assumptions in a similar way. Those tacit parts are the cause for difference in interpretations, or even misinterpretations, in particular by knowledge engineers.
- **K**-s are **partial.** In addition to being subjective in their interpretations, the knowledge stakeholders often do not possess a comprehensive view of the domain. Subject experts focus on their narrow context of professional interest and expertise, and have only a shallow coverage of the broader area within the domain. The partiality and fragmentation of their **K**-s is the reason for (a) contradictions between different views on the overlapping contexts; and (b) gaps in the coverage of the domain.
- **K**-s are **not available.** Last but not least, the knowledge stakeholders are not readily willing to spend their time for materializing their **K**-s or revealing them to knowledge engineers in another form. The subject experts often consider as inefficient the ratio of their own resource to be spent versus the utility of the resulting ontology as an artifact usable in their professional activity. One more de-motivating reason for the subject experts is that they fear becoming less competitive and valuable to the company or community after making their knowledge available to the public. The non-availability of **K**-s seriously diminishes the effectiveness of any knowledge elicitation activity.

One possible way to build a good ontology in these circumstances is to use the materialized product of the knowledge stakeholders which describes their domain. Such products, in contrast to **K**-s, are normally available in the form of information resources, for example professional texts authored by the subject experts in the domain. As the stakeholders use their **K**-s in their professional activity, and therefore contribute to the relevant information resources, those texts could be regarded as the footprints of their **K**-s. These footprints could be extracted from the available resources and without any direct involvement of the stakeholders. OntoElect methodology for ontology engineering follows this approach and also offers the abovementioned mechanisms for eliciting requirements and transforming them into the ontology.

For resolving the abovementioned complications and providing the necessary mechanisms in the presence of these complications OntoElect exploits the allusion of public elections [1]. Following this allusion, an ontology is regarded as a “parliament” comprising those members that win the election campaign. The candidates to the “parliament” are the structural contexts of the ontology elements [21], i.e. the concepts together with all their properties and related individuals. Those contexts either win or lose in the election process depending on the ratings computed from the votes expressed by the electorate. The members of the electorate are the knowledge stakeholders in the respective domain. Their votes could be expressed directly via the semantic tagging of the domain descriptions [3]. Getting those semantic tags is however a very challenging task which often fails to be accomplished with acceptable quality. Therefore, a reasonable alternative pathway is to extract the votes from the footprints of the stakeholders’ knowledge about the domain. Votes could be positive, partial positive, propagated, and negative, all weighted by confidence factors expressed by a knowledge engineer.

3.2 OntoElect Workflow and Tools

The flow of activities in OntoElect is pictured in Fig. 3. The workflow involves the three roles: knowledge engineers, subject experts, and all the knowledge stakeholders comprising subject experts. The major workload and coordination function falls on the knowledge engineers. The subject expert role involves those domain knowledge stakeholders who contributed their professional texts to the domain description resource.

The initial phase of a knowledge engineer workflow is requirements elicitation. In OntoElect it is done by (**a**) identifying a domain description resource and adding it to the document corpus; (**b**) extracting the terms from the document corpus together with the scores based on the frequencies of the appearance of the particular terms in the document corpus.

The identification of relevant domain description resources is done manually and preferably with the involvement of the subject experts in the domain. A very important question about the identified collection of resources concerns their representativeness – i.e. if they are indeed based on the qualifying majority sample of the **K**-s by the domain knowledge stakeholders. A tricky part of the work to be done is to answer this question without having the stakeholders directly involved in voting. OntoElect uses a saturation-based metric for estimating representativeness (Section 3.3).

For term extraction the TerMine service[3] was used as a tool in our experiments, which identifies key phrases in a plain text. The service uses C-value [13] – a

[3] TerMine service is provided by the UK National Centre for Text Mining (NaCTeM, `http://www.nactem.ac.uk/`). NaCTeM is operated by the University of Manchester.

domain-independent method for automatic term recognition (ATR) which combines linguistic and statistical analyses with the emphasis on the statistical part. Using linguistic analysis all candidate terms in a given text are enumerated by applying part-of-speech tagging, extracting word sequences based on adjectives/nouns, and stop-list. A candidate term is assigned to a bag of terms by means of the statistical analysis that uses the following four characteristics:

- The occurrence frequency of the candidate term
- The frequency of the candidate term as part of other longer candidate terms
- The number of these longer candidate terms
- The length of the candidate term

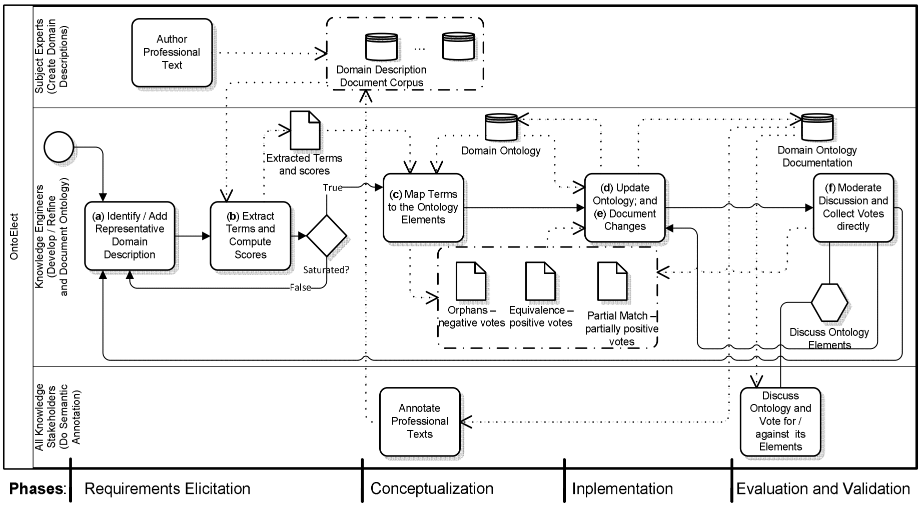
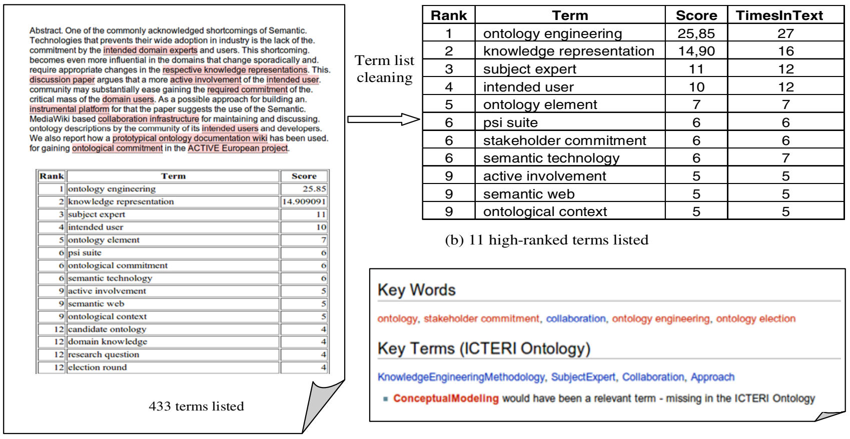

OntoElect
Subject Experts (Create Domain Descriptions)
Knowledge Engineers (Develop / Refine and Document Ontology)
All Knowledge Stakeholders (Do Semantic Annotation)
Author Professional Text
Domain Description Document Corpus
Extracted Terms and scores
Domain Ontology
Domain Ontology Documentation
(a) Identify / Add Representative Domain Description
(b) Extract Terms and Compute Scores
True
False
Saturated?
(c) Map Terms to the Ontology Elements
(d) Update Ontology; and (e) Document Changes
(f) Moderate Discussion and Collect Votes directly
Orphans – negative votes
Equivalence – positive votes
Partial Match – partially positive votes
Discuss Ontology Elements
Annotate Professional Texts
Discuss Ontology and Vote for / against its Elements
Phases:
Requirements Elicitation
Conceptualization
Inplementation
Evaluation and Validation

Fig. 3. OntoElect workflow

The processing pipeline is illustrated by the example based on a single document from the ICTERI document corpus ([1]) in Fig. 4.

The results of term mining are provided in several forms. All the terms defined in the text are shaded grey (upper part of Fig. 4a). The information about the overall number of the terms mined from the text is also given (433 terms listed – in the bottom of Fig. 4a). The terms are also presented in the table view, each preceded by the assigned rank number and followed by the statistical score measure (lower part of Fig. 4a). The rank of a term means the position of each term in the table sorted by the score; the rank values of the terms with the same score are equal. The scores are computed automatically using the term recognition technique [13] which uses the information about the frequencies of term occurrence.


Abstract. One of the commonly acknowledged shortcomings of Semantic. Technologies that prevents their wide adoption in industry is the lack of the. commitment by the intended domain experts and users. This shortcoming. becomes even more influential in the domains that change sporadically and. require appropriate changes in the respective knowledge representations. This. discussion paper argues that a more active involvement of the intended user. community may substantially ease gaining the required commitment of the. critical mass of the domain users. As a possible approach for building an. instrumental platform for that the paper suggests the use of the Semantic. MediaWiki based collaboration infrastructure for maintaining and discussing. ontology descriptions by the community of its intended users and developers. We also report how a prototypical ontology documentation wiki has been used. for gaining ontological commitment in the ACTIVE European project.

Rank	Term	Score
1	ontology engineering	25.85
2	knowledge representation	14.909091
3	subject expert	11
4	intended user	10
5	ontology element	7
6	psi suite	6
6	ontological commitment	6
6	semantic technology	6
9	active involvement	5
9	semantic web	5
9	ontological context	5
12	candidate ontology	4
12	domain knowledge	4
12	research question	4
12	election round	4

433 terms listed

Term list cleaning

Rank	Term	Score	TimesInText
1	ontology engineering	25,85	27
2	knowledge representation	14,90	16
3	subject expert	11	12
4	intended user	10	12
5	ontology element	7	7
6	psi suite	6	6
6	stakeholder commitment	6	6
6	semantic technology	6	7
9	active involvement	5	5
9	semantic web	5	5
9	ontological context	5	5

(b) 11 high-ranked terms listed

Key Words

ontology, stakeholder commitment, collaboration, ontology engineering, ontology election

Key Terms (ICTERI Ontology)

KnowledgeEngineeringMethodology, SubjectExpert, Collaboration, Approach

- **ConceptualModeling** would have been a relevant term - missing in the ICTERI Ontology

(a) Statistical results provided by TerMine

(c) Selected KeyWords and KeyTerms by the authors

Fig. 4. An example of the data processed in the term extraction activity

This technique extracts a bag of terms which have to be further cleaned by filtering out those highly ranked, but having no substantial contribution to the semantics of the domain (for example names, affiliations, cities, etc). This is performed manually by a knowledge engineer. For the example in Fig. 4 the top part of the result of the manual cleaning of the term list is pictured in Fig. 4b. The comparison of the extracted terms with the results of the manual annotation of this very document by the concepts of the ICTERI scope ontology (Fig. 4c) hints that the used term extraction technique and tool yield reasonably correct results. Those results however need to be further mapped to the elements of the ontology. The votes are then computed in the subsequent conceptualization phase based on the mappings, confident factors, and associated term scores.

Conceptualization in OntoElect is a manual work for a knowledge engineer done by (**c**) mapping the extracted terms to the elements of the existing ontology. If OntoElect is used in the refinement phase of the lifecycle, this existing ontology is the previous revision that is being refined. If however a new ontology is initially being built, a third party reference domain ontology could be used as the target for this mapping. In the latter case OntoElect also allows checking if this third party ontology fits the stakeholders needs well enough and could therefore be re-used, and to which extent.

The mapping of a particular term comprises several consecutive activities, all elaborating a wiki page describing this term:

- (**c1**) Elaborating a natural language definition of the term based on the document corpus and, possibly, external resources
- (**c2**) Creating a structural context for the term by specifying its properties based on the analysis of the elaborated definition
- (**c3**) Categorizing the structural context of the term by attributing it to concepts, properties, or individuals; and

- (**c4**) Mapping the term (using its structural context representation) to the relevant elements of the ontology by introducing new equivalence, membership[4], meronymy, or association relationships and providing confidence factors for them

The mappings introduced in (**c4**) are further interpreted as positive votes. The absence of the mappings for the term is interpreted as a negative vote – indicating the shortcoming in the ontology. The value of a vote is computed as a function of: (i) the normalized score of the corresponding term indicating its frequency of use in the document corpus; and (ii) the confidence factor. Each different term mapping counts as a separate vote. The details are given in Section 3.3.

The fitness of the ontology to the stakeholder requirements is further computed as a sum of votes. A more detailed analysis of the votes in favour of the parts of the ontology may also reveal: (i) which is the most fitting part of the theory; (ii) which part is of a minor fitness and may be dropped; (iii) what is missing and needs to be added to the ontology. These updates to the ontology are further done and documented in the implementation phase.

The changes suggested by the computed stakeholder votes are implemented in the ontology – activity (**d**). For that OntoElect does not provide any specific recommendation of a working pattern or a software tool for a knowledge engineer. In our development case studies ontology implementation is a two step process. The first step is (**d1**) updating the conceptual model coded as a UML 1.4 class diagram [22] using the ArgoUML editor.[5] Protégé ontology editor is used in the second step (**d2**) for coding the ontology in OWL 2 specified with an account for DL restrictions [23]. The transformation patterns from UML to OWL follow the recommendations of [24] and are described in detail at the ICTERI ontology Wiki[6]. OntoElect is more specific in recommending a way for documenting the ontology under development. It suggests that the ontology is documented in a set of Semantic MediaWiki[7] pages. Some of those pages provide the overviews of the ontology modules, but the rest, which are the majority, are dedicated to documenting the concepts – one page per concept. A documentation wiki page of a particular concept contains:

- The natural language definition of the concept
- The UML diagram of the concept's structural context
- The description of the concept's properties grouped according to the property types: datatype and object properties

The last phase in the flow – Evaluation and Validation – is beyond the focus of this paper. Therefore the objectives of this phase are only outlined. The activities within this phase use a Web 2.0 based approach for the semantic tagging of the documents belonging to the domain description resources. In fact this last phase aims at closing

[4] Membership is the relationship connecting an individual assertion to the corresponding element in the ontology schema. It is also known as an is_a relationship.

[5] ArgoUML is an open source UML modeling tool (`http://argouml.tigris.org/`).

[6] `http://isrg.kit.znu.edu.ua/icteriwiki/index.php/Naming_Convention,_Notations,_Transformations`

[7] `http://semantic-mediawiki.org/`

the loop and making the knowledge stakeholders (a) informed about and (b) committed to the ontology developed in the previous phases. Therefore we attempt to actively involve the community of the knowledge stakeholders in these activities and provide their semantic tags directly – using a collaborative platform for discussion and tagging as described in [1]. The semantic tags allow us to measure the ratio of satisfaction/dissatisfaction by the stakeholders [4] – leading to the estimation of their potential commitment to the ontology. The platform is based on the Semantic MediaWiki.

3.3 OntoElect Measurable Objectives and Metrics

The important questions for OntoElect as a methodology are:

- How to ensure that the chosen document corpus fully and correctly represents the **K**-s of the domain knowledge stakeholders?
- How to represent and measure as votes the fitness of the ontology or its elements to the stakeholders **K**-s without explicitly asking about their opinions?

The answers to those questions required the introduction of the two groups of metrics – one for representativeness and the other for fitness using votes.

Saturation as a Metric for Representativeness. OntoElect follows the approach of assessing the saturation of the extracted bag of terms for estimating the completeness of the domain coverage and hence the representativeness of the document corpus used for term extraction. Let $\mathbf{D} = D_1, ..., D_{i+1} = D_i \bigcup \Delta_{i+1}, ..., D_n$ be the sequence of the samples of the document corpus which are built incrementally – i.e. each subsequent sample D_{i+1} in the sequence is created by adding a number of new relevant documents (Δ_{i+1}) to the previous sample D_i. Let $T_i = \{(t_j^i, s_j^i, ns_j^i)\}$ be the bag of terms and their normalized scores extracted from the sample D_i. A normalized score ns_j^i of a term t_j^i is computed as $ns_j^i = s_j^i / s_{\max}^i$ where $s_{\max}^i$ is the maximal score among all the terms in the bag – displayed in the first line of the score table (Fig. 4b).

Definition 1 (Termhood). A bag of terms T_i is the termhood related to D_i if T_i contains only:

- Significant terms – i.e. those scored above the significance threshold ε_s; and
- Valid terms – i.e. those after filtering out the terms that are highly ranked, but have no substantial contribution to the semantics of the domain (see also Section 3.2).

One reasonable way to choose ε_s is to ensure that the terms in the termhood reflect the majority of the stakeholders' opinions. This could be done by taking in those terms from the top of the bag of terms, sorted by term score, having the sum of the scores slightly higher than the 50 per cent of the sum of all scores in the bag of terms. To account for the consensual nature of the requirements to be elicited, the sum of all scores may count only the scores higher than 1.

Definition 2 (Identical Terms). Terms t^i, t^{i+1} are considered (lexically) identical, denoted as $(t^i, t^{i+1})_{\equiv}$, if $(t^i, t^{i+1}) : t^i \in T_i, t^{i+1} \in T_{i+1}, t^i \equiv t^{i+1}$, where '$\equiv$' is the lexical equivalence operator.

Definition 3 (Orphan Terms). Term $t^i \in T_i$ is a dropped term, denoted as $(t^i, \_)$ – orphan, if $(t^i \in T_i) \wedge (\neg \exists t^{i+1} \in T_{i+1} : t^i \equiv t^{i+1})$. Term $t^{i+1} \in T_{i+1}$ is a newly extracted term, denoted as $(\_, t^{i+1})$ –orphan, if $(t^{i+1} \in T_{i+1}) \wedge (\neg \exists t^i \in T_i : t^i \equiv t^{i+1})$.

It could be stated that $(t^i, \_)$ –orphans will not appear for the termhood pairs (T_i, T_{i+1}) of **D** due to its incremental nature.

Definition 4 (Termhood Difference). The termhood difference $thd(T_i, T_{i+1})$ between the termhoods T_i, T_{i+1} is the sum of the differences in normalized weights in the pairs of identical terms $(t^i, t^{i+1})_{\equiv}$ and $(\_, t^{i+1})$ –orphans computed using the following THD algorithm.

```
Algorithm THD. Compute Termhood Difference
Input: the termhoods $T_i, T_{i+1}$
Output: $thd(T_i, T_{i+1})$
for $k = 1, \|T_{i+1}\|$
    $ident := .F.$
    for $m = 1, \|T_i\|$
        if $(t^i_m, t^{i+1}_k)_{\equiv}$ then do $thd := thd + |ns^i_m - ns^{i+1}_k|$; $ident := .T.$ end do
    end for
    if $ident := .F.$ then $thd := thd + |ns^{i+1}_k|$
end for
```

Definition 5 (Document Corpus Saturation Criterion). $\mathbf{D} = D_1, \ldots, D_{n-1}, D_n$ is considered saturated if $thd(T_{n-1}, T_n) < \varepsilon_{st}$, where: ε_{st} is the saturation threshold chosen empirically by a knowledge engineer for the given domain; T_{n-1}, T_n are the termhoods related to the two final document samples D_{n-1}, D_n of **D**.

It is assumed in our work that the sequence of *thd* values monotonically going down below ε_{st} indicates that D_n is a complete document corpus possessing sufficient representativeness. Non-monotonicity of *thd* values sequence signals that the corresponding Δ_{i+1} is either not very relevant to the domain or is a valuable addition containing the terminology not used in the previous samples (D_i). Anyhow, saturation indicates that the chosen document corpus is complete. Unfortunately the question about the correctness of the chosen document corpus can not be answered using a similar technique. Therefore OntoElect recommends involving human subject experts for checking correctness.

Term Mappings, Votes, and Ontology Fitness. For measuring the fitness of the entire ontology or its particular constituents OntoElect recommends to use votes. Votes are computed based on:

- The scores of the relevant terms in the termhood
- The mappings of the terms to the ontology elements

Let $O = \{C, P, I, A\}$ be an ontology comprising the specifications of concepts (*C*), properties (*P*), individual assertions (*I*) that are the members of the elements of *C* or *P*, and axioms (*A*) over *C*, *P*, or *I*. Let also *T* be the termhood extracted from the final sample of the document corpus D_n of **D**.

Definition 6 (Term Mapping). A term mapping of the term $t \in T$ to ontology *O* is the function that establishes a relationship between *t* and the element of *O*: $\mu = (t, r, o, cnf)$, where: *r* is the relationship type $r \in \mathbf{R} = \{equivalence, membership, subsumption, meronymy, association\}$; *o* is the element of O; and *cnf* is the confidence factor with a value from $[0,1]$. $\mathbf{M}_o = \{\mu\}$ is the set of all term mappings to the ontology element *o*.

Definition 7 (Positive Vote). A positive vote v_o for an ontology element $o \in O$ is the value reflecting the evidence of referring to *o* by the term $t \in T$ through the term mapping μ:

$$v_o = \sum_{\mathbf{M}_o} ns \times w(r) \times cf\ , \tag{2}$$

where *ns* is the normalized score of *t*, *cf* is the confidence factor of the respective mapping μ, and $w(r)$ is the weight of the mapping based on the type of the relationship *r* of μ. The weights are introduced to reflect that different types of mappings could be regarded as the arguments of different strength in favour of this ontological element. Indeed, if a term is *equivalent* to the element then it is a strong direct argument in favour of the element. However a statement about being an individual member of the element, a direct subsumption to an element, a part of an element, or having an association to an element is considered as a weaker argument. So the weights for our experiments were specified as: equivalence – 1.0; membership – 0.7; subsumption, meronymy – 0.5; association – 0.3. Those values may further be reconsidered if any experimental evidence is collected in this respect. Direct subsumption mappings to very abstract elements in the ontology should however be avoided. For example, all concepts, and therefore the terms categorized as concepts, subsume to the root concept of a Thing present in any OWL ontology. This subsumption mapping however has very little to do with domain semantics and therefore should not be counted as a vote. Valid direct subsumption mappings should be sought to the most specific possible ontology elements.

Indirect subsumption mappings could further be accounted for propagating votes up the concept hierarchy as denoted below in Definition 11. Propagated votes may be used to further clarify the distribution of the fitness up the subsumption hierarchy of the ontology – refining our result reported in [4].

Positive votes are further used for computing the fitness factors of the elements in *O*.

Definition 8 (Ontology Element Fitness). The fitness ϕ_o of the ontology element $o \in O$ is the value reflecting the strength of the evidence in favour of *o* provided by the positive votes:

$$\phi_o = \sum_{o \in O} v_o \,. \tag{3}$$

Let the set of properties of *O* be $P = \{P^D, P^O\}$, where P^D is the set of datatype properties, P^O is the set of object properties. Let P_c be the set of properties of the concept $c \in C$, I_c – the set of individuals which are the members of the concept $c \in C$, and A_c – the set of axioms over P_c and I_c. Then the structural context of c:

$$SC_c = c \cup P_c \cup I_c \cup A_c \,. \tag{4}$$

Definition 9 (Structural Context Fitness). The fitness ϕ_{SC_c} of the structural context of the concept *c* is the sum of the positive votes in favor of the elements in the structural context of *c*:

$$\phi_{ST_c} = \sum_{o \in SC_c} v_o \,. \tag{5}$$

Let us suppose that the mappings of the terms from $T^{miss} \subset T$ can not be specified because *O* does not contain the elements semantically matching those terms. Such terms are further referred to as missing terms.

Definition 10 (Negative Vote). A negative vote with respect to *O* is the vote based on the term $t_i \in T^{miss}$ pointing out that t_i is not described by *O*: $v_{t_i}^- = -ns_i$.

So far only direct positive votes with respect to ontology elements have been discussed. So, the overall ontology fitness computed based on these votes reflects only the arguments focused on an element and without any influence on the surrounding of this element. This however might not be fully correct with respect to the fitness of the surrounding elements. Indeed, let us for example assume that the concept SubjectExpert gets a vote. Then it may be expected that the concept Person, subsuming SubjectExpert, also qualifies for the part of the value of this vote. A straightforward reason is that, due to the subsumption relationship, the more specific concept inherits the properties of the more abstract concept in the subsumption hierarchy. So the vote has to be propagated up the hierarchy with attenuation – factored empirically or possibly aligned with the proportion of the inherited properties in each individual case.

Definition 11 (Propagated Vote). A propagated vote v_o^p for an ontology element $o \in O$ is the value reflecting the contribution of *o* to the semantics of the ontology element o^{sub} subsumed by *o*:

$$v_o^p = att \times v_{o^{sub}} , \tag{6}$$

where *att* is the attenuation coefficient chosen empirically.

Definition 12 (Ontology Fitness). The fitness of ontology *O* with respect to the termhood *T* is the sum of all positive, propagated, and negative votes:

$$\Phi_O = \sum_{o \in O} v_o + \sum_{o \in O} v_o^p + \sum_{t \in T^{miss}} v_t^- . \tag{7}$$

4 Experimental Set-Up

OntoElect methodology described in Section 3 has been evaluated using the current revision of the ICTERI scope ontology and the document samples composed incrementally of the papers of ICTERI 2011, 2012, and 2013.

Domain Knowledge Stakeholders. It is rather easy to identify the knowledge stakeholders in the domain of our case study. Those are the authors of the papers submitted to ICTERI conference series within the three consecutive years and those who attended the conference without submitting a paper. The overall number of those individual ICTERI community members is 367. It has been obtained from the conference management system and the records of the organizing committee. Among the knowledge stakeholders, the subject experts in the domain are also identified – those whose papers have been accepted for the publication. Hence, the selection of the subject experts is done based on the decisions by the conference program committees. The overall number of the subject experts is 231 which are 62.94 per cent of the knowledge stakeholders.

Domain Description Resource. Two document collections are available in the ICTERI domain: (i) papers submitted to the conference series – 323 papers; and (ii) papers accepted for publication – 120 papers. It has been decided that the pool of the accepted papers sufficiently covers the domain as those papers have been selected based on the expert opinions by the members of the conference program committees.

Domain Ontology for Refinement. The revision of the ICTERI scope ontology[8] available as an input for our experiment has been developed in the previous two consecutive years. It contains 109 concepts.

Our experimental study is organized in two consecutive phases: (i) Requirements Elicitation; and (ii) Conceptualization. Phases (iii) Implementation and (iv) Evaluation and Validation, though performed, are beyond the focus of this paper and therefore not presented below.

[8] The description of this revision of the ICTERI scope ontology can be retrieved from `http://isrg.kit.znu.edu.ua/icteriwiki/index.php/Top-level_view_of_the_ICTERI_Scope`

4.1 Requirements Elicitation

The objectives of the Requirements Elicitation phase are: (i) checking the representativeness of the document corpus of the accepted ICTERI papers; and (ii) building the termhood T of the bag of terms T_n extracted from the final document sample D_n. The sequence of document samples $\mathbf{D}$ is formed as indicated in Table 1.

Table 1. The sequence of document samples containing ICTERI accepted papers

$i+1$	Composition of D_{i+1} (Δ_{i+1})	No of papers
1	D_1 = All published papers of ICTERI 2011	15
2	D_1 plus first 20 published papers of ICTERI 2012	35
3	D_2 plus the rest of published papers of ICTERI 2012	55
4	D_3 plus first 20 published papers of ICTERI 2013	75
5	D_4 plus the rest of published papers of ICTERI 2013 up to the TerMine constraint for the size of the processed plain text (2Mb)	80

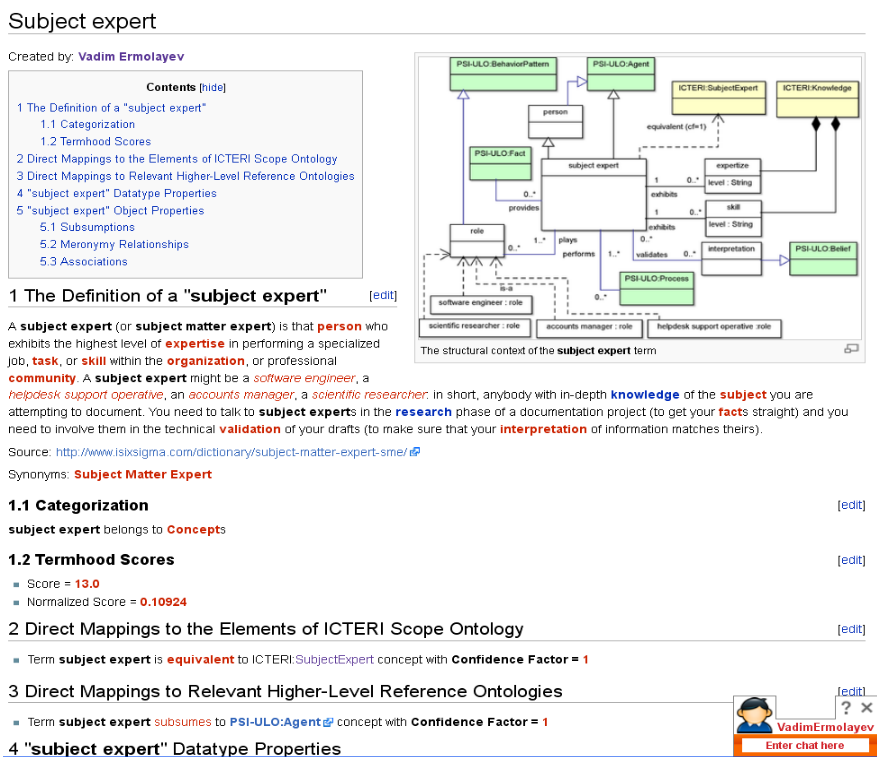
The representativeness of $\mathbf{D}$ is measured using the saturation criterion (Definition 5). The termhoods T_i, T_{i+1} for computing termhood differences have been created as specified in Definition 1.

4.2 Conceptualization

The objective of the Conceptualization phase is to obtain the values of the votes and finally to analyse the fitness of the current revision of the ontology to the requirements by the paper authors learned from their papers in the previous phase.

For computing the votes, the mappings are specified for the terms from the termhood T built over the final document sample in $\mathbf{D}$. Term mapping is done as described in Section 3.2 by performing activities (**c1** – **c4**) for each term starting from the top of the scorelist and ending at the significance threshold. Wiki pages of the `category:Term` are created for the illustrative subset of the processed terms. Figure 5 pictures an example of the result of the Conceptualization phase for the term "subject expert" (see also Fig. 4).

The mappings are manually specified by the knowledge engineers for the terms from T as stated in Definition 6. Following equation (2), the positive (direct) votes are further computed for all those elements of the ICTERI scope ontology which are found in the mappings. Further, the direct votes are propagated up the subsumption hierarchies according to (6). The attenuation coefficient is chosen equal to 0.5. Fitness factors are then computed for all the concepts of the ontology, affected by the mappings, using (3). The set of the missing terms $T^{miss} \subset T$ is extracted and the negative votes with respect to the input revision of the ontology are computed following Definition 10. Finally the fitness of the ontology is computed using equation (7).



Subject expert

Created by: **Vadim Ermolayev**

Contents [hide]

The structural context of the **subject expert** term

1 The Definition of a **"subject expert"** [edit]

A **subject expert** (or **subject matter expert**) is that **person** who exhibits the highest level of **expertise** in performing a specialized job, **task**, or **skill** within the **organization**, or professional **community**. A **subject expert** might be a *software engineer*, a *helpdesk support operative*, an *accounts manager*, a *scientific researcher*: in short, anybody with in-depth **knowledge** of the **subject** you are attempting to document. You need to talk to **subject experts** in the **research** phase of a documentation project (to get your **fact**s straight) and you need to involve them in the technical **validation** of your drafts (to make sure that your **interpretation** of information matches theirs).

Source: http://www.isixsigma.com/dictionary/subject-matter-expert-sme/

Synonyms: **Subject Matter Expert**

1.1 Categorization [edit]

subject expert belongs to **Concept**s

1.2 Termhood Scores [edit]

- Score = **13.0**
- Normalized Score = **0.10924**

2 Direct Mappings to the Elements of ICTERI Scope Ontology [edit]

- Term **subject expert** is **equivalent** to ICTERI:SubjectExpert concept with **Confidence Factor = 1**

3 Direct Mappings to Relevant Higher-Level Reference Ontologies [edit]

- Term **subject expert** subsumes to **PSI-ULO:Agent** concept with **Confidence Factor = 1**

4 **"subject expert"** Datatype Properties

Fig. 5. An example of the ICTERI wiki page (upper part of the page) describing the conceptualization of the term "subject expert". See also http://isrg.kit.znu.edu.ua/icteriwiki/index.php/subject_expert.

A more detailed analysis of the ontology is further undertaken to provide the recommendations for the consecutive Implementation phase. Using the fitness factors for the structural contexts (5) we identify:

- The most valuable part of the ontology which will remain unchanged in the upcoming revision
- The part of the ontology which has low fitness contribution and will be recommended to be dropped
- The selection of the missing terms (T^{miss}) that may be regarded as a valuable addition to the ontology improving its fitness noticeably. This addition will be recommended for implementation.

Further, in the Implementation phase we focus on the recommendation on the valuable addition to the ontology. Those missing terms which are categorized as concepts are added to the ICTERI scope ontology as concepts using term conceptualizations as the instructions for OWL coding. New concepts are also documented in the ICTERI Wiki by moving and minimal editing of the respective wiki pages describing term conceptualizations.

5 Results and Discussion

In this section we present the results of the experiment. The discussion is structured along the phases of OntoElect and corresponding measurable objectives.

5.1 Representativeness of the Document Corpus

The sequence of document samples $\mathbf{D} = D_1, D_2, ..., D_5$ described in Table 1 has been processed in the Requirements Elicitation phase of our experimental study. For each document sample D_i we:

- Extracted the Bag of Terms using the TerMine service, sorted by term scores s_j^i, and computed normalized scores ns_j^i
- Deleted all the terms with the score value less or equal to 1 as those which do not represent a consensual opinion of the subject experts and are therefore not significant
- Computed the significance threshold ε_s as described in Section 3.3 using a variation of the binary search algorithm
- Formed the termhood T_i by deleting the terms scored below ε_s
- Computed the absolute termhood difference value $thd(T_{i-1}, T_i)$ assuming that $T_0 = \varnothing$ and using the THD algorithm (Section 3.3)
- Computed the relative termhood difference value $thdr = thd(T_{i-1}, T_i) / \sum_{T_i} ns_j^i$

The results are summarized in Table 2.

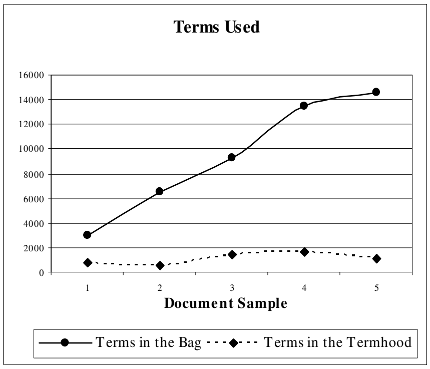
Table 2. The processed sequence of document samples

Document Sample (Table 1)	No of Papers	Terms in the Bag	Terms in the Termhood	% Terms	ε_s	*thd*, value	*thdr*, %
D_1	15	2930	742	25.32	2.00	66.239	100.00
D_2	35	6471	603	9.32	3.17	21.4239	23.95
D_3	55	9319	1402	15.04	4.00	7.5833	6.65
D_4	75	13514	1708	12.64	4.00	6.1689	4.64
D_5	80	14597	1151	7.89	4.64	3.7749	3.52

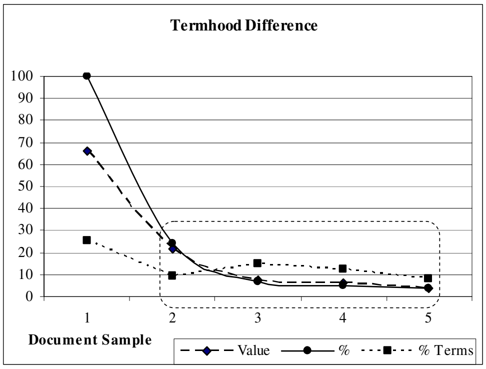
The analysis of the results reveals several important findings. First of, though the number of terms extracted from the document samples increases quite substantially, the numbers of significant terms, forming the respective termhoods, is first growing much more slowly and finally goes down quite substantially in T_5 – see also Fig. 6a. If the proportions are taken into consideration, one may notice that the tendency of decreasing the percentage of the terms in a termhood versus terms in the bag could be observed starting from T_3 . In T_5 the substantial terms constitute only 7.89 per cent of the overall number of terms mined from D_5 . This is a very lucky result allowing us to assume that only this relatively small subset of terms expresses the qualifying majority of the opinions of the subject experts. The rest – insignificant – could be discarded for further processing phases. This conclusion is also reinforced by the tendency of the growth of the ε_s values.

The absolute (*thd*) and relative (*thdr*) termhood difference values indicate that the sequence of the document samples becomes saturated – as graphically illustrated in Fig. 6b and in a finer grained detail in Fig. 6c.

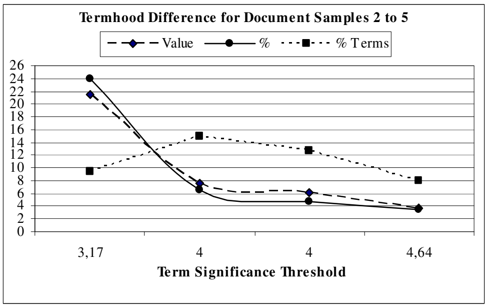
Indeed the relative difference between the termhoods T_4 and T_5 , both containing more than a thousand terms, is only 3.52 per cent. It is also important to notice that only the terms scored higher or equal to 4.64 were included in T_5 which is a relatively big significance compared to the previous samples and termhoods. So the weight of the high-scored terms has been substantially increased, which also confirms the saturation subsumption made. Hence, T_5 may be used as the representative termhood T for the subsequent Conceptualization phase.



(a) The proportion between the quantities of terns in the extracted Bags of Terms and respective Termhoods



(b) The saturation of the document corpus illustrated by the absolute and relative *thd* values and the percentage of terms scored above the significance threshold ε_s



(c) A closer view on the saturation diagrams (the area within the dashed rectangle in Fig. 6b) aligned with the values of the term significance threshold ε_s given in the X-axis. This view excludes the evident case of D_1 .

Fig. 6. Saturation of the document corpus

5.2 Term Conceptualizations, Mappings, and Votes

In the Conceptualization phase termhood *T* has been taken as the input. Before performing the activities (**c1-c4**), termhood cleaning and term merging has been undertaken.

By cleaning, the invalid, irrelevant, or insignificant entries, though sometimes scored highly, were discarded from further processing. The topical cases were: (i) phrases having no valid meaning in the context of ICTERI scope ontology – for example "a a" or "such approach" or the fragments of e-mail addresses accompanied by a valid shorter term; (ii) terms with very limited relevance and low significance – conference, publisher names. By merging the terms not identified as the same in the previous phase but still meaning the same are substituted by one term. Some merge cases are: (i) plural and singular of the same – for example "system analysis" and "systems analysis"; (ii) terms with permuted parts but meaning the same – "kiev taras shevchenko university" and "taras shevchenko kiev university"; (iii) terms using synonymic alternatives as their parts – "abstract automaton" and "abstract finite state machine". By cleaning and merging the number of terms in the termhood *T* has been decreased by 12.18 percent (from 1151 to 1023).

After cleaning we proceeded with performing activities (**c1-c4**) for each individual term starting from the top scored. We skipped the detailed elaboration of the natural language definitions for the majority of the cases as the meanings of those terms were clear enough. We also skipped the detailed elaboration of the structural contexts for the majority of the processed terms and focused on the straightforward and clear mappings development to lower the effort necessary to accomplish the experiment. The illustrative subset of the top scored 25 terms has however been elaborated in full detail[9].

The 1023 terms from *T* have been categorized as follows: concepts – 856; properties – 43; instances – 93; uncategorized – 31. The meaning of the 31 uncategorized terms was not full clear to the knowledge engineers who took part in the experiment. This lack of clarity prevented the determination of the category of those terms. The significance of those terms was however moderate to low and the quantity – not very large. So it has been decided that those terms are skipped for further processing.

As the ICTERI scope ontology is based on the PSI upper-level ontology (PSI-ULO) [25] and imports PSI-ULO, the mappings were created:

- To the elements of the ICTERI ontology
- If not possible – to the elements of the PSI-ULO
- And if not possible, the term was qualified as the member of T^{miss}

The result of the term mapping activity is illustrated in Table 3 which pictures in a condensed way the outcomes of the conceptualization activities for the first 14 most highly scored terms in the termhood. This illustration, though limited in volume, contains all the topical cases of the developed mappings.

[9] ICTERIWiki pages describing the conceptualizations of those top scored terms may be accessed from `http://isrg.kit.znu.edu.ua/icteriwiki/index.php/OntoElect_Methodology`

Table 3. A sample of the terms and their mappings (the top scored in the termhood T). For those terms conceptualizations have been built.

Rank	Term	Score	Norm. Score	Category				MT	Mapping						
									Type					To	cf
				Concept	Property	Instance	Uncat.		Equiv	Memb	Mero	Assoc	Subs		
				11	0	5	0	2	2	5	3	0	10		
1	software engineering (process)[10]	119.00	1.0000	✓					✓					ICTERI:SoftwareEngineeringProcess	1.0
	(discipline)					✓				✓				ICTERI:SubjectDomain	0.7
2	management	118.00	0.9916	✓									✓	PSI-ULO:Process	1.0
	(discipline)					✓				✓				ICTERI: SubjectDomain	0.7
3	programming language	108.43	0.9112	✓									✓	PSI-ULO:ImmaterialArtifact	1.0
4	key term	96.50	0.8109	✓				✓					✓	term (MT)	1.0
5	educational process	95.83	0.8053	✓									✓	PSI-ULO:Process	1.0
											✓			(comprises) ICTERI:TeachingProcess	0.9
6	computer science	92.37	0.7762	✓				✓			✓			(areaOf) science (MT)	0.8
7	software product	87.63	0.7364	✓									✓	PSI-ULO:ImmaterialArtifact	1.0
9[11]	information system	79.78	0.6704	✓									✓	ICTERI:SoftwareSystem	1.0
10	insertion modeling	79.04	0.6642	✓									✓	PSI-ULO:ProcessPattern	0.7
11	distance learning (R&D area)	78.13	0.6566	✓				✓			✓			computer science (MT) .subs. science (MT)	0.5
	(technology)												✓	ICTERI:InformationCommunicationTechnology	0.8
	(learning pattern)												✓	PSI-ULO:ProcessPattern	0.6
12	petri net	75.86	0.6375			✓				✓				PSI-ULO:ImmaterialArtifact	0.5
13	insertion machine	73.00	0.6134			✓				✓				PSI-ULO:ImmaterialArtifact	0.4
14	ksu feedback	71.67	0.6022			✓				✓				ICTERI:SoftwareSystem	1.0
15	software system	68.86	0.5786	✓					✓					ICTERI:SoftwareSystem	1.0
													✓	PSI-ULO:Holon	0.8

Legend: **MT** – missing term; **Equiv** – equivalent to; **Memb** – individual member of; **Mero** – part/whole of; **Assoc** – associated to; **Subs**: subsumes to.

[10] Some terms have been split reflecting different facets in their intended meaning, for example software engineering as a process and as a discipline.
[11] Term "a a" ranked 8 and scored 87.5 by TerMine has been removed as noise.

The numbers of the terms within the categories and the numbers of mapping types in Table 3 are given only for the chosen sample but not for the whole set of the processed terms of *T*.

It is worth noticing that only 3 of the 14 terms in the top scored sample (Table 3) were identified as missing terms. This number would have been substantially higher if the ICTERI scope ontology was not based on PSI-ULO. This proportion is also valid for the rest of *T* with a minor increase in the bottom of the score list where the terms become too specific. One more interesting observation is that the terms categorized as concepts tend to group closer to the top of the termhood than the terms categorized as properties. This is why there are no properties among the top scored terms in Table 3.

The votes for the ontology elements involved in the developed mappings were computed using equation (2) for direct positive votes and equation (6) for the propagated positive votes. Table 4 lists the direct positive votes for the ontology elements appearing in the mappings of the terms mentioned in Table 3. The values of the votes are given per mapping and also grouped for each concept.

Table 4. Ontology elements, mentioned in the mappings of the top scored terms in Table 3, and their votes

Mapping	To Concept	Votes		
		DPV		Negative
		Mapping	Concept	
distance learning (R&D area) .mero (areaOf).	(MT) computer science			-0.6566
computer science .mero (areaOf).	(MT) science			-0.7762
key term .subs.	(MT) term			-0.8109
distance learning (technology) .subs.	ICTERI:InformationCommunicationTechnology	0.2626	0.2626	
software engineering (process) .equiv.	ICTERI:SoftwareEngineeringProcess	1.0000	1.0000	
information system .subs.	ICTERI:SoftwareSystem	0.3352	1.3353	
ksu feedback .memb.		0.4215		
software system .equiv.		0.5786		
software engineering (discipline) .memb.	ICTERI:SubjectDomain	0.4900		
management (discipline) .memb.		0.4859	0.9759	
educational process .mero (comprises).	ICTERI:TeachingProcess	0.3624	0.3624	
programming language .subs.	PSI-ULO: ImmaterialArtifact	0.4556	1.4501	
software product .subs.		0.3682		
petri net .memb.		0.2231		
insertion machine . memb.		0.1718		
software system .subs.		0.2314		
management (process) .subs.	PSI-ULO:Process	0.4958	0.8985	
educational process .subs.		0.4027		
insertion modeling .subs.	PSI-ULO:ProcessPattern	0.2325	0.4295	
distance learning (learning pattern) .subs.		0.1970		
Legend: DPV – direct positive vote		**Subtotal:**	**6.7143**	**-2.2437**

Table 4 also shows the negative votes associated with the missing terms. For the illustrative subset of terms, the direct positive contribution to ontology fitness is 6.71. The sum of the negative votes is 2.24 which results in the proportion of 33.3 per cent to the direct positives.

Table 5. Subsumption hierarchies and propagated votes

DPV	**Concept**	**Subsumption Hieracrchy**	**PrV**
0.9759	ICTERI:SubjectDomain	ICTERI:Domain	0.4879
		PSI-ULO:Context	0.2440
0.2626	ICTERI:InformationCommunication Technology	ICTERI:Technology	0.1313
		PSI-ULO:ProcessPattern	0.0657
		PSI-ULO:Pattern	0.0328
		PSI-ULO:Rule	0.0164
		PSI-ULO:ImmaterialObject	0.0082
		PSI-ULO:Object	0.0041
		PSI-ULO:Holon	0.0021
1.0000	ICTERI:SoftwareEngineeringProcess	PSI-ULO:Process	0.5000
		PSI-ULO:Event	0.2500
1.3353	ICTERI:SoftwareSystem	PSI-ULO:ImmaterialArtifact	0.6677
		PSI-ULO:ImmaterialObject	0.3338
		PSI-ULO:Object	0.1669
		PSI-ULO:Holon	0.0835
0.3624	ICTERI:TeachingProcess	PSI-ULO:Process	0.1812
		PSI-ULO:Event	0.0906
1.4501	PSI-ULO: ImmaterialArtifact	PSI-ULO:ImmaterialObject	0.7251
		PSI-ULO:Object	0.3625
		PSI-ULO:Holon	0.1813
0.8985	PSI-ULO:Process	PSI-ULO:Event	0.4492
0.4295	PSI-ULO:ProcessPattern	PSI-ULO:Pattern	0.2147
		PSI-ULO:Rule	0.1074
		PSI-ULO:ImmaterialObject	0.0537
		PSI-ULO:Object	0.0268
		PSI-ULO:Holon	0.0134

Legend: **DPV** – direct positive vote; **PrV** – propagated vote.

Table 5 shows how those direct votes (Table 4) are propagated along the subsumption hierarchies of the ontology with the chosen attenuation coefficient $att = 0.5$. As summarized in Table 6, the contribution of those propagated votes to the fitness of the ontology equals to 5.40. Overall the sum of the positive votes amounts to 12.31 with the ratio of 87.8 per cent to the negatives. Those ratios are roughly aligned with the estimation of the ICTERI ontology fitness, computed as satisfaction/dissatisfaction ratio, we reported in [4].

The fitness values for the structural contexts within the ontology have not yet been computed as the development of the structural contexts for the terms in *T* is ongoing. The overall ontology fitness value has therefore not yet been computed. However, even the illustrative subset[12], mentioned above, showcases:

[12] Ontology documentation pages for these concepts may be retrieved from `http://isrg.kit.znu.edu.ua/icteriwiki/index.php/OntoElect_Methodology`

- The concepts which received high vote values; and
- The missing terms which could be used as a valuable addition to the ontology due to their high scores

Table 6. Summary of the propagated Votes

Concept	PrV
ICTERI:Domain	0.4879
ICTERI:Technology	0.1313
PSI-ULO:Context	0.2440
PSI-ULO:Event	0.7898
PSI-ULO:Holon	0.2803
PSI-ULO:ImmaterialArtifact	0.6677
PSI-ULO:ImmaterialObject	1.1208
PSI-ULO:Object	0.5603
PSI-ULO:Pattern	0.2475
PSI-ULO:Process	0.6812
PSI-ULO:ProcessPattern	0.0657
PSI-ULO:Rule	0.1238
Subtotal:	**5.4003**

Legend: **PrV** – propagated vote.

It is also worth reporting about the effort spent for processing the illustrative subset of terms in the conceptualization phase. A novice knowledge engineer with a limited experience in conceptual modelling and ontology engineering has been intentionally assigned for this task. This person also had no background knowledge of the ICTERI domain. We made this assignment to check if the OntoElect methodology can be easily taken up by a newcomer and to find out what the effort would be. Our novice knowledge engineer has been instructed by an expert both in OntoElect and the domain of ICTERI by providing a running example. The example was based on the complete conceptualization of one term (Subject Expert) taken from the middle of the termhood scorelist. The result of the conceptualization is the wiki page describing the term (Fig. 5).

The average effort spent for conceptualizing the terms listed in Table 3 by our newcomer was 2.75 hours per term. Those of course included the time needed for learning the tools used in the workflow and differed quite substantially from term to term. For those terms that were not known to the engineer before and required checking the papers for compiling an adequate definition, the effort was up to four times higher than for the known terms. The wiki pages describing the terms were further checked by the expert and some corrections were made. This resulted in the additional effort of approximately 0.2 hours per page. It is expected that at the tail of the learning curve the effort will be indicatively about 1.0 – 1.2 hours per term conceptualization. An experiment for verifying that will be required in the future.

Having reported this we admit that the high amount of the manual work needed to conceptualize all the terms in a termhood makes the activity very laborous. There are ways however to reduce the cost of the conceptualization phase of OntoElect.

One promising approach is to develop and provide the software tools for partially automating the most complex and effort consuming tasks of:

- Developing the structural context of a term from its natural language definition
- Finding the mappings of the term to the ontology elements

The automation of the discovery of the mappings could for example be done using our structural difference discovery engine [26].

6 Concluding Remarks and Future Work

This paper presented the OntoElect methodology for ontology engineering which is focused on ensuring that the resulting knowledge representation artefact fits to the requirements of the domain knowledge stakeholders as much as possible. OntoElect thus facilitates reaching a higher stakeholder commitment to the developed ontology. Ontology fitness in OntoElect is quantified using the collection of the interpretations of the domain by the subject experts in this domain. Those interpretations are not elicited directly from the experts, as it is often very hard. Instead, the terms describing the domain are extracted from the representative document corpus. The paper described how the representativeness could be assessed using the saturation-based metric. This part of the methodology, referred to as the Requirements Elicitation phase, has been evaluated in the experiment. The experimental setting for this phase has been presented in Section 4.1. The results proving the representativeness (saturation) of the document corpus of ICTERI papers were presented in Section 5.1.

The collection of terms is further processed by a knowledge engineer, and the mappings are developed between the terms and the elements of the developed ontology. Those mappings are interpreted as the stakeholder votes. The votes are further used for computing the fitness of the characteristic parts in the ontology and the whole ontology. This part of the methodology, referred to as the Conceptualization phase, has also been evaluated in our experiment. The settings for the Conceptualization phase were described in Section 4.2 and the results – in Section 5.2.

The experiment reported in this paper is not accomplished yet as not all the concepts are processed fully. This work is currently being continued. So, the final analysis of the ontology fitness, followed by the Implementation and Evaluation phases of OntoElect for the ICTERI scope ontology are in our short-term plan. In particular, for validating the approach we will compare the results of the Conceptualization phase with the results of the manual semantic annotation of the ICTERI papers by the domain knowledge stakeholders.

The ICTERI use case demonstrated that the domain description document corpus: (a) has become saturated quite quickly; and (b) is not very big in size. We consider it worth checking if this tendency is valid for the other use cases in the domains: (a) where saturation could not be reached that easily – for example those changing in time quite substantially; and (b) having much more bulky datasets, hence longer bags of terms. For that we planned the case study in the Bio-Design Automation domain based on the corresponding corpus of workshop and conference papers.

In this paper we reported about only a partial and shallow way of extracting knowledge from paper texts – as a bag of terms. A possible refinement to this solution could be sought in using a hybrid iterative knowledge extraction workflow proposed in [27]. This may partially automate the most laborious part of the conceptualization phase – structural context development. Further on this way, the mappings to the ontology elements could also be discovered partially automatically via the use of an appropriate ontology alignment tool, for example our structural difference discovery engine. Those improvements in the tool support are also planned for our future work.

Finally we plan undertaking an experiment with a group of subjects for measuring the learning curve for OntoElect and finding out ways of optimising the effort required for the conceptualization phase.

References

1. Tatarintseva, O., Ermolayev, V., Fensel, A.: Is Your Ontology a Burden or a Gem? – Towards Xtreme Ontology Engineering. In: Ermolayev, V., et al. (eds.) ICTERI 2011, vol. 716, pp. 65–81. CEUR-WS.org (2011)
2. Tatarintseva, O., Borue, Y., Ermolayev, V.: OntoElect Approach for Iterative Ontology Refinement: a Case Study with ICTERI Scope Ontology. In: Ermolayev, V., et al. (eds.) ICTERI 2012, vol. 848, p. 244. CEUR-WS.org (2011)
3. Tatarintseva, O., Ermolayev, V.: Refining an Ontology by Learning Stakeholder Votes from their Texts. In: Ermolayev, V., et al. (eds.) ICTERI 2013, vol. 1000, pp. 64–78. CEUR-WS (2013)
4. Tatarintseva, O., Borue, Y., Ermolayev, V.: Validating OntoElect Methodology in Refining ICTERI Scope Ontology. In: Kop, C., et al. (eds.) UNISON 2012. LNBIP, vol. 137, pp. 128–139. Springer, Heidelberg (2013)
5. Gupta, M., Li, R., Yin, Z., Han, J.: Survey on Social Tagging Techniques. SIGKDD Explorations 12(1), 58–72 (2010)
6. Uren, V., Cimiano, P., Iria, J., Handschuh, S., Vargas-Vera, M., Motta, E., Ciravegna, F.: Semantic Annotation for Knowledge Management: Requirements and a Survey of the State of the Art. Science. Services and Agents on the World Wide Web 4(1), 14–28 (2006)
7. Hunter, J., Khan, I., Gerber, A.: HarvANA – Harvesting Community Tags to Enrich Collection Metadata. In: Paepcke, A., Borbiha, J., Naaman, M. (eds.) 8th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 147–156. ACM, New York (2008)
8. Miles, A., Bechhofer, S.: SKOS Simple Knowledge Organization System reference. Technical report, W3C (2009)
9. Braun, S., Schmidt, A., Walter, A., Nagypal, G., Zacharias, V.: Ontology Maturing: a Collaborative Web 2.0 Approach to Ontology Engineering. In: Proc. Workshop on Social and Collaborative Construction of Structured Knowledge (CKC 2007) at WWW 2007, Banff, Canada, vol. 273, CEUR-WS (2007)
10. Siorpaes, K., Hepp, M.: Games with a Purpose for the Semantic Web. IEEE Intelligent Systems 23(3), 50–60 (2008)
11. Wong, W., Liu, W., Bennamoun, M.: Ontology learning from text: A look back and into the future. ACM Comput. Surv. 44(4), Article 20, 36 pages (2012), doi:10.1145/2333112.2333115
12. Buitelaar, P., Cimiano, P. (eds.): Ontology Learning and Population: Bridging the Gap between Text and Knowledge. IOS Press (2008)

13. Frantzi, K., Ananiadou, S., Mima, H.: Automatic recognition of multi-word terms. Int. J. of Digital Libraries 3(2), 117–132 (2000)
14. Peroni, S., Motta, E., d'Aquin, M.: Identifying Key Concepts in an Ontology, through the Integration of Cognitive Principles with Statistical and Topological Measures. In: Domingue, J., Anutariya, C. (eds.) ASWC 2008. LNCS, vol. 5367, pp. 242–256. Springer, Heidelberg (2008)
15. Gangemi, A., Catenacci, C., Ciaramita, M., Lehmann, J.: Modelling ontology evaluation and validation. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 140–154. Springer, Heidelberg (2006)
16. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering. Springer, London (2004)
17. Pinto, H.S., Tempich, C., Staab, S., Sure, Y.: DILIGENT: Towards a Fine-Grained Methodology for Distributed, Loosely-Controlled and Evolving Engineering of Ontologies. In: de Mántaras, R.L., Saitta, L. (eds.) 16th European Conf. on Artificial Intelligence, ECAI, pp. 393–397. IOS Press (2004)
18. Sure, Y., Staab, S., Studer, R.: On-To-Knowledge Methodology. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies. Series on Handbooks in Information Systems, pp. 117–132. Springer, Heidelberg (2003)
19. Suárez-Figueroa, M.C., Gómez-Pérez, A., Fernández-López, M.: The NeOn Methodology for Ontology Engineering. In: Suárez-Figueroa, M.C., Gómez-Pérez, A., Motta, E., Gangemi, A. (eds.) Ontology Engineering in a Networked World, pp. 9–34. Springer, Heidelberg (2012)
20. Suárez-Figueroa, M.C., Gómez-Pérez, A., Motta, E., Gangemi, A. (eds.): Ontology Engineering in a Networked World. Springer, Heidelberg (2012)
21. Ermolayev, V., Copylov, A., Keberle, N., Jentzsch, E., Matzke, W.-E.: Using Contexts in Ontology Structural Change Analysis. In: Ermolayev, V., Gomez-Perez, J.-M., Haase, P., Warren, P. (eds.) CIAO 2010, vol. 626, CEUR-WS (2010)
22. Booch, G., Jacobson, I., Rumbaugh, J.: OMG Unified Modeling Language Specification. Object Management Group (2000)
23. Motik, B., Patel-Schneider, P.F., Parisa, B. (eds.): OWL 2 Web Ontology Language, 2nd edn. Structural Specification and Functional-Style Syntax, `http://www.w3.org/TR/2012/REC-owl2-syntax-20121211/`
24. Schreiber, G.: OWL Restrictions, `http://www.cs.vu.nl/~guus/public/owl-restrictions/`
25. Ermolayev, V., Keberle, N., Matzke, W.-E.: An Upper-Level Ontological Model for Engineering Design Performance Domain. In: Li, Q., Spaccapietra, S., Yu, E., Olivé, A. (eds.) ER 2008. LNCS, vol. 5231, pp. 98–113. Springer, Heidelberg (2008)
26. Davidovsky, M., Ermolayev, V., Tolok, V.: Agent-Based Implementation for the Discovery of Structural Difference in OWL-DL Ontologies. In: Kop, C., et al. (eds.) UNISON 2012. LNBIP, vol. 137, pp. 87–95. Springer, Heidelberg (2013)
27. Alferov, E., Ermolayev, V.: Extracting Knowledge Tokens from Text Streams. In: Ermolayev, V., et al. (eds.) ICTERI 2013, vol. 1000, pp. 108–116. CEUR-WS (2013)

Evaluation of the Ontology Instance Migration Methodology and Solution in Industrial Settings

Maxim Davidovsky[1], Vadim Ermolayev[1], and Vyacheslav Tolok[2]

[1] Department of IT, Zaporozhye National University,
66 Zhukovskogo st., 69600 Zaporozhye, Ukraine
[2] Department of Mathematical Modeling, Zaporozhye National University,
66 Zhukovskogo st., 69600 Zaporozhye, Ukraine
m.davidovsky@gmail.com, vadim@ermolayev.com,
vyacheslav-tolok@yandex.ru

Abstract. The recent growth in maturity of semantic technologies has led to a substantially wider adoption of semantics enabled solutions in industries. These solutions typically include knowledge specifications in the form of different ontologies. Semantic heterogeneity can be resolved by applying ontology alignment methods to these ontologies. Therefore the development of the industrial strength software tools for solving different ontology alignment problems becomes important in the research and development. One of the practically important problems in the pool is the ontology instance migration problem. The paper presents the results of the evaluation of our solution of the ontology instance migration problem in industrial settings. The solution follows an iterative incremental approach involving a knowledge engineer as a process supervisor. Technically, the problem is solved by (i) automatically computing the structural difference between the schemas of ontologies; (ii) automatically generating the transformation rules for the individual assertions; (iii) automatically transforming the assertions; and (iv) manually controlling the completeness of those transformations and resolving problem cases. The solution has been evaluated by applying it to the ontologies in (i) libraries; and (ii) construction industry.

Keywords: Ontology alignment, industrial application, ontology instance migration, evaluation experiment.

1 Introduction

Ontologies are widely adopted today in the academic world and increasingly attract the attention of industrial researchers and practitioners in information technology and knowledge-based system development and applications. Many publications (e.g. [1], [2], and [3]) argue that ontologies constitute the substance of the advanced technologies for solving the problems of interoperability, communication, and cooperation between different applications within the common environment. Indeed, ontologies conceptualize semantics of the domains within a discourse that are common for interoperating systems. Thus, ontologies serve as a bridge for "understanding" between the systems or their parts. For example, in [1] the authors analyze interoperability problems faced by software solutions in Computer Aided Design (CAD), Architecture Engineering and Construction (AEC), and Geographic Information Systems (GIS). Subject to correct interaction, integration of heterogeneous systems into a single holonic system may significantly improve the efficiency of an enterprise in comparison with the operation of disparate systems. Despite that, application of ontologies in industry still faces several problems.

The first group of problems concerns the inertia that is typical for the process of application of advanced technologies in industry, e. g. [4]. This paper reflects the views of the practitioners who have met incomprehension and opposition in trying to solve customer problems using ontologies. It should be mentioned that in most such cases it is not just a "luddite protest" by domain stakeholders but rather the lack of understanding of how ontologies can help solve their problems. For many industrial customers an ontology seems to be an overcomplicated theoretical or philosophical thing suitable only for scientific or research purposes but not for business Information Systems (IS). These problems are attempted to be resolved through establishing a closer contact with the domain knowledge stakeholders [5] and their more active involvement in the development of ontologies. Another complementary and important activity is lowering the effort for developing ontologies which could be done via providing a tool support for domain experts taking part in ontology development.

Another important stratum of problems in the application of ontologies in industry is related to the re-use of existing large industrial knowledge bases, collections, or ontologies and the exploitation of those knowledge assets within large enterprise Information Systems (EIS). Enterprise Resource / Material Requirements Planning, Computer-Aided Design / Manufacturing / Engineering, or Customer Relationship Management systems are most often referenced in this context. Obviously it is obligatory to provide stable interoperation of ISs in industrial settings to prevent substantial errors in maintenance, production, and sales. However it has to be stated that the use of ontologies per se doesn't completely solve interoperability issues as it essentially raises heterogeneity problems to a higher (semantic) level [6]. Even if ontologies represent semantics of the same domain of discourse they may reflect different viewpoints or interpretations of the domain. The statement is a fortiori correct in the case of different though semantically overlapping domains. So, the methods for aligning ontologies need to be provided to understand and explicitly specify semantic correlations between these different conceptualisations. The process of building such an alignment is denoted as ontology matching. Industrial ontologies as a rule contain large quantity of individual entities (or concept instances). Hence, an important and typical sub-problem of ontology alignment in industrial settings is instance migration that is the process of transferring instances between aligned ontologies. Thus, providing of tool support for the resolution of heterogeneity problem must be brought to the fore as in large industrial systems ontology matching and related tasks cannot be performed manually due to the number of ontological entities.

Another important aspect in the exploitation of ontologies in industrial settings is that industrial ISs are usually distributed. In such settings it is rational to perform ontology matching and instance migration in a distributed way, for example by using intelligent software agents. Use of an agent paradigm allows the adjustment of ontology management infrastructure to the entire structure of distributed knowledge repositories. Knowledge used within ISs evolves and this requires a high level of dynamics from knowledge management apparatus. To the authors' opinion, mobility and flexibility of multi-agent systems more completely meets the requirements of distributed industrial ISs.

The paper presents the continuation of our work on the implementation of the iterative interactive ontology instance migration methodology and solution involving intelligent software agents. We started our research with the development of ontology instance migration engine [7]. The engine carried out migration using manually coded instance transformation rules. In [8] we described the proof-test evaluation of the engine with small- and moderate-size ontologies. In these test series we focused on the evaluation of the quality of instance migration applied to the ontologies which allowed checking instance migration results manually. In [7] we described performance testing which allowed us to judge about the applicability of our approach to large industrial ontologies. For that we used the ontologies with a large number of instances which required primitive transformations. In [9] we provided the implementation details of the ontology matcher used for discovering mappings and structural differences between the ontologies involved in the migration process. The matcher complemented the instance migration engine with the means for automatic transformation rules generation. Thus, at this stage we obtained a comprehensive instance migration solution. It has still to be checked if this solution is suitable for industrial applications. In this paper we present the evaluation of the entire process of ontology instance migration in industrial settings including mapping discovery and determination of structural differences together with instance transformation rules generation and immediate migration of ontology instances instead of evaluating of all these tasks separately. The original contribution of this paper is in presenting the details of the evaluation of the mapping discovery step and reviewing the typical problem cases and the strengths and weaknesses of different mapping discovery methods. Compared to our previous work, we evaluate our solution on real industrial ontologies not only for scalability but for the quality of results and using a broader suit of metrics. According to our evaluation methodology we evaluated the obtained results with the involvement of domain experts twice for each iteration: (i) after the mapping discovery step the experts analyzed the quality of the obtained mappings whereupon we controlled the correctness of the generated instance transformation rules; (ii) after the population of target ontology with the appropriately transformed instances we analyzed the quality of migration in the terms of completeness and correctness of the target ontology ABox.

The paper is organized as follows. In section 2 we provide a classification of industrial applications of ontology alignment, the types of problems, and describe some typical use cases. Based on this classification, we analyze potential industrial requirements to ontology alignment solutions. We determine the specificity of real-world industrial ontologies and data, and correlations between these specifics and ontology alignment methods and implementations. Section 3 outlines the classification of ontology alignment problems and states the problem of ontology instance migration as one of the ontology alignment problems. Section 4 offers the overview of our software solution for the ontology instance migration problem. In section 5 we present the experimental set-up and describe the test ontologies that are used in our evaluation experiments. Section 6 discusses our evaluation results. In particular it analyses the quality of instance migration. The paper is concluded with the summary of the presented contributions and our plans for the future work.

2 Related Work, Applications, and Use Cases

Ontology alignment in broad and ontology instance migration as a specific sub-problem are important in ontology engineering and management lifecycle, especially for evolving knowledge representations of sizes at industrial scale. For example, in the process of the development of an ontology a knowledge engineer can use existing ontologies – either another domain ontologies that have semantically relative domains or upper ontologies in order to derive domain specific concepts from more general concepts [10]. This is the case for ontology reuse, integration and merging. Thus, we can list the following typical tasks requiring the use of ontology instance migration techniques within ontology engineering and management activities ([6], [11]):

- Ontology reuse
- Knowledge extraction and reasoning
- Ontology fusion/integration/merging
- Ontology editing and import
- Ontology evolution and versioning
- Knowledge sharing

We omit the detailed analysis of ontology alignment use cases in ontology management tasks since they are beyond the scope of this paper and focus on industrial applications. Surveys of ontology alignment for a wide range of applications can be found in [6], [11], [12]. Applications of agent-based ontology alignment and respective requirements are analyzed in [13], [14].

Further, we consider the specifics of applying instance migration solution in industries. As a result we analyze the requirements to ontology TBox matching (phase (i) of our methodology) and instance migration (phase (ii) of our methodology) that condition this specificity. We outline the following industrial application categories that require ontology instance migration solution.

1. Industrial knowledge-driven simulation models. Simulation models are widely used in industry ([15], [16], [17]). They used for optimization of complex manufacturing process parameters and allow reducing the time of preproduction and modernization costs according to varying requirements. Consequently, simulation significantly increases the competitiveness of industrial enterprises, making the results of production closer to the customer needs and saves material resources.

The complexity level of modern simulation systems requires the use of knowledge-based models. This knowledge may be related to various branches of science, engineering disciplines, can contain different models satisfying different demands. This requires the use of ontologies and related activities such as ontology merging and alignment.

2. Industrial information systems in the context of Semantic Web and eCommerce. eCommerce is a type of industry where buying and selling of a product or service is conducted over electronic systems such as the Internet and other computer networks. In order to perform such an exchange of business information, this information must contain product (or service) descriptions such as electronic catalog along with information about the vendor, the manufacturer etc. As a rule such information is presented in the form of product or service ontology [18]. Good examples of such ontologies are [19], [20], [21]. When a business process involves more than one party or in a case of using more than one source respective ontologies obviously have to be aligned. This situation is also typical for The Semantic Web where ontologies along with intelligent software agents are the main pillars [22].

3. Integration and interoperability of heterogeneous enterprise information systems. Today information ecosystem of a modern enterprise as a rule contains numbers of applications from different vendors and used for different purposes. In order to effectively use these heterogeneous applications together with distributed data and knowledge repositories they must be integrated into a single system. Likewise implementation and deployment of new software solutions must be reconciled and integrated with legacy software systems. These reasons make integration of data, information and knowledge as well as heterogeneous applications within an EIS or between ISs of different, but related enterprises a very important task. Here ontologies may be used not only as domain knowledge representation models, but also as mediators for integration of heterogeneous applications. Enterprise integration attracts substantial interest of research community and a number of solutions are proposed (e.g., [23], [24]).

4. Knowledge sharing and migration between enterprise information systems. Interaction and cooperation of modern enterprises often implies knowledge sharing and migration. In such a way enterprise may enrich and harmonize their knowledge assets. In this case knowledge models obviously must be reconciled and aligned. This issue is not widely addressed in literature (but some early efforts, e.g. [25], are described) as it usually requires some (combination of) typical ontology management activities (such as ontology evolution and knowledge sharing – please see some details above).

Enterprise ISs regardless of particular category they belong to are characterized by a wide variety of ontologies describing particular domains or representing database schemes. Industrial ontologies can be both simple and small (with a relatively small number of entities and having a simple structure – usually taxonomy [26], [27]), and large (comprising hundreds of thousands of entities, e.g. BauDataWeb – The European Building and Construction Materials Database for the Semantic Web, http://semantic.eurobau.com/). The only common observation perhaps is that industrial ISs as a rule operate on big data which imposes certain requirements on industrial ontologies. A number of techniques can be used to cope with matching large ontologies. All of them can be roughly divided into 3 types: reduction of search space, parallel matching, and self-tuning [28]. Reduction of the match search space is achieved by early pruning dissimilar element pairs and limiting the number of element comparisons [29], [30]. A similar approach is based on partitioning the ontologies into smaller modules or clusters. This approach is the most widely addressed by researches and a substantial number of methods are proposed (e.g., [31], [32], [33], [34]). Among non-partitioning methods are [35], [36]. A good example of parallel matching technique is [37]. It should be mentioned that some publications directly address the problem of ontology alignment in networked enterprise systems. For example, in [38] proposed a common shared ontology-based framework for networked enterprise interoperability. However, it should be stated that there scarcely exists a common "one-fits-all" solution and such solutions strictly depends on specific context. In [38] the authors propose a common shared framework for networked enterprise interoperability based on the supply chain ontology. The proposed approach identifies concept alignments in order to propose a shared ontology as the reference information model for application interoperability in the context of information flows in supply chain interactions.

Each of the application categories sets up some requirements to specific alignment methods used within the category. Due to the wide variety of ontologies used in industry it is difficult to set up a detailed set of requirements for ontology matching methods. As described above these requirements may substantially vary depending on ontology size and structure so we outline only the most general observations. We analyze the requirements for ontology alignment regardless to industrial application in [13] and [14].

Run-Time. 1st and 2nd categories assumed the matching process to be performed at run-time. In that case the maximum level of automation must be reached. In 3rd and 4th categories it is allowed to perform matching and relative activities previously and separately. This allows active involvement of experts to the matching process (for alignment validation, relevance verification, etc.).

Completeness. Completeness is of the most importance in the 1st and 3rd cases. It is important not to miss knowledge in these cases. At the same time, in the 2nd category the response time of method implementation to a system query is more critical as in that case matching is usually performed during runtime.

Relevance. In the 4th case, the relevance of knowledge is the most critical (particularly during migration from an older system to a newer one). Here it is first of all important to save actual knowledge, but some obsolete knowledge may be discarded.

The instance migration problem per se can be solved in several ways. Often, the instance migration is performed manually with the help of an ontology editor (e.g. Protégé[1] or NeOn Toolkit[2]). A knowledge engineer usually uses some matcher to discover TBox mappings and then tries to apply them manually by the editor. It

[1] `http://protege.stanford.edu/` - The Protégé Ontology Editor and Knowledge Acquisition System.

[2] `http://neon-toolkit.org` – The Neon Ontology Engineering Environment.

should be mentioned that a mapping representation is not always human-friendly and, hence, some ontology schema difference visualizer (e.g., [39]) can be very useful in this case. Obviously, manual migration of ontology instances is hardly applicable in industrial settings as industrial ontologies usually contain a large number of ontological entities.

The process of ontology instance migration can be sufficiently simplified by using some instance matching techniques[3] [6], [40], [41]. Instance matching allows to determine similarity between an individual migrating from the source ontology and the set of individuals already existent in the target one. Thus, instance matching is a crucial task to avoid duplication of assertions and perform correct migration. The issue of instance matching for ontology population is addressed for example in [42], where the authors analyze the ontology population problem in the context of ontology evolution. The authors propose an algorithm and a solution for matching ontology instances represented in the form of OWL ABoxes. It should be noted that in most cases the instance-level matching techniques doubtlessly should be combined with the concept-level ones in order to form a comprehensive instance migration solution.

Another useful approach is based on the use of an ontology reasoner (a good example of such an approach is [43]). Execution of ontology instance migration by a reasoner usually implies population of target ontology based on a set of logical axioms (or rules) which represent required transformations over the source individuals. However, in general such an approach scarcely can be enough automated as it requires the availability of off-the-shelf rules in order to drive the migration process. So, it has to be combined with some TBox and ABox matching approaches. Moreover, reasoner-based approaches usually require ontology merging (as the majority of reasoners operate only on single ontology) that is a separate problem which can bring more difficulties to the solution.

Alternative approaches can be based on the use of query engine. The engine allows executing special queries for extracting the required set of individuals from the source ontology, transforming them according to the predetermined schema diff and populating the target ontology ABox. The queries are most often expressed in the SPARQL language[4]. It should be noted that this approach usually requires additional reasoning because SPARQL does not support RDFs and OWL inference. Likewise reasoner-based approaches, query-based ones also should be combined with matching techniques in order to discover differences between the source and target ontologies.

In large industrial knowledge bases assertions are often persisted and maintained with the use of specific RDF triple stores (such as Sesame[5] or OWLIM[6]). In such cases the instance migration task can be partially solved by means of the used stores. An overview of industrial triple stores can be found at

[3] `http://www.instancematching.org/` - Instance Matching

[4] `http://www.w3.org/TR/rdf-sparql-query/` - SPARQL Query Language for RDF

[5] `http://www.openrdf.org/` - An open-source framework for querying and analyzing RDF data.

[6] `http://www.ontotext.com/owlim` - A family of semantic repositories (RDF database management systems)

http://www.w3.org/wiki/LargeTripleStores. An approach and implementation for offering quality management during migrations of RDF instance data (RDF Instance Migration Quality Assistant) in the context of ontology evolution is presented in [44]. The tool allows identifying the set of instances potentially liable to migration, produce recommendations for respective instance descriptions and provide means for updating the target ABox after the curator of the RDF repository accepts or rejects them.

In contrast to the approaches described above, we use the integrated approach that allows obtaining ontology instance migration rules automatically. Generated rules are immediately ready for use which allows performing the routine steps of ontology instance migration workflow (Fig. 1) in a fully automated way and substantially reduce the required effort. However, if the knowledge engineer or domain expert considers the correctness of the automatically generated rules insufficient, our solution provides a convenient tool with a friendly graphical user interface which allows making the necessary changes in the rules manually. This interactivity makes the system more flexible, facilitates to obtain substantially better quality of results, and therefore increases the overall efficacy of the solution.

3 Ontology Instance Migration as an Alignment Problem

This section presents the formal problem statement and classification of ontology alignment problems. Further, it puts the ontology instance migration problem into the context of ontology alignment pool of problems. Finally, it discusses the problem statement for ontology instance migration in more detail.

3.1 Ontology Alignment

Following Euzenat and Shvaiko [6], an **ontology** is formally denoted as a tuple $O = \langle C, P, I, T, V, \leq, \perp, \in, = \rangle$ where C is the set of *concepts* (or *classes*); P is the set of *properties* (*object* and *datatype properties*); I is the set of *individuals*(or *instances*); T is the set of *datatypes*; V is the set of *values*; $\leq$ is a *reflexive*, *anti-symmetric* and *transitive relation* on $(C \times C) \cup (P \times P) \cup (T \times T)$ called *specialization*, that form partial orders on C and P called *concept hierarchy* and *property hierarchy* respectively; $\perp$ is an *irreflexive* and *symmetric relation* on $(C \times C) \cup (P \times P) \cup (T \times T)$ called *exclusion*; $\in$ is a relation over $(I \times C) \cup (V \times P)$ called *instantiation*; $=$ is a relation over $I \times P \times (I \cup V)$ called *assignment*; (the sets C, P, I, T, V are pairwise disjoint). It is also assumed ([45]), that an ontology O comprises its schema S and the assertional part A :

$$O = \langle S, A \rangle ; S = \langle C, P, T \rangle ; A = \langle I, V \rangle \qquad (1)$$

Ontology schema is also referred to as a **terminological component** (TBox). It contains the statements describing the concepts of O, the properties of those concepts, and the axioms over the schema constituents. The **set of individuals**, also referred to as an **assertional component** (ABox), is the set of the ground statements about the individuals and their attribution to the schema – i.e. where these individuals belong.

Ontology matching is denoted as a process of discovering the correspondences (or *mappings*) between the elements of different ontologies. A *mapping* (or a *mapping rule* [6]) is a tuple $m = \langle e, e', \Re, n \rangle$, where: e, e' are the elements belonging to C, R, I, T, V of the respective ontologies O and O'; $\Re = \{\subset, \subseteq, \equiv, \supset, \supseteq\}$ is a set of relations; and n is a confidence value (typically in the range of $[0,1]$).

Finally, **ontology alignment** is denoted as the result of applying the discovered set of mapping rules to the respective ontologies. A generic ontology matching process and ontology alignment are described and pictured in more detail at http://isrg.kit.znu.edu.ua/a-boa/index.php/Basic_Definitions_and_Generic_Problem_Statement.

3.2 Classification of Ontology Alignment Problems

Following [14] and based on the features of participating ontologies and the span of e, e' across C, P, I, T, V-s of O and O' a classification of the problems of finding ontology alignments could be outlined and formally stated. Graphical interpretation of some of these problems is described in more detail at http://isrg.kit.znu.edu.ua/a-boa/index.php/. The dimensions along which the problems are classified are:

- Complete (C), structural (S), or assertional (A) alignment
- Static (S) versus dynamic (D) aligned ontologies
- Bi-directional (B) versus uni-directional (U) alignment
- Fully distributed (D) settings versus the presence of a central (C) referee ontology

A generic ontology alignment problem may therefore be classified as a complete static bi-directional alignment using central referee ontology (CSBC). Ontology instance migration described in detail below could be classified as an assertional, static, uni-directional, distributed (ASUD) ontology alignment problem.

Yet another important feature for classifying ontology alignment processes is the presence of iterations for the refinement of alignments. All the processes discussed above are one-shot. However, the resulting alignments may appear to be of insufficient quality after their evaluation. Iterative ontology alignment processes aim at improving this shortcoming by incorporating the evaluation step and the refinement cycle in the process – please refer to (http://isrg.kit.znu.edu.ua/a-boa/index.php/ Classification_of_Ontology_Alignment_Problems) for a graphical illustration. Iterations in ontology instance migration process are discussed in more detail below.

3.3 Ontology Instance Migration Problem

One of the practically important ontology alignment problems, especially in fully distributed and dynamic settings, is the problem of transferring the individuals of one (source) ontology to the empty assertional part of the other (target) ontology [7].

Let us consider two arbitrary ontologies $O^s = (S^s, A^s)$ and $O^t = (S^t, A^t)$ conceptualizing the semantics of the same universe of discourse U – for example O^s and O^t are the two ontologies describing the same subject domain. U could be regarded as a collection of ground facts: $U = \{f\}$. Essentially, O^s and O^t are the interpretations [46] of *U*. These ontologies would be considered identical if and only if:

$$\forall f \in U \operatorname{int}_{I^s}(f) \equiv \operatorname{int}_{I^t}(f), \tag{2}$$

where $\operatorname{int}_I(f)$ is the interpretation of the fact *f* by the individuals from *I* of ontology *O*.

Consequently, an abstract metric of interpretation difference $idiff(U, O^s, O^t)$ could be introduced. The value of *idiff* will be equal to zero for identical ontologies and will increase monotonically to one with the increase of the number of $f \in U$ such that $\neg(\operatorname{int}_{I_s}(f) \equiv \operatorname{int}_{I_t}(f))$. Hence, $idiff = 1$ iff $\forall f \in U \neg(\operatorname{int}_{I^s}(f) \equiv \operatorname{int}_{I^t}(f))$. $(1 - idiff)$ may further be interpreted [14] as balanced *F*-measure.

Ontologies O^s and O^t are structurally different if their schemas differ: $S^s \neq S^t$. This structural difference may be presented as a transformation $\mathrm{T}: S^s \to S^t$. If a finer grained look at an ontology schema is taken, one may consider *S* comprising the following interrelated constituents:

$$S = \{SC, SO, SD, SA\}, \tag{3}$$

where: SC is the set of statements describing concepts; SO is the set of statements describing object properties; SD is the set of statements describing datatype properties; and SA is the set of axioms specifying constraints over SC, SO, and SD. One may notice that these constituents correspond to the types of the schema specification statements of an ontology representation language *L* which is used for specifying O^s and O^t. The transformation *T* may be sought in the form of nested transformation rules over the constituents of S^S resulting in the corresponding constituents of S^t.

Let us assume now that, given two structurally different ontologies O^s and O^t, the ABox of O^s contains individuals ($I^s \neq \varnothing$), while the ABox of O^t is empty ($I^t = \varnothing$). The problem of minimizing $idiff(U, O^s, O^t)$ by: (i) taking the individuals from I^s; (ii) transforming them correspondingly to the structural difference between O^s and O^t using T; and (iii) adding them to I^t – is denoted as **ontology instance migration problem**.

Theoretically ontology instance migration problem can be solved in one shot. In practice however each of the sub-tasks (ii-iii) may result in the loss of assertions [7]. Therefore an iterative refinement of the solution could yield results with a lower resulting *idiff* value. Hence, the problem has to be solved using an iterative ontology alignment process. Essentially, an iterative solution of ontology instance migration problem develops a sequence of O^s states $O^s_{st_i}$ in a way to minimize the $idiff(U, O^s, O^t)$ in a way that:

$$idiff(U, O^s_{st_i}, O^t) < idiff(U, O^s_{st_j}, O^t) \to i > j, \tag{4}$$

where: $O^s_{st_i}$ is O^s in the state after accomplishing iteration *i*; *i*, *j* are iteration numbers.

3.4 Instance Transformation Patterns and Rules

For correctly and completely migrating instances between a source and target ontology the transformation process has to be explicitly and formally specified. The specification implies an (implicit) declaration of the set of transferable individuals that is achieved by the explicit selection of the set of concepts and properties which instances have to be migrated. Also it is necessary to specify the set of required transformations over the migrating individuals. Having done that, we obtain the set of transformation rules for instance migration.

We have defined all possible kinds of (structural) transformations that are further applied to the sets of individuals I^s and I^t in [7]. These structural transformations are defined as the mappings between the constituents of the ontology schemas S^s and S^t :

$$TX : SX^s \to SX^t, \tag{5}$$

where X is the placeholder for *C*, *O*, *D*, or *A*, and are specified as the sets of rules, for example: $TC = \{tC\}$. These rules may have a nested structure. Concept transformations may nest the rules from TO , TD , or TA ; property transformations may nest the rules from TA . The transformation rules for the individuals are derived from the structural transformations $T = \{TC, TO, TD, TA\}$ and applied to the set of assertions:

$$TrX : IX^s \to IX^t, \tag{6}$$

where X is the placeholder for *C*, *O*, *D*, or *A*. *Tr*-s as the derivatives of *T*-s may also have a nested structure.

Complex transformation rules are assembled using atomic transformation specifications that represent indivisible transformation operations. However, not all theoretically possible atomic transformations make sense for ontology instance migration. For example, an atomic operation of concept deletion is irrelevant because no individuals belonging to the deleted concept will be migrated. We have defined instance transformation patterns for those atomic operations that make sense. A transformation pattern is denoted as a mapping $Tp : L \to L$ where L is the used ontology representation language[7]. Similarly to transformation rules, transformation patterns belong to different categories that apply to different kinds of statements in *L*:

- $TpC = \{tpC\} : L \to L$ are the patterns for concept instance transformations
- $TpO = \{tpO\} : L \to L$ are the patterns for object property instance transformations
- $TpD = \{tpD\} : L \to L$ are the patterns for datatype property instance transformations
- $TpA = \{tpA\} : L \to L$ are the patterns for the specification of the individual constraints (axioms)

[7] The solution we have implemented uses the DL-restricted subset of OWL [47].

The transformation patterns are coded as XMLSchema (www.w3.org/XML/Schema) using the editor developed as a part of our solution. A complete specification for the atomic transformation patterns, including also the problems related to the use of those patterns, is given in [7]. Here we provide one of them, for the addition of a new object property, as an example:

```
<concept concept_name="name1">
    <addRelation domain="name1" range="name2">
     relationName</addRelation>
    …
</concept>
```

4 Ontology Instance Migration Methodology and Software

Our software solution (see also [9] for more details) implements a (semi-) automated iterative process of ontology mapping discovery and instance migration with possible human involvement for checking the correctness or completeness of the results, and setting-up the process. The workflow consists of the two phases: (i) the discovery of the structural difference and mappings between the TBoxes of the source and target ontologies; and (ii) ontology instance transformation and migration between the ABoxes of the source and target ontologies according to the discovered differences. Figure 1 details those phases in six steps and shows human involvement in step (II) of phase (i) and steps (V) and (VI) of phase (ii). The rest of the workflow is performed automatically by our agent-based software solution.

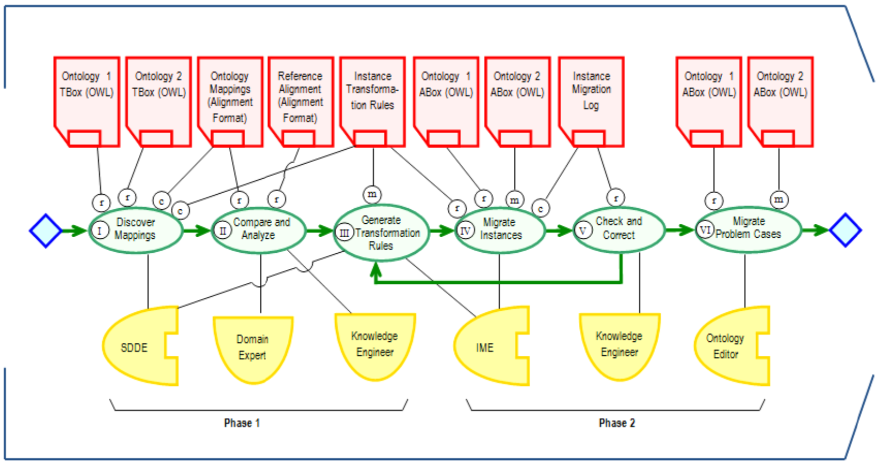


Fig. 1. The methodology for iterative ontology instance migration specified in ISO/IEC 24744 notation for describing methodologies [48]

Phase (i) comprises:

- The automated step (I) for discovering the structural difference in the form of mappings
- The interactive step (II) for checking the validity of the discovered difference by a human knowledge engineer with a help of a domain expert if required
- The automated step (III) for generating the instance transformation rules of the mappings discovered at step (I)

Step (I) is essentially the process of ontology TBox matching with the difference that it results not only in ontology alignment but also produces the set of ontology instance transformation rules as its output. These transformation rules are further used in phase (ii) for ontology instance migration. Our software solution for TBox matching is based on the implementation of meaning negotiation between intelligent software agents. We call this agent-based solution the Structural Difference Discovery Engine (SDDE) [14]. The matching process embodies the meaning negotiation strategy [49] and is described in details in [9]. The negotiation strategy implies aligning ontologies by structural contexts [50]. Those structural contexts are essentially the sub-graphs of the TBoxes of the source and target ontologies. Each negotiation encounter is arranged around a pair of those sub-graphs – one representing the source and the other representing the target ontology. Negotiation is conducted in iterations with the goal to reduce the semantic distance between the contexts. The semantic distance is measured using the combination of linguistic, contextual or feature, and instance-based similarity metrics as described in detail in [49, 9]. Agents iteratively communicate the arguments in the form of propositional substitutions (Type Theory [51] statements). Those propositional substitutions are checked by the counterpart to find out if they help reduce the semantic distance between the two structural contexts in the negotiation set. If so, the arguments are accepted as valid and the successful substitutions are further regarded as appropriate mappings. Negotiation is accomplished when the semantic distance reaches the commonly accepted threshold or the parties exhaust their arguments.

At step (II) the graphical interface, based on the extended UML notation [52] for visualizing structural difference, is provided to human experts who decide the validity of the discovered difference between the ontology TBoxes. They also have the function to edit the difference in the parts they consider requiring correction. The mappings for the corrected fragments are automatically adjusted.

At step (III) ontology instance transformation rules are automatically generated based on the mappings coming from step (II) in the ontology Alignment format [53]. The notation for instance transformation rules is presented in [7].

Phase (ii) comprises:

- The automated step (IV) for migrating the instances from the ABox of the source ontology to the ABox of the target ontology
- The interactive step (V) for checking the completeness of the instance migration by a knowledge engineer
- The interactive step (VI) for the manual migration of those instances that were marked as problem cases and not migrated at step (IV)

Steps (IV) and (V) are done iteratively. The refinement of the transformation rules for the next iteration is done at step (V). Step (VI) is performed once at the very end of the workflow to do the migration for the individual problem cases in which the refinement of the transformation rules cannot be done at step (V) due to the limitations of the transformation rule notation or practical considerations.

At step (IV) software agents in our solution use the Instance Migration Engine (IME) to transfer the instances from the source to the target ontology. The instances are transformed using the transformation rules generated at step (III). The information about the failure in migrating an instance or a group of instances is recorded in the Instance Migration Log.

Instance Migration Log is used as the input information at step (V). At this step the knowledge engineer analyses the Log and may decide that the transformation rules need to be refined in order to ensure the automated transfer in the detected problem cases in the next iteration. If this decision is taken, the corresponding transformation rules are manually edited using the graphical user interface offered at this step. If the rules are refined steps (IV) and (V) are repeated. Otherwise the knowledge engineer may decide that the remaining instances are either not required in the target ABox or may be migrated manually at step (VI). If so the manual transfer of the required assertions is done using an ontology editor.

5 Set-Up of Evaluation Experiments

To evaluate our methodology (Fig. 1) and agent-based software solution (Section 4) for ontology instance migration we chose two different ontology test sets. The first one consists of two ontologies from Ontology Alignment Evaluation Initiative (OAEI[8]) benchmark test suites (data sets) of 2012 campaign [54]. The benchmark test library consists of data sets that are built from reference ontologies of different sizes and from different domains. We chose two ontologies relating to Bibliographic references domain.

The first ontology (further referred to as the Biblio ontology) can be thought of as a common classification of publications among scholars based on the mean of publications[9] and is reminiscent to BibTeX[10]. The versions of this ontology have been the main reference ontologies since the beginning of OAEI campaigns. It comprises a number of circular relations and utilizes some external resources: e.g., FOAF[11] and iCalendar[12] for expressing non bibliographic information such as the People, Organizations, and Events. The structural characteristics of the Biblio ontology are presented in Table 1. The second ontology, which is supposed to be aligned with the

[8] http://oaei.ontologymatching.org/ – Ontology Alignment Evaluation Initiative.

[9] http://oaei.ontologymatching.org/2012/benchmarks/index.html

[10] http://www.bibtex.org/ – A tool and a file format to describe and process lists of references.

[11] http://xmlns.com/foaf/0.1/ – A machine-readable ontology describing persons, their activities and their relations to other people and objects.

[12] http://www.w3.org/2002/12/cal/ – An effort to apply the Resource Description Framework (RDF) to iCalendar data.

Biblio one, is the INRIA bibliographic ontology. This is an actual ontology created from the BibTeX in OWL ontology and OAEI Bibliographic XML DTD. It contains a number of BibTeX entries found on the web and transformed into the list of RDF items. The structural characteristics of the INRIA ontology are also presented in Table 1.

Table 1. Structural characteristics of the Biblio and INRIA ontologies

Ontology	Total number of axioms	Number of logical axioms	Number of classes	Number of object properties	Number of datatype properties	Number of named individuals	Number of anonymous individuals
Biblio	911	562	37	26	46	57	20
INRIA	474	202	41	40	11	1	22

In addition to the test ontologies OAEI contest provides the reference alignment to which the results of ontology matching can be compared. This alignment follows the Alignment format [53] that makes the results of different matchers be comparable and reproducible. Actually, the reference alignment contains only one-to-one direct mappings that is not in general a compete alignment. Hence, the resulting complete alignment can be obtained by supplementing the alignment provided by OAEI contest with non-trivial mappings determined by domain experts. It should be mentioned that we chose the Biblio and INRIA ontologies from the whole set of OAEI suite because they provide human-understandable entity names and labels. This allows domain experts determine more precise and complete mappings for the reference alignment according to our methodology (Fig 1).

The experiment with OAEI ontologies allows evaluating our solution on "good" ontologies (in a sense that they do not cause any collisions at run-time) and validating its applicability to real-world ontologies of small and moderate sizes. As the ontologies contain small sets of instances the experiment aims more at the evaluation of: our TBox matching solution (SDDE, step (I)); and verification of the generation of instance transformation rules. The intelligibility of the ontologies and clarity of the domain along with a relatively small size of the ontologies allow evaluating and analyzing the results fully and with a considerably small effort. Also, as suggested by the OAEI team, the ontologies can be regarded as different versions of the same ontology. Hence, the Biblio and INRIA ontologies fit perfectly for our evaluation experiment as ontology versioning is a typical case that often requires ontology instance migration solution [7].

The experiment consisted of six stages. Each class of mapping discovery method implemented in the SDDE can be weighted according to its importance [9]. Therefore, at each stage our goal was to investigate the correlation between the obtained alignment and the respective weights of the method classes. Thus, we can judge about the effectiveness of each method on concrete data that in turn allows us to choose the most appropriate weights in order to obtain the most correct results. At the first stage it is considered that all the methods are equally important and hence equal weights have been assigned to all of them. In the subsequent stages each class of methods have been considered as the most important (with a higher weight than the others) and the results of this weighting were analyzed.

The objective of the second experiment is to evaluate the applicability of our solution in industrial settings. Therefore, the second test set of ontologies has been chosen to contain larger set of assertions in order to assess the correctness of instance transformation rules generation as well as the completeness and quality of the instance migration result. Hence, we opted to experiment with the real industrial ontologies: freeClass[13] ontology for construction and building materials and services and eClassOWL[14] [20] – the web ontology for products and services. Both ontologies are based on the GoodRelations Web Vocabulary for E-Commerce[15] and used in the experiment as the test-set for ontology matching step. The dataset of the European building and construction materials market for the Semantic Web (BauDataWeb[16]) has been selected as the set of assertions for migration. Essentially, the BauDataWeb dataset represents the suite of individual sets described in terms of the freeClass and eClassOWL ontologies. The structural characteristics of the ontologies used in the second evaluation experiment are presented in Table 2.

Table 2. Structural characteristics of industrial ontologies used in the second experiment

	Total number of axioms	Number of logical axioms	Number of classes	Number of object properties	Number of data properties	Number of individuals
freeClass	78414	9622	5231	168	3	1335
eClassOWL	360243	117090	60662	4900	2453	4766
BauDataWeb	-	-	-	-	-	Over 60 million instances

The set-up for both evaluation experiments follows our ontology instance migration methodology presented in Fig. 1.

6 Results of Evaluation Experiments

In this section the results of our evaluation experiments are presented and analysed.

6.1 Experiment with OAEI Ontologies

As shown in Fig. 1, we conducted the experiment in a few iterations. The first iteration involves the automatic discovery of mappings and validation by domain experts. Also this phase implies automated instance transformation rules generation on the basis of the obtained mappings. Then, these rules are applied to the set of

[13] `http://www.freeclass.eu/` – The ontology for construction and building materials and services.

[14] `http://www.heppnetz.de/projects/eclassowl/` – The web ontology for products and services.

[15] `http://www.heppnetz.de/projects/goodrelations/` – The web vocabulary for e-commerce.

[16] `http://semantic.eurobau.com/` – BauDataWeb: the European Building and Construction Materials Database for the Semantic Web.

ontology instances; the instances are transformed according to the rules and reclassified. The process of ontology instance migration is traced to the appropriate log. Then, according to the methodology the results are analyzed.

Fig. 2 represents a fragment of the alignment obtained by SDDE on the first iteration. The figure shows fragments of class hierarchy trees of Biblio and INRIA ontologies (class hierarchies are initially taken from the Protégé ontology editor[1]).

For clarity Fig. 2 shows only those concepts that represent the most typical cases of correctly or incorrectly discovered mappings, or orphans. Further these sets of concepts are used to illustrate the work of various classes of methods implemented in the matcher. The concepts included in the mapping are connected by straight lines in Fig. 2. Those lines that are labeled with «m» show the mappings discovered by SDDE. The mappings included in the reference alignment are labeled with «r». Orphans that do not enter into any mapping are marked with a red border.

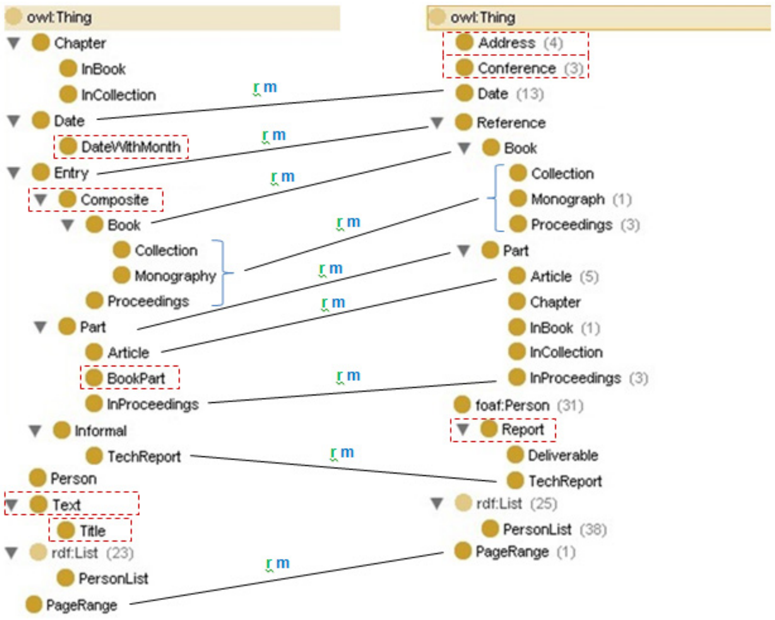


Fig. 2. Ontology mapping after the 1st iteration. Reference alignment and matcher determined mappings.

In the second iteration we involved the domain experts from the Scientific library and Publishing department of Zaporozhye National University. Experts (with the aid of the knowledge engineer) have edited the mappings discovered at the first iteration and also added some new ones. Some typical mappings obtained at the second iteration are shown in Fig. 3. Some excerpts from the results of the experiment with the OAEI ontologies are presented below in Tables 3, 4 and 5.

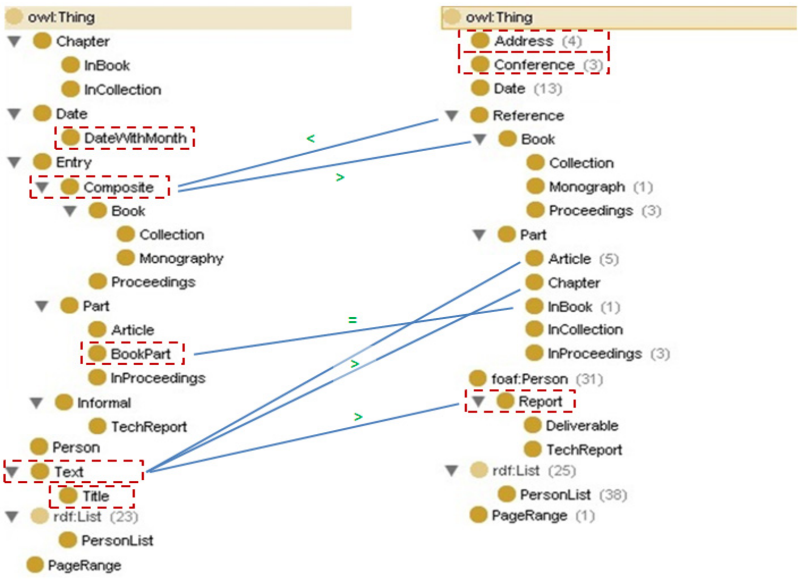


Fig. 3. Ontology mapping after the 2nd iteration. Domain expert mappings.

Table 3. Characteristic mappings conditioned by using string-based mapping discovery methods

Method/Metric			Weights	Mapped Entities		ABOA matcher results		Reference alignment			
				INRIA	Biblio	Relation Type	Integrated Metrics Value	Relation Type	Metrics Value	Provided by OAEI	Expert
String-based	Stage										
	1	+	~ 0.17	Proceedings	Proceedings	=	~ 0.8426	=	1.0	•	•
		–		Chapter	Chapter	=	~ 0.5406	≠	-	•	•
	2	+	0.5	Person	foaf:Person	=	~ 0.8789	=	-	-	•
		*		Entry	Reference	=	~ 0.2097	=	1.0	•	•
	3,4,5,6		0.125	Nonessential results[17]							

In the «Mapped entities» column of Tables 3–5 the typical examples of "good" and "bad" mappings for each class of methods are presented. We consider a mapping as "good" if it is *correct* and has high scores within the respective class of methods and relatively low in other classes. The results were compared to the reference alignment provided by the OAEI contest and to the alignment specified with the help of domain experts. Thus, we consider a mapping as *correct* if (and only if) its type coincide with respective mapping from OAEI reference alignment and domain expert alignment, and its score insignificantly (i.e. less than the predefined delta) differs from the respective value in the reference alignment. Empirically we have identified the delta

[17] Due to the low weight of the method we consider these results as non-essential.

equal to 0.2 because this value gave the highest number of matches between the matcher-defined mappings and the reference alignment. Good mappings are placed in the rows marked «+» in the tables.

Table 4. Characteristic mappings conditioned by using structure-based mapping discovery methods

Method/Metric				Weights	Mapped Entities		ABOA matcher results		Reference alignment			
					INRIA	Biblio	Relation Type	Integrated Metrics Value	Relation Type	Metrics value	Provided by OAEI	Expert
Structure-based	S	Stage	+	~ 0.17	Entry	Reference	=	~ 0.5096	=	1.0	•	•
		1	*		Book	Book	=	~ 0.4566	=	1.0	•	•
		3	+	0.5	Entry	Reference	=	~ 0.7920	=	1.0	•	•
			*		Book	Book	=	~ 0.4	=	1.0	•	•
		2,4,5,6		0.125	Nonessential results							
	OP	1	+	~ 0.17	Entry	Reference	=	~ 0.5096	=	1.0	•	•
			*		Book	Book	=	~ 0.4566	=	1.0	•	•
		4	+	0.5	Entry	Reference	=	~ 0.7919	=	1.0	•	•
			–		Tech-Report	Report	=	~ 0.5475	>	-	-	•
		2,3,5,6		0.125	Nonessential results							
	DP	1	+	~ 0.17	Page-Range	Page-Range	=	0.75	=	1.0	•	•
			–		DateWith-Month	Date	=	~ 0.3119	<	-	-	•
		5	+	0.5	Page-Range	Page-Range	=	~ 0.85	=	1.0	•	•
			–		DateWith-Month	Date	=	~ 0.3966	<	-	-	•
		2,3,4,6		0.125	Nonessential results							

Legend: **S** – subsumption relationship metrics, **O** – object properties metrics; **DP** – data properties metrics.

Table 5. Characteristic mappings conditioned by using extensional mapping discovery methods

Method/Metric			Weights	Mapped Entities		ABOA matcher results		Reference alignment			
				INRIA	Biblio	Relation type	Integrated Metrics Value	Relation Type	Metrics Value	Automated	Expert
Extensional	Stage	+	~ 0.17	List	List	=	~ 0.7863	=	1.0	•	•
	1	*		Date	Date	=	~ 0.6955	=	1.0	•	•
	6	+	0.5	List	List	=	~ 0.7773	=	1.0	•	•
		*		Date	Date	=	~ 0.4591	=	1.0	•	•
	2,3,4,5	Nonessential results									

There are 2 types of "bad" mappings in our experimental case. The first type is incorrectly defined mappings – those having no matches in the reference alignment and not confirmed by experts, but having relatively high (i.e., above the given threshold) integral values of the semantic similarity measures. Such mappings are marked «–». The second type is mappings of "missing knowledge" type – those not found by the matcher, but available in reference alignment and/or determined by domain experts. This kind of "bad" mappings is marked «*». Potentially, it may also happen that a mapping from a reference alignment contradicts to some mapping specified by domain experts. For example, some concepts are defined in the reference alignment as equivalent, but the experts regard those differently. In our experiment we didn't face any collision of this kind.

In Table 5 the mapping <Date ↔ Date> represents a case of *potentially missing knowledge*. At the sixth stage the mapping has a rather low metric value (~ 0.4591) as the sets of instances of respective entities substantially differ. However other methods produce high results for this mapping. Thus, in this case, extensional methods only reduce the value of integrated metric. This is well illustrated by comparing the results at this stage with the result of the first stage (equal weighted methods). At the first stage the mapping has similarity value circa 0.7863, and at the sixth – about 0.4591. Obviously, (when having a lesser threshold) there may be cases when such a mapping would be lost.

Potentially missing knowledge. This is the case when particular method weighting leads to possible discarding of correct mappings due to the low similarity values (e.g., see the mapping <Date ↔ Date> presented in Table 5). Evidently, discarding of such mappings lead to the loss of assertions during the migration. In fact, in our experiment, we do not lose such mappings. Though the metrics values were low (on the verge of the threshold), but still acceptable. However, having the same methods' weights, but more rigorous selection the knowledge will be lost.

Potentially redundant mappings. This case represents the situation when incorrect mappings are accepted. Such a situation may appear for example in the case of homonymy of different ontological entities. In such a case string-based methods give high similarity values and weighting these methods with high weights lead to generation of wrong mappings. In our experiment we had similar problems with datatype property based methods. For example on the first iteration (stage 5) the SDDE generated a trivial equivalence mapping <Date-With-Month ↔ Date> (see Table 4) with high enough similarity value (0.3966) as these concepts have similar datatype property sets and the value of datatype metrics is rather high. In fact, this value was on the verge of the threshold. So, with decrease of the threshold such maps could be accepted as correct, however these pairs are determined by domain experts as non-equivalent.

According to our methodology both problem cases (of potentially missing or redundant mappings) must be resolved by a knowledge engineer with possible involvement of the domain experts (see Fig. 1).

6.2 Experiment with BauDataWeb Dataset

The second test case doesn't contain any reference alignment. Hence, we had to determine reference mappings manually in order to objectively judge about the obtained results. For convenience both freeClass and eClassOWL ontologies may be divided into 2 parts. The first parts are actually the sets of entities directly inherited from the GoodRelation ontology (and also some concepts from other common-sense vocabularies). The schemes of these parts are almost identical and differ mainly by the sets of individual entities. These parts do not cause any problems during the determination of reference mappings as the entities mainly have human-understandable names and labels. We analyzed these sets and constructed a set of reference mappings (further referred to as Alignment 1). The second parts consist of internal entities that don't have understandable names (the names actually represent some identifiers composed of numbers and characters), but a number of them still have labels with descriptions. Due to the number of such entities we did not analyze the whole sets and arbitrary choose 20 entities that in our opinion have semantic correlations. Then we determined respective mappings for these entities (further referred to as Alignment 2). The characteristics of the both alignments are presented in Table 6 below (to be short, only the information about classes and properties is provided there).

Table 6. Parameters of the reference alignment for the experiment with the BauDataWeb dataset

	Number of mapped entities		
	Classes	Object properties	Datatype Properties
Alignment 1	53	55	53
Alignment 2	20	11	0

The experiment with the BauDataWeb dataset was performed in two stages. Within the first stage we constructed reference alignment (Alignment 1) and started the matching process using SDDE. Then we found those mappings that correspond to the Alignment 1 and compared them. Measured similarity values in this stage are very high (see Table 7 for the details) as these parts of the ontologies are almost identical. However, the data used in this stage represents a good case for validating the generated instance transformation rules and the instance migration quality. In this case all of the generated transformation rules are correct.

At the second stage we determined the Alignment 2 and tried to find the respective mappings within the alignment determined by the matcher. The results are presented below. Measure values for the second stage are low that is conditioned by relatively weak semantic similarity between the contexts that correspond to these parts of ontologies.

Table 7. Matching results (2nd experiment)

Experiment Stage	Measures		
	Precision	Recall	F-Measure
1	0.99999	0.99999	0.99999
2	0.69552	0.42384	0.52671

The migration results for stage 1 are presented in Table 8. It should be noticed that despite the fact that we had almost absolutely correct mappings generated by SDDE in the 1st iteration, we still had not very high recall of migrated instances since we missed some non-trivial mappings. After the 1st iteration instance transformation rules were refined by the knowledge engineer. Thus, after the 2nd iteration we had substantially higher quality measure values. However, as one can see from the Table 8, we still do not have absolutely correct and complete results due to the collisions that occurred during the migration of the transformed source instances into the target ontology ABox. Those collisions may arise for example when we add a new datatype property to an individual, but do not know the actual value of the property. In this case the respective value slot remains blank. More details on the types of possible collisions and incurred migration problems are provided by [7] and [8].

Table 8. The migration results for the 2nd experiment (stage 1)

Iteration	Measures		
	Precision	Recall	F-Measure
1	0,91667	0.73333	0.81481
2	0.99107	0.88095	0.93277

The migration results for stage 2 are presented in Table 9 below. It should be mentioned that the presented results are approximate as we could not evaluate the correctness of the migration fully due to the size of the dataset used in the 2nd experiment (the BauDataWeb dataset).

Table 9. The migration results for the 2nd experiment (stage 2)

Iteration	Measures		
	Precision	Recall	F-Measure
1	0,64286	0.60000	0.62069
2	0.83333	0.88235	0.85714

We carried out the migration of about several million instances and checked the target ontology for consistency after the migration. Though we cannot judge about the correctness of the reclassifying of all the migrated instances fully confidently, we can state that the ontology remains consistent and valid for further exploitation. As described above in Section 4, the transformations that violate consistency are not performed, but recorded into the Instance Migration Log). Migration quality was evaluated on 20 arbitrary chosen ontological contexts. It is characteristic that the F-measure value noticeably increased compared to the first iteration. It is so because a substantial number of typical problem cases were resolved by the knowledge engineer and domain expert intervention at step (V) of our workflow. For example, in the 2nd experiment string-based methods produce high values of similarity measures on labels. Labels can contain words that are common to many of them, however the corresponding ontology elements are not semantically similar. For instance, the labels "construction technology" and "pump technology" will provide for high similarity measured using a string-based metric, but obviously belong to the elements having different semantics. The problems of that sort were successfully resolved by the knowledge engineer between the iterations 1 and 2 of our experiment.

7 Summary and Future Work

The paper presents the evaluation experiments with our methodology for ontology instance migration using real-world industrial ontology use cases: one pair of ontologies from the Bibliography domain and the other – from the construction industry.

The pair of the bibliographic ontologies has been chosen as it is one of the benchmark ontologies of the OAEI. Though the number of individual assertions in the Bibliographic ontologies was small compared to real industrial scale, we used this case for evaluating in our 1st experiment primarily the phase (i) of our methodology and the SDDE as the software solution for TBox matching and generating instance transformation rules. The experiment was conducted with the involvement of domain experts for the comprehensive assessment of mapping discovery quality and completeness and correctness of the migration results. In the paper we provided a detailed analysis of the typical problem cases and the strengths and weaknesses of different mapping discovery methods coupled with different types of semantic similarity metrics. The experiment proved the effectiveness of our iterative approach with active user involvement for the refinement of automatically generated instance transformation rules as the overall accuracy was observed as substantially increasing after each iteration. The experiment showed acceptable results that allow positively judging about applicability of our solution in industrial settings.

For our second experiment the pair of ontologies having a real industrial size has been chosen. Our focus in the second experiment was on the phase (ii) of our methodology. One complication in assessing the quality of instance migration we had exactly because of the size of the processed ABox taken from the BauDataWeb dataset and the absence of the reference alignment. The complication was resolved by selecting a small part of the ontology TBoxes with human understandable names of the ontology elements and creating the reference alignment manually. We found out that for this part of the ontologies the migration worked very well. For the rest of the ontologies the involved of a human expert at step (V) of our workflow helped substantially to increase the quality of migration.

Hence the experimental results we obtained prove that our ontology instance migration solution is robust enough to be transferred to industries.

The results also suggest some directions for future work. The experiment with large ontologies (BauDataWeb dataset) shows that the ontology instance migration engine allows migrating about several million instances using conventional PC. Hence, some techniques to overcome this upper limit are needed. In future we plan to conduct series of experiments with OWL sublanguages[18] and OWL 2 profiles[19].

[18] http://www.w3.org/TR/2004/REC-owl-features-20040210/#s1.3
[19] http://www.w3.org/TR/owl2-profiles/

Another important direction for future research is testing of our approach on ontologies with different typical structures such as tree-type structure, network structure, graphs with high and low vertex degrees, etc.

Acknowledgments. The authors are thankful to the colleagues who provided the industrial ontologies for our experiments – Univ. Prof. Dr. Martin Hepp and Dipl. Ing. Andreas Radinger from E-Business and Web Science Research Group of the Universität der Bundeswehr, München. The authors also acknowledge the assistance of our volunteer subject experts at ZNU Library and Publishing Department.

References

1. Bittner, T., Donnelly, M., Winter, S.: Ontology and Semantic Interoperability. In: Prosperi, D., Zlatanova, S. (eds.) Large-scale 3D Data Integration: Problems and Challenges. CRCPress, London (2005)
2. Yang, Q.Z., Zhang, Y.: Semantic Interoperability in Building Design: Methods and tools. Computer-Aided Design 38, 1099–1112 (2006)
3. Obrst, L.: Ontologies for Semantically Interoperable Systems. In: Proceedings of the Twelfth International Conference on Information and Knowledge Management, CIKM 2003, New Orleans, LA, USA, November 3-8 (2003)
4. Malzahn, D.: Industrial Application of Ontologies. In: The Third International Conference on Information, Process, and Knowledge Management, eKNOW 2011 (2011)
5. Tatarintseva, O., Ermolayev, V.: Refining an Ontology by Learning Stakeholder Votes from their Texts. In: Proc. 9th Int Conf. ICTERI 2013, Kherson, Ukraine, vol. 1000, pp. 64–78. CEUR-WS (June 2013), http://ceur-ws.org/vol-1000
6. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Heidelberg (2007)
7. Davidovsky, M., Ermolayev, V., Tolok, V.: Instance Migration between Ontologies having Structural Differences. International Journal on Artificial Intelligence Tools 20(6), 1127–1156 (2011)
8. Davidovsky, M., Ermolayev, V., Matzke, W.-E., Tolok, V.: Evaluation of Semi-Automated Ontology Instance Migration. In: Essaaidi, M., Malgeri, M., Badica, C., et al. (eds.) Intelligent Distributed Computing IV. SCI, vol. 315, pp. 179–190. Springer, Heidelberg (2010)
9. Davidovsky, M., Ermolayev, V., Tolok, V.: Agent-Based Implementation for the Discovery of Structural Difference in OWL-DL Ontologies. In: Kop, C. (ed.) UNISON 2012. LNBIP, vol. 137, pp. 87–95. Springer, Heidelberg (2013)
10. Corcho, O.: Methodologies, Tools and Languages for Building Ontologies. Where is their meeting point? Data & Knowledge Engineering 46, 41–64 (2003)
11. Ehrig, M.: Ontology Alignment: Bridging the Semantic Gap (Semantic Web and Beyond). Springer (2006)
12. Zhdanova, A.V., de Bruijn, J., Zimmermann, K., Scharffe, F.: Ontology Alignment Solution. Deliverable D14 v2.0 (2004)
13. Davidovsky, M., Ermolayev, V., Tolok, V.: A Survey on Agent-based Ontology Alignment. In: Filipe, J., Fred, A.L.N. (eds.) Proceedings of 4th International Conference on Agents and Artificial Intelligence, ICAART 2012, vol. 2, pp. 355–361. SciTe Press (2012)

14. Ermolayev, V., Davidovsky, M.: Agent-based Ontology Alignment: Basics, Applications, Theoretical Foundations, and Demonstration. In: Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics, WIMS 2012. Article No. 3. ACM, New York (2012)
15. Silver, G., Hassan, O.H., Miller, J.: From Domain Ontologies to Modeling Ontologies to Executable Simulation Models. In: Proc. of the 2007 Winter Simulation Conference, pp. 1108–1117 (2007)
16. Novák, P., Šindelář, R.: Applications of Ontologies for Assembling Simulation Models of Industrial Systems. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM-WS 2011. LNCS, vol. 7046, pp. 148–157. Springer, Heidelberg (2011)
17. Ermolayev, V., Keberle, N., Matzke, W.-E.: An Upper Level Ontological Model for Engineering Design Performance Domain. In: Engelfriet, J. (ed.) Simple Program Schemes and Formal Languages. LNBIP, vol. 20, pp. 127–141. Springer, Heidelberg (1974)
18. Ding, Y., Fensel, D., Klein, M., Omelayenko, B., Schulten, E.: The Role of Ontologies in eCommerce. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies. International Handbooks on Information Systems, pp. 593–616. Springer (2004) ISBN 3-540-40834-7
19. Hepp, M.: GoodRelations: An Ontology for Describing Products and Services Offers on the Web. In: Gangemi, A., Euzenat, J. (eds.) EKAW 2008. LNCS (LNAI), vol. 5268, pp. 329–346. Springer, Heidelberg (2008)
20. Hepp, M.: Products and Services Ontologies: A Methodology for Deriving OWL Ontologies from Industrial Categorization Standards. Int'l Journal on Semantic Web & Information Systems 2(1), 72–99 (2006)
21. Morgenstern, L., Riecken, D.: SNAP: An Action-Based Ontology for E-commerce Reasoning. In: Proceedings of the 1st Workshop on Formal Ontologies Meet Industry, FOMI 2005 (2005)
22. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web (Berners-Lee et. al 2001). Scientific American 284, 28–37 (2001)
23. Izza, S., Vincent, L., Burlat, P.: A Unified Framework for Enterprise Integration: An Ontology-Driven Service-Oriented Approach. In: Pre-proceedings of the First International Conference on Interoperability of Enterprise Software and Applications, INTEROP-ESA 2005, Geneva, Switzerland, February 23-25, pp. 78–89 (2005)
24. Stoutenburg, S., et al.: Ontologies in OWL for Rapid Enterprise Integration. Time 122, 82–89 (1994)
25. Lochovsky, F.H., Woo, C.C., Williams, L.J.: A Micro-organizational Model for Supporting Knowledge Migration. In: Conference on Supporting Group Work Proceedings of the Conference on Office Information Systems, pp. 194–204. ACM, New York (1990)
26. Hedden, H.: Taxonomies for E-Commerce: Best Practices and Design Challenges. In: Gilbane Conference, Boston (November 29, 2012)
27. Information Intelligence: Content Classification and Enterprise Taxonomy Practice. Technical report, Delphi Group (2004)
28. Rahm, E.: Towards Large-scale Schema and Ontology Matching. In: Data-Centric Systems and Applications, 5258, pp. 1–25. Springer (2010)
29. Ehrig, M., Staab, S.: QOM – Quick Ontology Mapping. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 683–697. Springer, Heidelberg (2004)
30. Peukert, E., Berthold, H., Rahm, E.: Rewrite Techniques for Performance Optimization of Schema Matching Processes. In: EDBT, pp. 453–464 (2010)

31. Hamdi, F., Safar, B., Reynaud, C., Zargayouna, H.: Alignment-based Partitioning of Large-scale Ontologies. In: Guillet, F., Ritschard, G., Zighed, D.A., Briand, H. (eds.) Advances in Knowledge Discovery and Management. SCI, vol. 292, pp. 251–269. Springer, Heidelberg (2010)
32. Seddiquia, M.H., Aono, M.: An Efficient and Scalable Algorithm for Segmented Alignment of Ontologies of Arbitrary Size. Web Semantics 7(4), 344–356 (2009)
33. Hu, W., Qu, Y., Cheng, G.: Matching Large Ontologies: A Divide-and-Conquer Approach. Data & Knowledge Engineering 67, 140–160 (2008)
34. Algergawy, A., Massmann, S., Rahm, E.: A Clustering-based Approach For Large-scale Ontology Matching. In: Proceedings of ADBIS, pp. 1–14 (2011)
35. Wang, P., Zhou, Y., Xu, B.: Matching Large Ontologies Based on Reduction Anchors. Most, 2343–2348 (2007)
36. Wang, P.: Lily-LOM: An Efficient System for Matching Large Ontologies with Non-Partitioned Method. In: Proceeding of the ISWC 2010, Posters & Demonstrations Track: Collected Abstracts, Shanghai, China (November 9, 2010)
37. Tenschert, A., Assel, M., Cheptsov, A., Gallizo, G.: Parallelization and Distribution Techniques for Ontology Matching in Urban Computing Environments. In: Proceedings of the Fourth International Workshop on Ontology Matching. Part of the ISWC Conference, Chantilly, USA (2009)
38. Luab, Y., Panettob, H., Nia, Y., Gua, X.: Ontology Alignment for Networked Enterprise Information System Interoperability in Supply Chain Environment. International Journal of Computer Integrated Manufacturing 26(1-2) (2013)
39. Ermolayev, V., Copylov, A., Keberle, N., Jentzsch, E., Matzke, W.-E.: Using Contexts in Ontology Structural Change Analysis. In: Ermolayev, V., Gomez-Perez, J.-M., Haase, P., Warren, P. (eds.) CEUR-WS, vol. 626 (2010)
40. Castano, S., Ferrara, A., Montanelli, S., Varese, G.: Ontology and Instance Matching. In: Paliouras, G., Spyropoulos, C.D., Tsatsaronis, G. (eds.) Multimedia Information Extraction. LNCS, vol. 6050, pp. 167–195. Springer, Heidelberg (2011)
41. Seddiqui, M.H., Das, S., Ahmed, I., Nath, R.P.D., Aono, M.: Augmentation of Ontology Instance Matching by Automatic Weight Generation. In: World Congress on Information and Communication Technologies, pp. 1390–1395. IEEE (2011)
42. Castano, S., Ferrara, A., Montanelli, L.D.: Instance Matching for Ontology Population. In: Proc. of the Sixteenth Italian Symposium on Advanced Database Systems (SEBD), Mondello (PA), Italy (2008)
43. Wang, Q., Yu, X.: Reasoning over OWL/SWRL Ontologies under CWA and UNA for Industrial Applications. In: Wang, D., Reynolds, M. (eds.) AI 2011. LNCS, vol. 7106, pp. 789–798. Springer, Heidelberg (2011)
44. Tzitzikas, Y., Kampouraki, M., Analyti, A.: Curating the Specificity of Metadata while World Models Evolve. In: 9th International Conference on Digital Preservation, iPRES 2012, Toronto (October 5, 2012)
45. Nardi, D., Brachman, R.J.: An Introduction to Description Logics. In: Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.) The Description Logic Handbook. Cambridge University Press, New York
46. Guarino, N.: Formal Ontology and Information Systems. Proceedings of FOIS 1998 46, 3–15 (1998)
47. Motik, B., Patel-Schneider, P.F., Parisa, B. (eds.): OWL 2 Web Ontology Language, 2nd edn. Structural Specification and Functional-Style Syntax, `http://www.w3.org/TR/2012/REC-owl2-syntax-20121211/`

48. Henderson-Sellers, B., Gonzalez-Perez, C.: Standardizing Methodology Metamodelling and Notation: An ISO Exemplar. In: Ershov, A.P., Nepomniaschy, V.A. (eds.) International Symposium on Theoretical Programming. LNBIP, vol. 5, pp. 1–12. Springer, Heidelberg (1974)
49. Ermolayev, V., Keberle, N., Matzke, W.-E., Vladimirov, V.: A Strategy for Automated Meaning Negotiation in Distributed Information Retrieval. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 201–215. Springer, Heidelberg (2005)
50. Ermolayev, V., Ruiz, C., Tilly, M., Jentzsch, E., Gomez-Perez, G.M., Matzke, W.-E.: A Context Model for Knowledge Workers. In: Ermolayev, V., Gomez-Perez, J.-M., Haase, P., Warren, P. (eds.) CIAO 2010, vol. 626, CEUR-WS (2010)
51. Luo, Z.: Computation and Reasoning: A Type Theory for Computer Science. Int. Series of Monographs on Computer Science. Clarendon Press, Oxford (1994)
52. Ermolayev, V., Copylov, A., Keberle, N., Jentzsch, E., Matzke, W.-E.: Using Contexts in Ontology Structural Change Analysis. In: Ermolayev, V., Gomez-Perez, J.-M., Haase, P., Warren, P. (eds.) CIAO 2010, vol. 626, CEUR-WS (2010)
53. David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The Alignment API 4.0. Semantic Web 2(1), 3–10 (2011)
54. Aguirre, J.L., Eckert, K., Euzenat, J., et al.: Results of the Ontology Alignment Evaluation Initiative 2012 (2012), `http://oaei.ontologymatching.org/2012/results/oaei2012.pdf`

Two Semantic Models for Clock Relations in the Clock Constraint Specification Language

Grygoriy Zholtkevych[1], Frédéric Mallet[2],
Iryna Zaretska[1], and Galyna Zholtkevych[1]

[1] V.N. Karazin Kharkiv National University,
Department of Theoretical and Applied Computer Science,
4, Svobody Sqr., 61022, Kharkiv, Ukraine
[2] Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271,
INRIA, Aoste, 06900 Sophia Antipolis, France

Abstract. The Clock Constraint Specification Language (CCSL) has been defined as a formal companion language of the UML Profile for MARTE to allow defining causal and temporal constraints. This chapter attempts to refine the mathematical foundations of the time model of CCSL. It proposes two semantic models both considering ordered binary relations on CCSL clocks and establishes the equivalence of the two models.

Keywords: embedded system, real-time system, time modeling, clock constraint, chronogram, run, formal specification, denotational semantics, operational semantics.

1 Introduction

The increasing importance and complexity of distributed real-time systems (including embedded systems) in the field of Information and Communication Technology demands a constant evolution of engineering models and their mathematical foundations [5]. For instance, whereas multiprocessing architectures were reserved to specific fields like high-performance computing, they have now become ubiquitous and are present in most end-user appliances and computers [3].

There are plenty of mathematical models that can deal with multiprocessor systems: e.g., CSP of C.A.R. Hoare [9], π-calculus of R. Milner [11], abstract state machine model [6], and processing algebra [14]. But these approaches need to be connected to engineering models.

The UML profile for Modelling and Analysis of Real-Time and Embedded systems (MARTE) [2,16] offers extends the Unified Modeling Language (UML) [17,18] with constructed dedicated to real-time embedded systems. It provides an engineering support to design systems but needs a formal support to conduct analysis.

The Object Constraint Language (OCL) [12] can be used along with UML models to specify structural and behavioral constraints. However, the OCL is not adequate to address real-time issues or parallel models. The Clock Constraint

Specification Language (CCSL) [2] has been defined in an annex of MARTE as a way to build logical and temporal constraints on UML model elements.

CCSL is intended to describe the temporal ordering of interactions between components of a distributed software system. It focuses on the ordering of event occurrences, but not on their chronometric characteristics. It relies on a logical time model inspired by the work on synchronous languages [4] and their polychronous extensions [7].

The operational semantics of CCSL [1] defines the set of admissible scenarios for a given CCSL specification. Each scenario is modeled by the sequence of sets. Every such a set consists of events happening at the same time (simultaneously). This paper proposes two semantic models for CCSL clock relations and shows that they are equivalent. It opens a new mathematical ground to reason on CCSL specifications.

2 Logical Time Modeling and CCSL

In this chapter, we restrict ourselves to a basic subset of CCSL constructs and more specifically to the so-called clock relations. A clock model is made of clocks and clock relations. The clocks are sequences of ticks and the relations on the clocks constrain the way ticks can occur. The basic concepts used are captured in the metamodel shown in Fig. 1 as a UML class diagram [17,18].

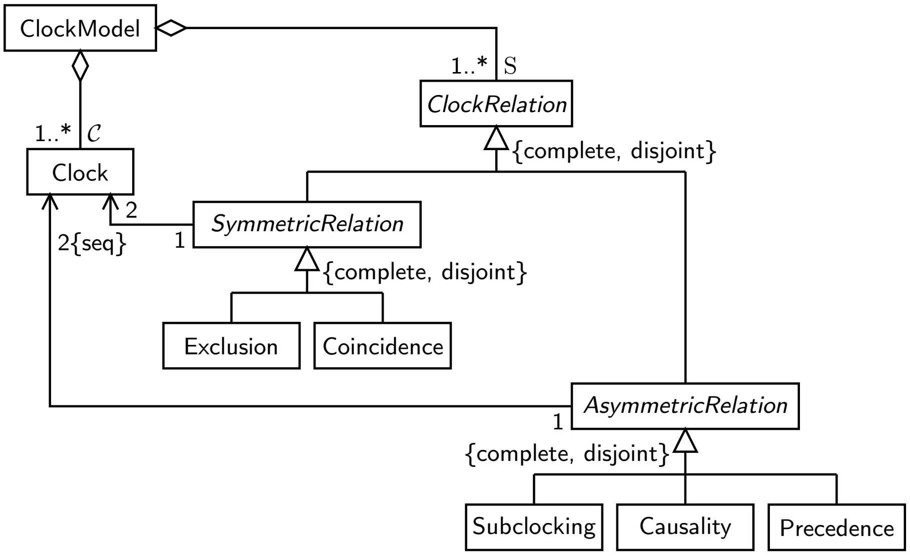


Fig. 1. Abstract syntax of clock relations

A clock model consists of two basic component: a set of clocks $\mathcal{C}$ and a set of clock relations S. Such a set of clock relation we shall call a relational CCSL specification. Table 1 shows the symbols used for clock relations.

Table 1. Symbols of clock relations

Relation name	**Relation symbol**
subclocking	$\boxed{\subset}$
exclusion	$\boxed{\#}$
coincidence	$\boxed{=}$
causality	$\boxed{\preccurlyeq}$
precedence	$\boxed{\prec}$

These five binary relations on a clock set $\mathcal{C}$ are determined as logical primitives for CCSL in [1].

The following three problems are very recurrent in CCSL for providing quality of project solutions.

Problem 1 (Consistency Checking). Let some relational CCSL specification of a system behavior be given.

Determine whether this specification is consistent.

Problem 2 (Checking of Semantic Dependency). Let some relational CCSL specification of a system behavior and a CCSL clock relation be given.

Determine whether this clock relation follows from the specification.

Problem 3 (Minimizing of Specification). Let some relational CCSL specification of a system behavior be given.

Find a relational CCSL specification that has a minimum number of elements in the class of relational CCSL specifications equivalent to the given specification.

CCSL constraints provide a formal language to describe requirements on a system. However, if different semantics are available then one needs to show some equivalences between them. If there are several ways of specifying the same behavior, having a kind of normal form (last problem) gives confidence that the solution provided is the right one.

3 Denotational Semantics of Relational Specification in CCSL

This section describes a denotational approach to define meaning of relational specifications in CCSL. This approach was first formulated in [10]. Under this approach members of a special class of quasi-ordered sets called below chronograms are considered as possible scenarios of system behaviors. The understanding of a clock relation is determined by specifying a class of chronograms that are consistent with this clock relation.

Hence, we start the section with a brief survey containing the necessary definitions about ordered and quasi-ordered sets.

3.1 Survey Some Facts about Ordered and Quasi-Ordered Sets

According to [8] we give below some definitions and assertions of ordered set theory.

Definition 1. *Let $\preccurlyeq$ be a binary relation on a set S. Then $\preccurlyeq$ is called a quasi-order, if it is reflexive and transitive. The pair $(S, \preccurlyeq)$ is called a quasi-ordered set.*

Definition 2. *Let $\equiv$ be a binary relation on a set S. Then $\equiv$ is called an equivalence, if it is reflexive, symmetric and transitive.*

Definition 3. *Let $\leq$ be a quasi-order on a set S. Then $\leq$ is called an order, if it is antisymmetric. The pair $(S, \leq)$ is called a (partially) ordered set or a poset.*

Definition 4. *Let $<$ be a binary relation on a set S. Then $<$ is called a strict order, if it is irreflexive and transitive.*

Proposition 1. *Let $(S, \leq)$ be a poset. For $x, y \in S$ denote by $x < y$ the assertion $x \leq y \ \& \ x \neq y$. Then $<$ is a strict order.*

Conversely, let S be a set, $<$ be a strict order on S. For $x, y \in S$ denote by $x \leq y$ the assertion $x < y \vee x = y$. Then $\leq$ is an order.

Proposition 2. *Let $(S, \preccurlyeq)$ be a quasi-ordered set. For $x, y \in S$ denote by $x \equiv y$ the assertion $x \preccurlyeq y \ \& \ y \preccurlyeq x$ and by $x \prec y$ the assertion $x \preccurlyeq y \ \& \ y \not\preccurlyeq x$. Then $\equiv$ is an equivalence, $\prec$ is a strict order, and the following condition is true:*

$$x \equiv x' \ \& \ y \equiv y' \ \& \ x \prec y \ \textit{implies} \ x' \prec y' \ \textit{for any} \ x, x', y, y' \in S \,. \qquad (1)$$

Conversely, let S be a set, $\equiv$ and $\prec$ be an equivalence and strict order on S respectively such that condition (1) holds. For $x, y \in S$ denote by $x \preccurlyeq y$ the assertion $x \prec y \vee x \equiv y$. Then $\preccurlyeq$ is a quasi-order.

Definition 5. *Let $(S, \preccurlyeq)$ be a quasi-ordered set and A be its subset. Then an element $x \in A$ is called a minimal element of A if for any $y \in A$ the relation $y \prec x$ does not hold.*

Definition 6. *An order relation $\leq$ is called a linear order on a set S, if it is an order and, in addition, the following condition holds: for every two elements $x \neq y$ of S the assertion $x \leq y \vee y \leq x$ is valid.*

Definition 7. *A quasi-order relation $\preccurlyeq$ is called a linear quasi-order on a set S, if it is a quasi-order and, in addition, the following condition holds: for every two elements $x \not\equiv y$ of S the assertion $x \prec y \vee y \prec x$ is valid.*

Definition 8. *Let $(S, \leq)$ be a linear ordered set. Then it is called well-ordered if every non empty subset of S has a minimal element. It is evident that in the case this minimal element is unique and it denotes by* $\inf A$ *.*

Definition 9. *Let $(S, \leq)$ be a poset, a be some element of S. Then we call the set $(a] = \{x \in S \mid x \leq a\}$ the principal ideal of a.*

Similarly, if $(S, \preccurlyeq)$ is a quasi-ordered set and a is some element of S then the set $(a] = \{x \in S \mid x \leq a\}$ is called the principal ideal of a.

Definition 10. *If R is a binary relation on S and $T \subset S$, we define the restriction of R to T, as the relation $R \bigcap (T \times T)$ on T.*

Theorem 1 (Szpilrajn's extension theorem). *Let $(S, \leq)$ be a poset, x_1 and x_2 be some incomparable elements of S then there exists a linear order $\leq_*$ on S such that*

1. *for any $x, y \in S$ the validity of the inequality $x \leq y$ implies the validity of the inequality $x \leq_* y$;*
2. *the inequality $x_1 \leq_* x_2$ is valid.*

Proof. See [8, p. 53, Theorem 3.2] □

3.2 Chronograms

Our approach to define a denotational semantics of clock relations is based on the idea that system behavior is determined by relationships between events that occurred in the system. These relationships are associated with a temporal ordering of the event occurrences. But we focus our attention on the relationship of necessary preconditions which express causal dependencies between event occurrences. In this context it is naturally to suppose that the system behavior is determined by the set of event occurrences, which are below called ***instants***, and a quasi-order on this set, which simulates causality relationship between instants. Our model is not restricted by specifying causality relationships only, but it fixes sources of instants too. More precisely, each instant is associated with some uniquely defined event source, which is called a ***clock***. Summarizing previous informal reasoning we come to the following definition.

Definition 11. *A quadruple $\mathcal{T} = (\mathcal{I}_\mathcal{T}, \preccurlyeq_\mathcal{T}, \mathcal{C}_\mathcal{T}, \pi_\mathcal{T})$, where $\mathcal{I}_\mathcal{T}$ is a set of instants, $\preccurlyeq_\mathcal{T}$ is a quasi-order on $\mathcal{I}_\mathcal{T}$, $\mathcal{C}_\mathcal{T}$ is a finite set of clocks, and $\pi_\mathcal{T}$ is a map from $\mathcal{I}_\mathcal{T}$ into $\mathcal{C}_\mathcal{T}$, is called a* ***chronogram*** *if the following conditions hold:*

$$\text{all principal ideals in } (\mathcal{I}_\mathcal{T}, \preccurlyeq_\mathcal{T}) \text{ are finite;} \tag{2}$$

$$\text{for each } c \in \mathcal{C}_\mathcal{T} \text{ restriction of } \preccurlyeq_\mathcal{T} \text{ on } \mathcal{I}_\mathcal{T}^c = \pi_\mathcal{T}^{-1}(c) \text{ is a linear order.} \tag{3}$$

Note 1. If the context determines the chronogram uniquely then subscripts for its components are usually omitted in the following.

Note 2. The set $\mathcal{I}_\mathcal{T}^c$ for a clock $c \in \mathcal{C}$ will be called a ***time-line*** of the clock c.

The class of all chronograms such that their sets of clocks are equal to some fixed set of clocks $\mathcal{C}$ is below denoted by $\mathfrak{C}(\mathcal{C})$.

The following theorem describes the structure of chronogram time-lines.

Theorem 2. *Let $\mathcal{T} = (\mathcal{I}, \preccurlyeq, \mathcal{C}, \pi)$ be a chronogram then*

1. *restricting of the quasi-order $\preccurlyeq$ on any time-line gives a well-order;*
2. *ordinal type of any time-line is less or equal to ω, where ω is the first infinite ordinal;*
3. *any nonempty subset of $\mathcal{I}$ has a minimal element.*

Proof. Suppose that A is some non-empty subset of $\mathcal{I}^c$ for an arbitrary $c \in \mathcal{C}$ and i is any element of A. We claim that $\inf A \in A$. Indeed, if for all $j \in A$ the statement $i \preccurlyeq j$ is true then $\inf A = i \in A$; and if $\{j \in A \mid j \prec i\} \neq \emptyset$ then it is a finite linearly ordered set (see, conditions (2) and (3) of Def. 11), hence, it contains its greatest lower bound, which is evidently equal to $\inf A$. Thus, the first item of the theorem conclusion is proved.

The supposition that ordinal type of $\mathcal{I}^c$ for some $c \in \mathcal{C}$ is greater than ω is inconsistent with condition (2) of Def. 11.

Let A be a nonempty subset of $\mathcal{I}$. Then $A = \bigcup_{c \in \mathcal{C}} (A \bigcap \mathcal{I}^c)$ and there exists $c \in \mathcal{C}$ such that $A \bigcap \mathcal{I}^c$ is not empty. Let $i(c) = \inf (A \bigcap \mathcal{I}^c)$ if $A \bigcap \mathcal{I}^c$ is not empty. The existence of such elements follows from the first item of the theorem. It is evident that minimal elements of A a members of $\{i(c) \mid c \in \mathcal{C}\}$. Taking into account that the last set is finite we can conclude that it has a minimal element which, evidently, is minimal element of A. □

The following definition identifies some important special subclasses of chronograms.

Definition 12. *A chronogram $\mathcal{T}$ is called* ***linear*** *if for any $i, j \in \mathcal{I}_\mathcal{T}$ the following disjunction is true:*

$$i \preccurlyeq j \vee i \equiv j \vee i \succcurlyeq j\,.$$

The denotation $\mathfrak{C}_{\text{lin}}(\mathcal{C})$ is below used to refer on this subclass of $\mathfrak{C}(\mathcal{C})$.

Definition 13. *Let $\mathcal{T} = (\mathcal{I}, \preccurlyeq_\mathcal{T}, \mathcal{C}, \pi)$ be a chronogram then a linear chronogram $\mathcal{L} = (\mathcal{I}, \preccurlyeq_\mathcal{L}, \mathcal{C}, \pi)$ is called a* ***linear extension*** *of $\mathcal{T}$ if the following conditions hold:*

1. *for any $i, j \in \mathcal{I}$ the coincidence $i \equiv_\mathcal{T} j$ is valid if and only if the coincidence $i \equiv_\mathcal{L} j$ is valid;*
2. *for any $i, j \in \mathcal{I}$ the validity of the precedence $i \prec_\mathcal{T} j$ implies the validity of the precedence $i \prec_\mathcal{L} j$.*

The existence of a linear extension for a chronogram follows from Szpilrajn's extension theorem (see Theorem 1).

Theorem 3. *Let $\mathcal{T} = (\mathcal{I}, \preccurlyeq_\mathcal{T}, \mathcal{C}, \pi)$ be a chronogram, i_1 and i_2 be incomparable instants then there exists a linear extension $\mathcal{L} = (\mathcal{I}, \preccurlyeq_\mathcal{L}, \mathcal{C}, \pi)$ of $\mathcal{T}$ such that the condition $i_1 \prec_\mathcal{L} i_2$ holds.*

Proof. Let us introduce the following notation.
Denote by $[j]$, where $j \in \mathcal{I}$, the $\equiv_\mathcal{T}$-equivalence class of j.

Denote by $\widehat{\mathcal{I}}$ the quotient set of $\mathcal{I}$ with respect to the equivalence $\equiv_{\mathcal{T}}$.
Denote by $\leq$ the order on $\widehat{\mathcal{I}}$ induced by $\preccurlyeq_{\mathcal{T}}$.

Incomparability of i_1 and i_2 implies the validity of $[i_1] \neq [i_2]$. Hence, Szpilrajn's extension theorem ensures that there exists a linear extension of the order $\leq$ denoted by $\leq_*$ such that $[i_1] <_* [i_2]$ holds.

Now define $\preccurlyeq_{\mathcal{L}}$ by the following way:

$$i \preccurlyeq_{\mathcal{L}} j \text{ if and only if } [i] \leq_* [j], \text{ where } i, j \in \mathcal{I}\,.$$

It is easy seen that $\preccurlyeq_{\mathcal{L}}$ is a linear extension of $\preccurlyeq_{\mathcal{T}}$; the equivalences $\equiv_{\mathcal{L}}$ and $\equiv_{\mathcal{T}}$ are identical; and $i_1 \prec_{\mathcal{L}} i_2$ is valid.

We claim that $\mathcal{L} = (\mathcal{I}, \preccurlyeq_{\mathcal{L}}, \mathcal{C}, \pi)$ is a chronogram. Indeed, it is evident that condition (3) of Def. 11 holds.

Further, let i be some instant then $(\,i\,]_{\mathcal{L}} = \bigcup_{a\in\mathcal{C}}(\mathcal{I}^a \bigcap (\,i\,]_{\mathcal{L}})$. Here $(\,i\,]_{\mathcal{L}}$ denotes the set $\{j \in \mathcal{I} \mid j \preccurlyeq_{\mathcal{L}} i\}$. Using item 1 of Theorem 2 we obtain that for each $a \in \mathcal{I}$ there exists $i_a \in \mathcal{I}^a$ such that $\mathcal{I}^a \bigcap (\,i\,]_{\mathcal{L}} = \mathcal{I}^a \bigcap (\,i_a\,]$. Therefore, condition (2) of Def. 11 ensures the finiteness of $(\,i\,]$ □

3.3 Defining of Denotational Semantics

The definition of the denotational semantics of CCSL relational specifications is now carried out step by step.

First of all, let us give a formal definition for a clock relation.

Definition 14. *A triple $a\ \boxed{*}\ b$ is call a clock relation on $\mathcal{C}$ if $\mathcal{C}$ is a finite set of clocks, $a, b \in \mathcal{C}$, and $\boxed{*}$ is a clock relation symbol.*

The set of all clock relations on $\mathcal{C}$ is below denoted by $\mathcal{Rel}(\mathcal{C})$.

Now formulate conditions ensuring correctness the statement 'a chronogram is a model of a clock relation'.

Definition 15. *Let $\mathcal{C}$ be a finite set of clocks then the relation $\mathcal{T} \models a\ \boxed{*}\ b$ on $\mathfrak{C}(\mathcal{C}) \times \mathcal{Rel}(\mathcal{C})$ is defined by the following way:*

$\mathcal{T} \models a\ \boxed{\subset}\ b$ means the existence of a strict monotonic map $h : \mathcal{I}^a \to \mathcal{I}^b$ such that for each $i \in \mathcal{I}^a$ the condition $i \equiv h(i)$ is true;

$\mathcal{T} \models a\ \boxed{\#}\ b$ means that for any $i \in \mathcal{I}^a$ and $j \in \mathcal{I}^b$ the condition $i \equiv j$ is false;

$\mathcal{T} \models a\ \boxed{=}\ b$ means the existence of a strict monotonic bijection $h : \mathcal{I}^a \to \mathcal{I}^b$ such that for each $i \in \mathcal{I}^a$ the condition $i \equiv h(i)$ is true;

$\mathcal{T} \models a\ \boxed{\preccurlyeq}\ b$ means the existence of a strict monotonic map $h : \mathcal{I}^b \to \mathcal{I}^a$ such that for each $i \in \mathcal{I}^b$ the condition $h(i) \preccurlyeq i$ is true;

$\mathcal{T} \models a\ \boxed{\prec}\ b$ means the existence of a strict monotonic map $h : \mathcal{I}^b \to \mathcal{I}^a$ such that for each $i \in \mathcal{I}^b$ the condition $h(i) \prec i$ is true.

We now further define the set of models for an arbitrary relational specification in CCSL.

Definition 16. *Let $\mathcal{C}$ be a finite set of clocks,* S *be some set of clock relations on $\mathcal{C}$ then by $[\![\,\mathrm{S}\,]\!]_{DM}$ we denote the following set*

$$[\![\,\mathrm{S}\,]\!]_{DM} = \left\{\mathcal{T} \in \mathfrak{C}(\mathcal{C}) \mid \mathcal{T} \models a \boxed{*}\, b \textit{ for all } a \boxed{*}\, b \in \mathrm{S}\right\} .$$

We shall need the following property of the set $[\![\,\mathrm{S}\,]\!]_{DM}$.

Proposition 3. *Let $\mathcal{C}$ be a finite set of clocks,* S *be some set of clock relations on $\mathcal{C}$, and $a \boxed{*}\, b$ be a clock relation on $\mathcal{C}$ such that $[\![\,\mathrm{S}\,]\!]_{DM} \subset [\![\, a \boxed{*}\, b \,]\!]_{DM}$ then $[\![\,\mathrm{S} \bigcup \{a \boxed{*}\, b\}\,]\!]_{DM} = [\![\,\mathrm{S}\,]\!]_{DM}$.*

Proof. Indeed, the embedding $[\![\,\mathrm{S} \bigcup \{a \boxed{*}\, b\}\,]\!]_{DM} \subset [\![\,\mathrm{S}\,]\!]_{DM}$ is evidently.
Conversely, if $\mathcal{T} \in [\![\,\mathrm{S}\,]\!]_{DM}$ then taking into account $[\![\,\mathrm{S}\,]\!]_{DM} \subset [\![\, a \boxed{*}\, b \,]\!]_{DM}$ we obtain that $\mathcal{T} \in [\![\,\mathrm{S} \bigcup \{a \boxed{*}\, b\}\,]\!]_{DM}$ □

Finally, define the set of derivable clock relations for each relational specification in CCSL.

Definition 17. *Let $\mathcal{C}$ be a finite set of clocks,* S *be some set of clock relations on $\mathcal{C}$ then we say that the set*

$$[\,\mathrm{S}\,]_{DM} = \left\{a \boxed{*}\, b \in \mathcal{R}el(\mathcal{C}) \mid [\![\,\mathrm{S}\,]\!]_{DM} \subset [\![\, a \boxed{*}\, b \,]\!]_{DM}\right\}$$

is the clock constraint specified by S (*in conformity with denotational semantics*). *We shall use the denotation $\mathrm{S} \Vdash_{DM} a \boxed{*}\, b$ to express the validity of the belonging $a \boxed{*}\, b \in [\,\mathrm{S}\,]_{DM}$.*

Taking into account Prop. 3 one can prove the following property of $[\,\mathrm{S}\,]_{DM}$.

Proposition 4. *Let $\mathcal{C}$ be a finite set of clocks and* S *be some set of clock relations on $\mathcal{C}$ then $[\![\,[\,\mathrm{S}\,]_{DM}\,]\!]_{DM} = [\![\,\mathrm{S}\,]\!]_{DM}$.*

Proof is trivial □

Proposition 5. *Let $\mathcal{C}$ be a finite set of clocks then the map $\mathrm{S} \mapsto [\,\mathrm{S}\,]_{DM}$ from $\mathbf{2}^{\mathcal{R}el(\mathcal{C})}$ into itself is a closure operator on $\mathcal{R}el(\mathcal{C})$.*

Proof. Indeed, if $a \boxed{*}\, b \in \mathrm{S} \subset \mathcal{R}el(\mathcal{C})$ then $[\![\,\mathrm{S}\,]\!]_{DM}$ is a subset of $[\![\, a \boxed{*}\, b \,]\!]_{DM}$ by Def. 16. Taking into account Def. 17 one can conclude that $a \boxed{*}\, b \in [\,\mathrm{S}\,]_{DM}$. It completes checking that the map is extensive.

Further, suppose that $\mathrm{S}_1 \subset \mathrm{S}_2 \subset \mathcal{R}el(\mathcal{C})$ and $a \boxed{*}\, b \in [\,\mathrm{S}_1\,]_{DM}$. It implies that $[\![\,\mathrm{S}_1\,]\!]_{DM} \subset [\![\, a \boxed{*}\, b \,]\!]_{DM}$. Def. 16 ensures that for an arbitrary $\mathcal{T} \in [\![\,\mathrm{S}_2\,]\!]_{DM}$ we have $\mathcal{T} \in [\![\,\mathrm{S}_1\,]\!]_{DM}$. Taking into account this reasoning we obtain that the embedding $[\,\mathrm{S}_1\,]_{DM} \subset [\,\mathrm{S}_2\,]_{DM}$ is true and the map is monotone.

Finally, the property of extensivity for the map ensures the truth of the embedding $\mathrm{S} \subset [\,\mathrm{S}\,]_{DM}$ for any $\mathrm{S} \subset \mathcal{R}el(\mathcal{C})$. Taking into account the monotonicity of the map we have $[\,\mathrm{S}\,]_{DM} \subset [\,[\,\mathrm{S}\,]_{DM}\,]_{DM}$. Conversely, let $a \boxed{*}\, b$ be an arbitrary element of $[\,[\,\mathrm{S}\,]_{DM}\,]_{DM}$ then it is true that $[\![\,[\,\mathrm{S}\,]_{DM}\,]\!]_{DM} \subset [\![\, a \boxed{*}\, b \,]\!]_{DM}$. Using Prop. 3 one can obtain that $[\![\,\mathrm{S}\,]\!]_{DM} \subset [\![\, a \boxed{*}\, b \,]\!]_{DM}$ and, hence, $a \boxed{*}\, b \in [\,\mathrm{S}\,]_{DM}$. □

Now we can give some formal representations for the main problems of specifications analysis.

Problem 1.DM (Consistency Checking). Let S be a relational CCSL specification on a clock set $\mathcal{C}$.

Determine whether the strict embedding $[\, S\,]_{DM} \subsetneq \mathcal{R}el(\mathcal{C})$ holds.

Problem 2.DM (Checking of Semantic Dependency). Let S be a relational CCSL specification on a clock set $\mathcal{C}$ and $a \boxed{*} b$ be a clock relation on the same clock set.

Determine whether the clock relation $a \boxed{*} b$ is a member of $[\, S\,]_{DM}$.

Problem 3.DM (Minimising of Specification). Let S be a relational CCSL specification on a clock set $\mathcal{C}$.

Find a relational CCSL specification S_1 such that

$$[\,\mathrm{S}_1\,]_{DM} = [\,\mathrm{S}\,]_{DM} \text{ and } |\mathrm{S}_1| = \min\{|\mathrm{S}'| \mid [\,\mathrm{S}'\,]_{DM} = [\,\mathrm{S}\,]_{DM}\}\,.$$

3.4 Linear Denotational Semantics Models

Theorem 4. *Let $\mathcal{T} = (\mathcal{I}, \preccurlyeq_{\mathcal{T}}, \mathcal{C}, \pi)$ be a chronogram and $a \boxed{*} b$ be a clock relation then the assertion $\mathcal{T} \models a \boxed{*} b$ if and only if $\mathcal{L} \models a \boxed{*} b$ for each linear extensions $\mathcal{L} = (\mathcal{I}, \preccurlyeq_{\mathcal{L}}, \mathcal{C}, \pi)$ of $\mathcal{T}$.*

We need several lemmas to prove Theorem 4.

Lemma 1. *Let $\mathcal{T} = (\mathcal{I}, \preccurlyeq_{\mathcal{T}}, \mathcal{C}, \pi)$ be a chronogram, $a \boxed{*} b$ be a clock relation, and the assertion $\mathcal{T} \models a \boxed{*} b$ is valid then the assertion $\mathcal{L} \models a \boxed{*} b$ is valid for any linear extension $\mathcal{L} = (\mathcal{I}, \preccurlyeq_{\mathcal{L}}, \mathcal{C}, \pi)$ of $\mathcal{T}$.*

Proof. Only the statement "$\mathcal{T} \models a \boxed{\#} b$ implies $\mathcal{L} \models a \boxed{\#} b$ for any chronogram $\mathcal{T}$, its linear extension $\mathcal{L}$ and a clock relation $a \boxed{\#} b$" is needed an explanation. But Def. 13 ensures the identity of the equivalences $\equiv_{\mathcal{T}}$ and $\equiv_{\mathcal{L}}$ that implies the validity of the analyzed statement □

Lemma 2. *For any chronogram $\mathcal{T} = (\mathcal{I}, \preccurlyeq, \mathcal{C}, \pi)$ the assertion $\mathcal{T} \models a \boxed{=} b$ is valid if and only if both of the assertions $\mathcal{T} \models a \boxed{\subset} b$ and $\mathcal{T} \models b \boxed{\subset} a$ are valid.*

Proof. Let the assertion $\mathcal{T} \models a \boxed{=} b$ be valid then Def. 15 ensures the validity of the assertion $\mathcal{T} \models a \boxed{\subset} b$. Further, let $h : \mathcal{I}^a \to \mathcal{I}^b$ be strict monotonic bijection from Def. 15. Then $h^{-1} : \mathcal{I}^b \to \mathcal{I}^a$ is strict monotonic too. More over, if $i \in \mathcal{I}^b$ then $h(h^{-1}(i)) \equiv h^{-1}(i)$ and, hence, $h^{-1}(i) \equiv i$. Therefore, the assertion $\mathcal{T} \models b \boxed{\subset} a$ is valid.

Conversely, let $\mathcal{T} \models a \boxed{\subset} b$ and $\mathcal{T} \models b \boxed{\subset} a$ be valid. Denote by $h_{ab} : \mathcal{I}^a \to \mathcal{I}^b$ and $h_{ba} : \mathcal{I}^b \to \mathcal{I}^a$ the corresponding strict monotonic maps from Def. 15. If $i \in \mathcal{I}^a$ then $h_{ba}(h_{ab}(i)) \equiv h_{ab}(i) \equiv i$. Taking into account that $h_{ba}(h_{ab}(i))$ and i belong $\mathcal{I}^a$ one can derives the equality $h_{ba}(h_{ab}(i)) = i$. Similar reasoning lead us to conclusion that for each $i\mathcal{I}^b$ the equality $h_{ba}(h_{ab}(i)) = i$ holds. Hence, h_{ab} is a bijection and the assertion $\mathcal{T} \models a \boxed{=} b$ is valid □

Now we are ready to prove Theorem 4.

Proof (of Theorem 4). Suppose that the assertion $\mathcal{T} \models a \boxed{*} b$ holds for a chronogram $\mathcal{T}$ and a clock relation $a \boxed{*} b$ then Lemma 1 ensures the validity of the assertion $\mathcal{L} \models a \boxed{*} b$ for any linear extension $\mathcal{L}$ of $\mathcal{T}$.

To prove converse is necessary to analyse each kind of clock relations separately.

First of all note that if for some linear extension of $\mathcal{T}$ and the clock relation $a \boxed{\subset} b$ the assertion $\mathcal{L} \models a \boxed{\subset} b$ is true then the assertion $\mathcal{T} \models a \boxed{\subset} b$ is true too. It follows directly from Def. 13.

Further, Lemma 2 ensures that the similarly statement is true for a clock relation $a \boxed{=} b$.

The validity of the similar statement for a clock relation $a \boxed{\#} b$ follows from Def. 13 too.

Now let us analyse the case of a clock relation $a \boxed{\preccurlyeq} b$. To do this we assume that the assertion $\mathcal{L} \models a \boxed{\preccurlyeq} b$ is true for any linear extension $\mathcal{L}$ of some chronogram $\mathcal{T}$. Denote by $h_{\mathcal{L}}$ a strict monotonic map from $\mathcal{I}^b$ into $\mathcal{I}^a$ such that $h_{\mathcal{L}}(i) \preccurlyeq_{\mathcal{L}} i$ for all $i \in \mathcal{I}^b$. Taking into account that $h_{\mathcal{L}}(i) \in \mathcal{I}^a$ for all $i \in \mathcal{I}^b$ and $\mathcal{I}^a$ is well-ordered we can conclude that

firstly, $h(i) = \inf\{h_{\mathcal{L}}(i) \mid \text{ for all linear extensions } \mathcal{L} \text{ of } \mathcal{T}\}$ is a correctly defined map from $\mathcal{I}^b$ into $\mathcal{I}^a$;

secondly, for each $i \in \mathcal{I}^b$ there exists a linear extension $\mathcal{L}(i)$ of $\mathcal{T}$ such that $h(i) = h_{\mathcal{L}(i)}(i)$.

Further, if $i, j \in \mathcal{I}^b$ and the precedence $i \prec_{\mathcal{T}} j$ is true then for each linear extension $\mathcal{L}$ of $\mathcal{T}$ the precedence $h_{\mathcal{L}}(i) \prec_{\mathcal{L}} h_{\mathcal{L}}(j)$ is true too. Hence, $h(i) \prec_{\mathcal{L}} h_{\mathcal{L}}(j)$ for each $\mathcal{L}$. The last precedence for $\mathcal{L} = \mathcal{L}(j)$ gives $h(i) \prec_{\mathcal{L}(j)} h_{\mathcal{L}(j)}(j) = h(j)$. Therefore, we obtain $h(i) \prec_{\mathcal{T}} h(j)$.

Finally, suppose that there exists $i_0 \in \mathcal{I}^b$ such that the causality $h(i_0) \preccurlyeq_{\mathcal{T}} i_0$ is false. Under this supposition only two cases can be possible: $h(i_0) \succ_{\mathcal{T}} i_0$ is true or $h(i_0) \parallel_{\mathcal{T}} i_0$ is true. But the validity of $h(i_0) \succ_{\mathcal{T}} i_0$ contradicts to the hypothesis of the theorem. Therefore only case of the validity of $h(i_0) \parallel_{\mathcal{T}} i_0$ should be analysed. Theorem 3 ensures the existence of a linear extension $\mathcal{L}$ of $\mathcal{T}$ such that $h(i_0) \succ_{\mathcal{T}} i_0$. It contradicts to the hypothesis of the theorem. Hence, we can conclude that $h(i) \preccurlyeq_{\mathcal{T}} i$ for any $i \in \mathcal{I}^b$ and, therefore, the assertion $\mathcal{T} \models a \boxed{\preccurlyeq} b$ is true.

Similar reasoning suggest that the theorem is true for clock relations having the form $a \boxed{\prec} b$ □

Corollary 1. *Let us define for an arbitrary* $\mathrm{S} \subset \mathcal{R}el(\mathcal{C})$ *the following sets*

$$[\![\,\mathrm{S}\,]\!]_{LDM} = \left\{\mathcal{L} \in \mathfrak{C}_{\text{lin}} \mid \mathcal{L} \models a \boxed{*} b \text{ for all } a \boxed{*} b \in \mathrm{S}\right\} \text{ and} \tag{4}$$

$$[\,\mathrm{S}\,]_{LDM} = \left\{a \boxed{*} b \in \mathcal{R}el(\mathcal{C}) \mid [\![\,\mathrm{S}\,]\!]_{LDM} \subset [\![\, a \boxed{*} b \,]\!]_{LDM}\right\} \tag{5}$$

then $[\,\mathrm{S}\,]_{DM} = [\,\mathrm{S}\,]_{LDM}$.

4 Operational Semantics of Relational Specifications in CCSL

An alternative way to determine meaning of a clock relation is to specify a sequence of firings for a system satisfying it. The way leads us to the notion of a run [1].

Definition 18. *Let $\mathcal{C}$ be a finite set of clocks and $\boldsymbol{r} = \{\boldsymbol{r}_t \mid t = 1, 2, \ldots\}$ be a sequence of its subsets such that the following condition holds*

$$\boldsymbol{r}_t = \emptyset \textit{ for some } t \textit{ implies } \boldsymbol{r}_s = \emptyset \textit{ for all } s > t \tag{6}$$

*then the sequence is called a **run**.*

The set of all runs for a set of clocks $\mathcal{C}$ is below denoted by $\mathfrak{R}(\mathcal{C})$.

For each run $\boldsymbol{r}$ let us define two functions $\tau_{\boldsymbol{r}}, \chi_{\boldsymbol{r}} : \mathcal{C} \times \mathbb{N} \to \mathbb{N}$ by the next formulae

$$\tau_{\boldsymbol{r}}(c, t) = \begin{cases} 1, & c \in \boldsymbol{r}_t \\ 0, & c \notin \boldsymbol{r}_t \end{cases}, \tag{7}$$

$$\chi_{\boldsymbol{r}}(c, t) = \sum_{s=1}^{t} \tau_{\boldsymbol{r}}(c, s)\,. \tag{8}$$

In [1] the function $\chi_{\boldsymbol{r}}(\cdot, t)$ is called a configuration of the system.

Now we can define semantic meaning for each clock relation by specifying the set of its admissible runs. Similarly to Subsection 3.3 we use the notation $\boldsymbol{r} \models a \boxed{*}\, b$ to indicate that the run $\boldsymbol{r}$ is admissible for the clock relation $a \boxed{*}\, b$ (in other words, the run is a model of the clock relation). To define meaning for each kind of clock relations we formulate the corresponding invariant conditions with respect to t.

Definition 19. *Let $\mathcal{C}$ be a finite set of clocks then the relation $\boldsymbol{r} \models a \boxed{*}\, b$ on $\mathfrak{R}(\mathcal{C}) \times \mathcal{R}el(\mathcal{C})$ is defined by the following way:*

$\boldsymbol{r} \models a \boxed{\subset}\, b$ *means that for all $t \in \mathbb{N}$ the inequality $\tau_{\boldsymbol{r}}(a, t) \le \tau_{\boldsymbol{r}}(b, t)$ is valid;*
$\boldsymbol{r} \models a \boxed{\#}\, b$ *means that for all $t \in \mathbb{N}$ the equality $\tau_{\boldsymbol{r}}(a, t) \cdot \tau_{\boldsymbol{r}}(b, t) = 0$ is valid;*
$\boldsymbol{r} \models a \boxed{=}\, b$ *means that for all $t \in \mathbb{N}$ the equality $\tau_{\boldsymbol{r}}(a, t) = \tau_{\boldsymbol{r}}(b, t)$ is valid;*
$\boldsymbol{r} \models a \boxed{\preccurlyeq}\, b$ *means that for all $t \in \mathbb{N}$ the inequality $\chi_{\boldsymbol{r}}(a, t) \ge \chi_{\boldsymbol{r}}(b, t)$ is valid;*
$\boldsymbol{r} \models a \boxed{\prec}\, b$ *means that the equality $\tau_{\boldsymbol{r}}(b, 1) = 0$ holds and for all $t \in \mathbb{N}$ either the inequality $\chi_{\boldsymbol{r}}(a, t) > \chi_{\boldsymbol{r}}(b, t)$ is valid or the equalities $\chi_{\boldsymbol{r}}(a, t) = \chi_{\boldsymbol{r}}(b, t)$ and $\tau_{\boldsymbol{r}}(b, t + 1) = 0$ are valid.*

Similarly to Def. 16 and Def. 17 we can define the following sets.

Definition 20. *Let $\mathcal{C}$ be a finite set of clocks, S be some set of clock relations on $\mathcal{C}$ then by $[\![\, \mathrm{S} \,]\!]_{OM}$ we denote the following set*

$$[\![\, \mathrm{S} \,]\!]_{OM} = \left\{ \boldsymbol{r} \in \mathfrak{R}(\mathcal{C}) \mid \boldsymbol{r} \models a \boxed{*}\, b \textit{ for all } a \boxed{*}\, b \in \mathrm{S} \right\}.$$

Definition 21. *Let $\mathcal{C}$ be a finite set of clocks,* S *be some set of clock relations on $\mathcal{C}$ then we say that the set*

$$[\,\mathrm{S}\,]_{OM} = \left\{a \boxed{*} b \in \mathcal{R}el(\mathcal{C}) \mid [\![\,\mathrm{S}\,]\!]_{OM} \subset \left[\!\!\left[\, a \boxed{*} b \,\right]\!\!\right]_{OM}\right\}$$

is the clock constraint specified by S *(in conformity with operational semantics). We shall use the denotation* $\mathrm{S} \Vdash_{OM} a \boxed{*} b$ *to express the validity of the belonging* $a \boxed{*} b \in [\,\mathrm{S}\,]_{OM}$.

Now we can give another formal representation for the main problem of specification analysis.

Problem 1.OM (Consistency Checking). Let S be a relational CCSL specification on a clock set $\mathcal{C}$.

Determine whether the strict embedding $[\,S\,]_{OM} \subsetneqq \mathcal{R}el(\mathcal{C})$ holds.

Problem 2.OM (Checking of Semantic Dependency). Let S be a relational CCSL specification on a clock set $\mathcal{C}$ and $a \boxed{*} b$ be a clock relation on the same clock set.

Determine whether the clock relation $a \boxed{*} b$ is a member of $[\,S\,]_{OM}$.

Problem 3.OM (Minimising of Specification). Let S be a relational CCSL specification on a clock set $\mathcal{C}$.

Find a relational CCSL specification S_1 such that

$$[\,\mathrm{S}_1\,]_{OM} = [\,\mathrm{S}\,]_{OM} \text{ and } |\mathrm{S}_1| = \min\{|\mathrm{S}'| \mid [\,\mathrm{S}'\,]_{OM} = [\,\mathrm{S}\,]_{OM}\}\,.$$

5 Linear Chronograms and Runs

It appears that linear chronograms and runs on the same set of clocks are closely connected. The aim of this section is to study this connection.

We start our studying from the following propositions.

Proposition 6. *Let $\mathcal{C}$ be a finite set of clocks, $\boldsymbol{r}$ be an element of $\mathfrak{R}(\mathcal{C})$ such that $\boldsymbol{r}_1 \neq \emptyset$ and let us define:*

- *the set $\mathcal{I}_{\boldsymbol{r}} = \{(c,t) \in \mathcal{C} \times \mathbb{N} \mid c \in \boldsymbol{r}_t\},$;*
- *the quasi-order $\preccurlyeq_{\boldsymbol{r}}$ on $\mathcal{I}_{\boldsymbol{r}}$ such that $(c_1,t_1) \preccurlyeq_{\boldsymbol{r}} (c_2,t_2)$ if and only if $t_1 \leq t_2$;*
- *$\pi_{\boldsymbol{r}}((c,t)) = c$ for any $(c,t) \in \mathcal{I}_{\boldsymbol{r}}$.*

Then the quadruple $\mathcal{L}(\boldsymbol{r}) = (\mathcal{I}_{\boldsymbol{r}}, \mathcal{C}, \preccurlyeq_{\boldsymbol{r}}, \pi_{\boldsymbol{r}})$ is a linear chronogram.

Proof. We need to check the finiteness condition for each principal ideal of $\mathcal{I}_{\boldsymbol{r}}$ and linear ordering of each clock time-line. Taking into account that $((c,t)] = \{(c',t') \in \mathcal{I}_{\boldsymbol{r}} \mid t' \leq t\}$ one can easy get that the number of elements of $((c,t)]$ is less or equal to $|\mathcal{C}| \cdot t$.

Further, it is easy seen that $\mathcal{I}^c_{\boldsymbol{r}} = \{(c,t) \in \mathcal{C} \times \mathbb{N} \mid (c,t) \in \mathcal{I}_{\boldsymbol{r}}\}$. Hence, $\mathcal{I}^c_{\boldsymbol{r}}$ is a subset of $\{c\} \times \{t \in \mathbb{N} \mid c \in \boldsymbol{r}_t\}$. More over, the restriction of $\preccurlyeq_{\boldsymbol{r}}$ on $\mathcal{I}_{\boldsymbol{r}^c}$ coincides with the natural order on $\mathbb{N}$, therefore, $\mathcal{I}^c_{\boldsymbol{r}}$ is linear ordered $\square$

Proposition 7. *Let $\mathcal{C}$ be a finite set of clocks, $\mathcal{L} = (\mathcal{I}, \preccurlyeq, \mathcal{C}, \pi)$ be an element of $\mathfrak{C}_{\mathrm{lin}}(\mathcal{C})$ and let us suppose that*

- *L_1 is the set of all minimal elements of $\mathcal{I}$;*
- *L_{t+1} is the set of all minimal elements of $\mathcal{I} \setminus \bigcup_{s=1}^{t} L_s$;*
- *$\boldsymbol{r}(\mathcal{L})_t = \pi(L_t)$ for all $t = 1, 2, \dots$.*

Then $\boldsymbol{r}(\mathcal{L}) = \{\boldsymbol{r}(\mathcal{L})_t \mid t = 1, 2, \dots\}$ is a run such that $\boldsymbol{r}_1 \neq \emptyset$.

Proof. Enough to check condition (6) to get the proof. Suppose that t_0 is such an element of $\mathbb{N}$ that satisfies the equality $\boldsymbol{r}(\mathcal{L})_{t_0} = \emptyset$. Taking into account that $\mathbb{N}$ is well-ordered we can choose t_0 such that $\pi(L_{t_0}) = \emptyset$ and t_0 is the least number among such ones. But for this t_0 we have $L_{t_0} = \emptyset$.

Firstly, let us consider the case when $t_0 = 1$. Then there is no any minimal element in $\mathcal{I}$. Thus, there exists an infinite decreasing sequence $i_0 \succ i_1 \succ \dots$ in $\mathcal{I}$ but it contradicts to property (2) of Def. 11. Hence, $t_0 > 1$ is true.

Further, in a similar manner one can demonstrate that the inequality $t_0 > 1$ contradicts to property (2) of Def. 11 too □

Theorem 5. *Let $\mathcal{C}$ be a finite set of clocks. Then for any $\boldsymbol{r} \in \mathfrak{R}(\mathcal{C})$ such that $\boldsymbol{r}_1 \neq \emptyset$ the equality $\boldsymbol{r}(\mathcal{L}(\boldsymbol{r})) = \boldsymbol{r}$ holds.*
Similarly, for any $\mathcal{L} \in \mathfrak{C}_{\mathrm{lin}}(\mathcal{C})$ the chronograms $\mathcal{L}$ and $\mathcal{L}(\boldsymbol{r}(\mathcal{L}))$ are isotonic.

Proof. To prove the first part of the theorem let us establish that for $\mathcal{L}(\boldsymbol{r})$ the equality $L_t = \{(c, t) \in \mathcal{C} \times \mathbb{N} \mid c \in \boldsymbol{r}_t\}$ is true.

Indeed, the following statement is evident:

$$\text{if } \boldsymbol{r}_1 \neq \emptyset \text{ then } L_1 = \{(c, 1) \in \mathcal{C} \times \mathbb{N} \mid c \in \boldsymbol{r}_1\}.$$

Hence, $\boldsymbol{r}(\mathcal{L}(\boldsymbol{r}))_1 = \pi_{\boldsymbol{r}}(L_1) = \boldsymbol{r}_1$.

Further, suppose that $L_s = \{(c, s) \in \mathcal{C} \times \mathbb{N} \mid c \in \boldsymbol{r}_s\}$ is true for all $s = 1, \dots, t$. In this case we have the following equality

$$\mathcal{I}_{\boldsymbol{r}} \setminus \bigcup_{s=1}^{t} L_s = \{(c, u) \in \mathcal{C} \times \mathbb{N} \mid u > t \text{ and } c \in \boldsymbol{r}_u\}.$$

This equality and property (6) imply that $L_{t+1} = \{(c, t+1) \in \mathcal{C} \times \mathbb{N} \mid c \in \boldsymbol{r}_{t+1}\}$. Thus, $\boldsymbol{r}(\mathcal{L}(\boldsymbol{r}))_{t+1} = \pi_{\boldsymbol{r}}(L_{t+1}) = \boldsymbol{r}_{t+1}$. Now the correctness of the first part of the theorem is evident.

To prove the second part of the theorem we note that the following statements are true:

1. if $t_1, t_2 \in \mathbb{N}$ then the equality $L_{t_1} = L_{t_2}$ implies $t_1 = t_2$;
2. $\mathcal{I} = \bigcup_{t \in \mathbb{N}} L_t$.

The first statement follows from linear ordering of $\mathbb{N}$ and the definition of the sets L_t where $t = 1, 2, \dots$.

The second statement is evident if the set $\{t \in \mathbb{N} \mid L_t \neq \emptyset\}$ is finite. But if this set is infinite then the supposition about the existence of such an instant i_0 that belongs to $\mathcal{I} \setminus \bigcup_{t\in\mathbb{N}} L_t$ implies the infiniteness of the principal ideal of i_0. It contradicts to property (2) of Def. 11.

These statements make possible to determine the function $\theta : \mathcal{I} \to \mathbb{N}$ by the following condition: $\theta(i) = t$ if $i \in L_t$.

Now let us define $\zeta : \mathcal{I} \to \mathcal{C} \times \mathbb{N}$ by the following way $\zeta(i) = (\pi(i), \theta(i))$. It is easy seen that for all $i, j \in \mathcal{I}$ the causality $i \preccurlyeq_{\mathcal{L}} j$ implies the validity of the inequality $\theta(i) \leq \theta(j)$. It means that the causality $i \preccurlyeq_{\mathcal{L}} j$ implies the causality $(\pi(i), \theta(i)) \preccurlyeq_{\boldsymbol{r}(\mathcal{L})} (\pi(j), \theta(j))$. Thus, the monotonicity of the map ζ is proved.

Further, suppose that for $i_1, i_2 \in \mathcal{I}$ the equality $(\pi(i_1), \theta(i_1)) = (\pi(i_2), \theta(i_2))$ is true then denote by c the same value of $\pi(i_1)$ and $\pi(i_2)$ and denote by t the same value of $\theta(i_1)$ and $\theta(i_2)$. Hence, we have $i_1, i_2 \in \mathcal{I}^c$ and $i_1, i_2 \in L_t$. It and property (3) of Def. 11 ensure the equality $i_1 = i_2$. Therefore, the injectivity of the map ζ is proved.

Let i be an instant such that $\pi(i) = c$ and $\theta(i) = t$ then $i \in \mathcal{I}^c \bigcap L_t$. The last belonging ensures that $\boldsymbol{r}(\mathcal{L})_t = \pi(L_t)$ contains c . Thus, $(c, t) = \zeta(i) \in \mathcal{I}_{\boldsymbol{r}(\mathcal{L})}$ and $\zeta(\mathcal{I}) \subset \mathcal{I}_{\boldsymbol{r}(\mathcal{L})}$. Conversely, let $(c, t) \in \mathcal{I}_{\boldsymbol{r}(\mathcal{L})}$. It means that $c \in \boldsymbol{r}(\mathcal{L})_t = \pi(L_t)$ and, therefore, there exists $i \in \mathcal{I}$ such that $i \in \mathcal{I}^c \bigcap L_t$. But in this case we have $\pi(i) = c$ and $\theta(i) = t$, i.e. $\zeta(i) = (c, t)$. Thus, $\zeta(\mathcal{I}) = \mathcal{I}_{\boldsymbol{r}(\mathcal{L})}$.

Finally, let $(c_1, t_1), (c_2, t_2) \in \mathcal{I}_{\boldsymbol{r}(\mathcal{L})}$ and $t_1 \leq t_2$. As it is shown above there exist $i_1, i_2 \in \mathcal{I}$ such that $\zeta(i_1) = (c_1, t_1)$ and $\zeta(i_2) = (c_2, t_2)$. Hence, $i_1 \in L_{t_1}$ and $i_2 \in L_{t_2}$. It, the inequality $t_1 \leq t_2$, and linearity of $\preccurlyeq_{\mathcal{L}}$ ensure the causality $i_1 \preccurlyeq_{\mathcal{L}} i_2$.

Summing these arguments one can conclude that ζ is an isotonic map □

Corollary 2. *Let $\theta : \mathcal{I} \to \mathbb{N}$ be the map built in the proof of the Theorem 5 then it is strict monotonic.*

Now we can establish the interdependence between the relationship $\models$ on $\mathfrak{C}_{\mathrm{lin}}(\mathcal{C}) \times \mathcal{R}el(\mathcal{C})$ and the relationship $\models$ on $\mathfrak{R}(\mathcal{C}) \times \mathcal{R}el(\mathcal{C})$. The following theorem describes it.

Theorem 6. *Let $\mathcal{C}$ be a finite set of clocks then $\mathcal{L} \models a \boxed{*} b$ if and only if $\boldsymbol{r}(\mathcal{L}) \models a \boxed{*} b$ for any $\mathcal{L} \in \mathfrak{C}_{\mathrm{lin}}(\mathcal{C})$ and $a \boxed{*} b \in \mathcal{R}el(\mathcal{C})$.*

We need the following lemmas to prove the theorem.

Lemma 3. *Let $\mathcal{C}$ be a finite set of clocks, $\mathcal{L}$ be some linear chrogram on $\mathcal{C}$, θ be a map that has built in the proof of Theorem 5 and let us denote the set $\{i \in \mathcal{I}^c \mid \theta(i) \leq t\}$ by $\mathcal{I}^c(t)$ for any $c \in \mathcal{C}$ and $t \in \mathbb{N}$. Then the following equality holds*

$$\chi_{\boldsymbol{r}(\mathcal{L})}(c, t) = |\mathcal{I}^c(t)| \,. \tag{9}$$

Proof. Indeed, it is evident that

$$\chi_{\boldsymbol{r}(\mathcal{L})}(c, t) =$$
$$|\{(c, s) \in \mathcal{C} \times \mathbb{N} \mid s \leq t \text{ and } \tau_{\boldsymbol{r}(\mathcal{L})}(c, s) > 0\}| = |\{(c, s) \in \mathcal{I}_{\boldsymbol{r}(\mathcal{L})} \mid s \leq t\}|$$

Taking into account that $\zeta = (\pi, \theta)$ is a one-to-one correspondence between $\{i \in \mathcal{I}^c \mid \theta(i) \leq t\}$ and $\{(c, s) \in \mathcal{I}_{\boldsymbol{r}(\mathcal{L})} \mid s \leq t\}$ one can conclude that equality (9) is valid $\square$

Lemma 4. *Let $\mathcal{C}$ be a finite set of clocks, $\mathcal{L}$ be some linear chronogram on $\mathcal{C}$, θ be a map that has built in the proof of Theorem 5, a and b be some different elements of $\mathcal{C}$. Then the existence of a strict monotonic map $h : \mathcal{I}^b \to \mathcal{I}^a$ such that $h(i) \preccurlyeq i$ for all $i \in \mathcal{I}^b$ implies the embedding $h(\mathcal{I}^b(t)) \subset \mathcal{I}^a(t)$ for any $t \in \mathbb{N}$.*

Proof. Suppose that $j \in h(\mathcal{I}^b(t))$ then there exists the $i \in \mathcal{I}^b(t)$ such that $j = h(i)$. Hence, the inequality $\theta(i) \leq t$ is valid. Thus, we have

$$\theta(j) \leq \theta(i) \leq t\,.$$

Therefore, $j \in \mathcal{I}^a(t)$ and the required inequality has been proven. $\square$

Proof (of Theorem 6). Let us prove the theorem for each kind of clock relations separately.

Subclocking. Let $\mathcal{L}$ be an element of $\mathfrak{C}_{\text{lin}}(\mathcal{C})$ and $a \boxed{\subset} b$ be the subclocking relation on $\mathcal{C}$ such that $\mathcal{L} \models a \boxed{\subset} b$. Then there exists a strict monotonic map $h : \mathcal{I}^a \to \mathcal{I}^b$ such that $h(i) \equiv i$ for any $i \in \mathcal{I}^a$. The linearity of $\mathcal{L}$ implies that $\theta(i) = \theta(h(i))$ for any $i \in \mathcal{I}^a$ where θ has been built in the proof of the previous theorem.

Suppose that $\tau_{\boldsymbol{r}(\mathcal{L})}(a, t) = 1$ then $a \in \boldsymbol{r}(\mathcal{L})_t$ and, hence, $(a, t) \in \mathcal{I}^a_{\boldsymbol{r}(\mathcal{L})}$. Taking into account the previous theorem we can assert that there exists the instant $i \in \mathcal{I}$ such that $\zeta(i) = (a, t)$ where ζ has been built in the proof of the previous theorem. Therefore, $i \in \mathcal{I}^a$ and $\theta(i) = t$. Now consider the pair $(b, \theta(h(i))) = (\pi(h(i)), \theta(h(i))) = \zeta(h(i))$. Thus, we have obtained that $(b, \theta(h(i))) \in \mathcal{I}^b_{\boldsymbol{r}}$ and $\theta(h(i)) = \theta(i) = t$. Hence, $b \in \boldsymbol{r}(\mathcal{L})_t$ and $\tau_{\boldsymbol{r}(\mathcal{L})}(b, t) = 1$. Summing these arguments we obtain inequality $\tau_{\boldsymbol{r}(\mathcal{L})}(a, t) \leq \tau_{\boldsymbol{r}(\mathcal{L})}(b, t)$ for any $t \in \mathbb{N}$ that means the validity of $\boldsymbol{r}(\mathcal{L}) \models a \boxed{\subset} b$.

Conversely, let $\boldsymbol{r}(\mathcal{L}) \models a \boxed{\subset} b$ be valid. It means that $\tau_{\boldsymbol{r}(\mathcal{L})}(a, t) \leq \tau_{\boldsymbol{r}(\mathcal{L})}(b, t)$ for all $t \in \mathbb{N}$. Hence, $a \in \boldsymbol{r}(\mathcal{L})_t$ implies $b \in \boldsymbol{r}(\mathcal{L})_t$ for any $t \in \mathbb{N}$. Taking into account it one can conclude that $(a, t) \in \mathcal{I}_{\boldsymbol{r}(\mathcal{L})}$ implies $(b, t) \in \mathcal{I}_{\boldsymbol{r}(\mathcal{L})}$. Thus, the formula $h(i) = \zeta^{-1}(b, \theta(i))$ defines the map $h : \mathcal{I}^a \to \mathcal{I}^b$ correctly. If $i_1, i_2 \in \mathcal{I}^a$ and the precedence $i_1 \prec_{\mathcal{L}} i_2$ is true then $\theta(i_1) < \theta(i_2)$ and $(b, \theta(i_1)) \prec_{\boldsymbol{r}(\mathcal{L})} (b, \theta(i_2))$. Hence, $h(i_1) = \zeta^{-1}(b, \theta(i_1)) \prec_{\boldsymbol{r}(\mathcal{L})} \zeta^{-1}(b, \theta(i_2)) = h(i_2)$ and the map h is strict monotonic. Further, for $i \in \mathcal{I}^a$ we have $\zeta(h(i)) = (b, \theta(i)) \equiv_{\boldsymbol{r}(\mathcal{L})} (a, \theta(i)) = \zeta(i)$. Hence, $h(i) \equiv_{\mathcal{L}} i$. Summing these arguments we obtain $\mathcal{L} \models a \boxed{\subset} b$.

Exclusion. Let $\mathcal{L}$ be an element of $\mathfrak{C}_{\text{lin}}(\mathcal{C})$ and $a \boxed{\#} b$ be an exclusion relation on $\mathcal{C}$ such that $\mathcal{L} \models a \boxed{\#} b$. Then for any $i \in \mathcal{I}^a$ and $j \in \mathcal{I}^b$ the coincidence $i \equiv_{\mathcal{L}} j$ is false. Taking into account the linearity of $\mathcal{L}$ one can conclude that either $i \prec_{\mathcal{L}} j$ is true or $i \succ_{\mathcal{L}} j$ is true, hence, for any t either the instant i is not a member of L_t or the instant j is not a member of L_t. Thus, for each $t \in \mathbb{N}$

the assertion $a \in \boldsymbol{r}(\mathcal{L})_t$ & $b \in \boldsymbol{r}(\mathcal{L})_t$ cannot be valid. Therefore, the equality $\tau_{\boldsymbol{r}(\mathcal{L})}(a,t) \cdot \tau_{\boldsymbol{r}(\mathcal{L})}(b,t) = 0$ holds for all t. It means the validity of $\boldsymbol{r}(\mathcal{L}) \models a \boxed{\#} b$. Conversely, let the assertion $\boldsymbol{r}(\mathcal{L}) \models a \boxed{\#} b$ be valid. It means that the equality $\tau_{\boldsymbol{r}(\mathcal{L})}(a,t) \cdot \tau_{\boldsymbol{r}(\mathcal{L})}(b,t) = 0$ holds for all $t \in \mathbb{N}$. Thus, the assertions $a \in \boldsymbol{r}(\mathcal{L})_t$ and $b \in \boldsymbol{r}(\mathcal{L})_t$ cannot be valid for the same t.

Further, suppose that there exist $i \in \mathcal{I}^a$ and $j \in \mathcal{I}^b$ such that $i \equiv_{\mathcal{L}} j$. But then $\theta(i) = \theta(j)$ and we can denote this common value by t_0. Hence, the belongings $i \in L_{t_0}$ and $j \in L_{t_0}$ are true. From this the validity of the statements $a \in \boldsymbol{r}(\mathcal{L})_{t_0}$ and $b \in \boldsymbol{r}(\mathcal{L})_{t_0}$ follows immediately. Derived contradiction suggests that the assumption of the existence $i \in \mathcal{I}^a$ and $j \in \mathcal{I}^b$ such that $i \equiv_{\mathcal{L}} j$ is false. Therefore, $\mathcal{L} \models a \boxed{\#} b$.

Coincidence. Let $\mathcal{L}$ be an element of $\mathfrak{C}_{\text{lin}}(\mathcal{C})$ and $a \boxed{=} b$ be a coincidence relation on $\mathcal{C}$ such that $\mathcal{L} \models a \boxed{=} b$. Using Lemma 2 we obtain that the following assertions $\mathcal{L} \models a \boxed{\subset} b$ and $\mathcal{L} \models b \boxed{\subset} a$ are true. As it is proved above the last assertions imply that the assertions $\boldsymbol{r}(\mathcal{L}) \models a \boxed{\subset} b$ and $\boldsymbol{r}(\mathcal{L}) \models b \boxed{\subset} a$ are valid too. Hence, for any $t \in \mathbb{N}$ the inequalities $\tau_{\boldsymbol{r}(\mathcal{L})}(a,t) \leq \tau_{\boldsymbol{r}(\mathcal{L})}(b,t)$ and $\tau_{\boldsymbol{r}(\mathcal{L})}(b,t) \leq \tau_{\boldsymbol{r}(\mathcal{L})}(a,t)$ hold. Thus, for any $t \in \mathbb{N}$ the equality $\tau_{\boldsymbol{r}(\mathcal{L})}(a,t) = \tau_{\boldsymbol{r}(\mathcal{L})}(b,t)$ is valid. Therefore, $\boldsymbol{r}(\mathcal{L}) \models a \boxed{=} b$ is valid too.

It is evident that this reasoning can be inverted and the inverse reasoning establishes that the validity of the assertion $\boldsymbol{r}(\mathcal{L}) \models a \boxed{=} b$ implies the validity of $\mathcal{L} \models a \boxed{=} b$.

Causality. Let $\mathcal{L}$ be an element of $\mathfrak{C}_{\text{lin}}(\mathcal{C})$ and $a \boxed{\preccurlyeq} b$ be a causality relation on $\mathcal{C}$ such that $\mathcal{L} \models a \boxed{\preccurlyeq} b$. Then there exists strict monotonic map $h : \mathcal{I}^b \to \mathcal{I}^a$ such that for each $i \in \mathcal{I}^b$ the condition $h(i) \preccurlyeq_{\mathcal{L}} i$ holds. The last causality ensures the validity of the inequality $\theta(i) \geq \theta(h(i))$ for each $i \in \mathcal{I}^b$. Lemma 3 ensures that the equalities $\chi_{\boldsymbol{r}(\mathcal{L})}(a,t) = |A_t|$ and $\chi_{\boldsymbol{r}(\mathcal{L})}(b,t) = |B_t|$ are true. Applying Lemma 4 one can obtain that $h(B_t) \subset A_t$. Taking into account that h is strict monotonic one can derive that

$$\chi_{\boldsymbol{r}(\mathcal{L})}(b,t) = |B_t| = |h(B_t)| \leq |A_t| = \chi_{\boldsymbol{r}(\mathcal{L})}(a,t)\,.$$

Therefore, the assertion $\boldsymbol{r}(\mathcal{L}) \models a \boxed{\preccurlyeq} b$ is true.

Conversely, let the assertion $\boldsymbol{r}(\mathcal{L}) \models a \boxed{\preccurlyeq} b$ be valid. It means that the inequality $\chi_{\boldsymbol{r}(\mathcal{L})}(a,t) \geq \chi_{\boldsymbol{r}(\mathcal{L})}(b,t)$ holds for all $t \in \mathbb{N}$. Our task is to build a strict monotonic map $h : \mathcal{I}^b \to \mathcal{I}^a$ such that $h(i) \preccurlyeq_{\mathcal{L}} i$. To solve the task let us enumerate elements of $\mathcal{I}^b$ and elements of $\mathcal{I}^a$ in ascending order: $\mathcal{I}^b = \{i_1, i_2, \dots\}$ and $\mathcal{I}^a = \{i'_1, i'_2, \dots\}$. Such enumerations are possible because $\mathcal{I}^b$ and $\mathcal{I}^a$ are well-ordered (see Theorem 2).

Let us define the map h by the formula $h(i_k) = i'_k$.

Firstly, note that if $\mathcal{I}^b$ is finite, i.e. $\mathcal{I}^b = \{i_1, \dots, i_N\}$, then by Lemma 3 we have

$$|\mathcal{I}^b| = \chi_{\boldsymbol{r}(\mathcal{L})}(b, \theta(i_N)) \leq \chi_{\boldsymbol{r}(\mathcal{L})}(a, \theta(i_N)) = |\{j \in \mathcal{I}^a \mid \theta(j) \leq \theta(i_N)\}| \leq |\mathcal{I}^a|\,.$$

Therefore, the map h is defined on the $\mathcal{I}^b$. By construction this map is strict monotonic.

Let i_k be an arbitrary element of $\mathcal{I}^b$ then

$$k = |\{i_1, \ldots, i_k\}| = \chi_{\boldsymbol{r}(\mathcal{L})}(b, \theta(i_k)) \leq \chi_{\boldsymbol{r}(\mathcal{L})}(a, \theta(i_k)) = |\{j \in \mathcal{I}^a \mid \theta(j) \leq \theta(i_k)\}| \,.$$

Hence, $i'_k \in \{j \in \mathcal{I}^a \mid \theta(j) \leq \theta(i_k)\}$ and $\theta(i'_k) \leq \theta(i_k)$. Further, we claim that $\zeta(i'_k) \preccurlyeq_{\mathcal{L}(\boldsymbol{r}(\mathcal{L}))} \zeta(i_k)$. Indeed, $\zeta(i'_k) = (\pi(i'_k), \theta(i'_k)) = (a, \theta(i'_k))$, $\zeta(i_k) = (\pi(i_k), \theta(i_k)) = (b, \theta(i_k))$, and $\theta(i'_k) \leq \theta(i_k)$ imply the required inequality. Taking into account the isotone property of ζ (see Theorem 5) we conclude that $i'_k \preccurlyeq_{\mathcal{L}} i_k$ and, hence, $h(i_k) \preccurlyeq_{\mathcal{L}} i_k$.

Precedence. Analysis of this clock relations kind can be performed similarly to analysis the previous kind of clock relations.

Let $\mathcal{L}$ be an element of $\mathfrak{C}_{\text{lin}}(\mathcal{C})$ and $a \boxed{\prec} b$ be a precedence relation on $\mathcal{C}$ such that $\mathcal{L} \models a \boxed{\prec} b$. Then there exists strict monotonic map $h : \mathcal{I}^b \to \mathcal{I}^a$ such that for each $i \in \mathcal{I}^b$ the condition $h(i) \prec_{\mathcal{L}} i$ holds.

We claim that $\tau_{\boldsymbol{r}(\mathcal{L})}(b, 1) = 0$. Indeed, if $\tau_{\boldsymbol{r}(\mathcal{L})}(b, 1) = 1$ then there exists $i \in \mathcal{I}^b$ such that $\zeta(i) = (b, 1)$. Taking into account $h(i) \prec_{\mathcal{L}} i$ we have

$$\zeta(h(i)) \prec_{\boldsymbol{r}(\mathcal{L})} \zeta(i) = (b, 1) \,.$$

It implies the validity of the inequality $\theta(h(i)) < 1$ but it is impossible. Hence, the equality $\tau_{\boldsymbol{r}(\mathcal{L})}(b, 1) = 0$ is valid.

Further, as it was shown above the inequality $\chi_{\boldsymbol{r}(\mathcal{L})}(b, t) \leq \chi_{\boldsymbol{r}(\mathcal{L})}(a, t)$ for all $t \in \mathbb{N}$.

We claim that if $\chi_{\boldsymbol{r}(\mathcal{L})}(b, t) = \chi_{\boldsymbol{r}(\mathcal{L})}(a, t)$ for some $t \in \mathbb{N}$ then $\tau_{\boldsymbol{r}(\mathcal{L})}(b, t+1) = 0$. If the equality $\chi_{\boldsymbol{r}(\mathcal{L})}(b, t) = \chi_{\boldsymbol{r}(\mathcal{L})}(a, t)$ holds then the equality $|\mathcal{I}^b(t)| = |\mathcal{I}^a(t)|$ follows from Lemma 3. Taking into account the strict monotonicity of h and Lemma 4 one can derive from the equality $|\mathcal{I}^b(t)| = |\mathcal{I}^a(t)|$ the equality $h(\mathcal{I}^b(t)) = \mathcal{I}^a(t)$. Now suppose that $\tau_{\boldsymbol{r}(\mathcal{L})}(b, t+1) = 1$ then there exists $i_0 \in \mathcal{I}^b$ such that $\zeta(i_0) = (b, t+1)$. Taking into account that $h(i_0) \prec_{\mathcal{L}} i_0$ we obtain $\zeta(h(i_0)) \prec_{\boldsymbol{r}(\mathcal{L})} \zeta(i_0) = (b, t+1)$ and, hence, $\theta(h(i_0)) < t+1$. Thus, we have $\theta(h(i_0)) \leq t$ and $h(i_0) \in \mathcal{I}^a(t) = h(\mathcal{I}^b(t))$. Therefore, we can assert that there exists some element i_1 of $\mathcal{I}^b(t)$ such that $h(i_1) = h(i_0)$. Taking into account that $i_0 \notin \mathcal{I}^b(t)$ one can conclude that $i_1 \neq i_0$. Hence, we have obtain the contradiction that proves the equality $\tau_{\boldsymbol{r}(\mathcal{L})}(b, t+1) = 0$. Therefore, the assertion $\boldsymbol{r}(\mathcal{L}) \models a \boxed{\prec} b$ has been proved.

Conversely, a check of the assertion "$\boldsymbol{r}(\mathcal{L}) \models a \boxed{\prec} b$ implies $\mathcal{L} \models a \boxed{\prec} b$" can be gotten by an almost verbatim repeat of the similar checking for the causality relations □

Corollary 3. *Let be a finite set of clock relations on $\mathcal{C}$ then the following assertions are true:*

1. *the belonging $\mathcal{L} \in [\![S]\!]_{LDM}$ implies the belonging $\boldsymbol{r}(\mathcal{L}) \in [\![S]\!]_{OM}$ for all $\mathcal{L} \in \mathfrak{C}_{\text{lin}}(\mathcal{C})$;*

2. *the belonging* $\boldsymbol{r} \in [\![S]\!]_{OM}$ *implies the belonging* $\mathcal{L}(\boldsymbol{r}) \in [\![S]\!]_{LDM}$ *for all* $\boldsymbol{r} \in \mathfrak{R}(\mathcal{C})$.

Proof. Suppose that $\mathcal{L} \in [\![S]\!]_{LDM}$ and $a \boxed{*} b$ is an arbitrary element of S . It means that the assertion $\mathcal{L} \models a \boxed{*} b$ is valid. Hence, the theorem ensures the validity of the insertion $\boldsymbol{r}(\mathcal{L}) \models a \boxed{*} b$. Thus, $\boldsymbol{r}(\mathcal{L}) \in [\![S]\!]_{OM}$ is true.

Now suppose that $\boldsymbol{r} \in [\![S]\!]_{OM}$ and $a \boxed{*} b$ is an arbitrary element of S . It means that the assertion $\boldsymbol{r} \models a \boxed{*} b$ is valid. Taking into account that $\boldsymbol{r}(\mathcal{L}(\boldsymbol{r})) = \boldsymbol{r}$ (see Theorem 5) and using the theorem one can conclude that $\mathcal{L}(\boldsymbol{r}) \models a \boxed{*} b$. Thus, $\mathcal{L}(\boldsymbol{r}) \in [\![S]\!]_{LDM}$ □

6 Equivalence of Denotational and Operational Semantic Models

Taking into account that is set forth above one can prove the main theorem of the chapter.

Theorem 7 (Identity of Denotational and Operational Closures). *Let* $\mathcal{C}$ *be a finite set of clocks,* S *be any finite set of clock relations on* $\mathcal{C}$ *then the following equality holds*

$$[\mathrm{S}]_{DM} = [\mathrm{S}]_{OM} \ . \tag{10}$$

Proof. Let $a \boxed{*} b$ be an arbitrary element in $[S]_{DM}$. Then using Corollary 1 we conclude that $a \boxed{*} b \in [S]_{LDM}$. Let us take an arbitrary run $\boldsymbol{r}$ in $[\![S]\!]_{OM}$ then Corollary 3 ensures that $\mathcal{L}(\boldsymbol{r})$ is an element of $[\![S]\!]_{LDM}$. The assertion $\mathcal{L}(\boldsymbol{r}) \models a \boxed{*} b$ follows from the belonging $a \boxed{*} b \in [S]_{LDM}$. Reusing of Corollary 3 ensures that the assertion $\boldsymbol{r} \models a \boxed{*} b$ is valid. Thus, the assertion $a \boxed{*} b \in [S]_{OM}$ is valid too. This reasoning allows to assert that the embedding $[S]_{DM} \subset [S]_{OM}$ is true.
Conversely, suppose that $a \boxed{*} b$ is an arbitrary element in $[S]_{OM}$. Let us take an arbitrary run $\mathcal{L}$ in $[\![S]\!]_{LDM}$ then Corollary 3 ensures that $\boldsymbol{r}(\mathcal{L})$ is an element of $[\![S]\!]_{OM}$. The assertion $\boldsymbol{r}(\mathcal{L}) \models a \boxed{*} b$ follows from $a \boxed{*} b \in [S]_{OM}$. Reusing of Corollary 3 ensures that the assertion $\mathcal{L} \models a \boxed{*} b$ is valid. Thus, the assertion $a \boxed{*} b \in [S]_{LDM}$ is valid too. Taking into account Corollary 1 we obtain the validity of $a \boxed{*} b \in [S]_{DM}$. This reasoning completes proving of the embedding $[S]_{OM} \subset [S]_{DM}$.

Therefore, the equality $[S]_{DM} = [S]_{OM}$ holds □

Corollary 4. *Let* $\mathcal{C}$ *be a finite set of clocks,* S *be any finite set of clock relations on* $\mathcal{C}$*, and* $a \boxed{*} b$ *be some clock relation on* $\mathcal{C}$ *then*

$$\mathrm{S} \Vdash_{DM} a \boxed{*} b \textit{ is satisfied if and only if } \mathrm{S} \Vdash_{OM} a \boxed{*} b \textit{ is satisfied.} \tag{11}$$

Proof is trivial □

7 Conclusion

In summary we can assert that analysis of two approaches to the determination of CCSL clock relations meaning, more precisely, the denotational approach proposed by F. Mallet in [10] and the operational approach proposed by C. André in [1], has allowed to establish the equivalence of the approaches. This equivalence means the identity of the corresponding closures for any relational CCSL specification. This results has been obtained by refining of the denotational and operational semantic definitions.

In the case of the denotational semantics we have described the special class of mathematical objects called chronograms. Studying properties of chronograms has given a possibility to distinguish a special subclass of this class. Chronograms of this subclass has been called linear chronograms. It has been established that the usage of the subclass of linear chronograms is sufficient for determining the denotational semantics.

In the case of the operational semantics we have refined the system of invariants to determine sets of admissible runs for each kind of clock relations.

These refinements have formed the base for proving the main chapter result about the equivalence of the denotational and operational semantics.

Further research should answer the following questions:

- whether it is possible to expand our main result for all constructions of CCSL;
- whether it is possible to build an axiomatic basis for CCSL clock relations and other CCSL constructions.

Studying state-charts for modeling CCSL constructions (see [15]) seems to be interesting also.

Acknowledgement. Authors express gratitude to participants of Workshop "Mathematical Foundations of Formal Software Verification", which has been organized by the Department of Theoretical and Applied Computer Science at V.N. Karazin Kharkiv National University. Especially, we are thankful to Prof. Boris Novikov and Lecturer Bogdan Chernomaz for the constructive discussions on this work.

References

1. André, C.: Syntax and Semantics of the Clock Constraint Specification Language (CCSL). Technical report, RR-6925, INRIA (2009), `http://hal.inria.fr/inria-00384077/en/`
2. André, C., Mallet, F., de Simone, R.: The Time Model of Logical Clocks available in the OMG MARTE profile. In: Shukla, S.K., Talpin, J.-P. (eds.) Synthesis of Embedded Software: Frameworks and Methodologies Correctness by Construction, pp. 201–227. Springer Science+Business Media, LLC, New York (2010)
3. Baer, J.-L.: Multiprocessing Systems. IEEE Trans. on Computers C-25(12), 1271–1277 (1976)

4. Benveniste, A., Caspi, P., Edwards, S.A., Halbwachs, N., Le Guernic, P., de Simone, R.: The synchronous languages 12 years later. Proc. IEEE. 91(1), 64–83 (2003)
5. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: Fog computing and its role in the internet of things. In: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, pp. 13–16. ACM, New York (2012)
6. Börger, E., Stärk, R.: Abstract State Machines: A Method for High-Level System Design and Analysis. Springer, Heidelberg (2003)
7. Le Guernic, P., Talpin, J.-P., Le Lann, J.-C.: Polychrony for system design. Journal of Circuits, Systems, and Computers 12(3), 261–304 (2003)
8. Harzheim, E.: Ordered Sets. Springer Science+Business Media, Inc., New York (2005)
9. Hoare, C.A.R.: Communicating Sequential Processes. Prentice Hall International (1985)
10. Mallet, F.: Logical Time @ Work for the Modeling and Analysis of Embedded Systems, Habilitation thesis. LAMBERT Academic Publishing (2011)
11. Milner, R.: Communicating and Mobile Systems: The Pi Calculus. Cambridge University Press, Cambridge (1999)
12. Information technology – Object Management Group – Object Constraint Language (OCL). ISO/IEC 19507:2012(E)
13. Nielsen, M., Plotkin, G., Winskel, G.: Petri nets, event structures and domains. Theor. Comp. Sc. 13(1), 85–108 (1981)
14. Alexander, M., Gardner, W.: Process Algebra for Parallel and Distributed Processing. CRC Press (2009)
15. Romenska, Y., Mallet, F.: Lazy Parallel Synchronous Composition of Infinite Transition Systems. In: Ermolayev, V., et al. (eds.) ICT in Education, Research and Industrial Applications: Integration, Harmonization and Knowledge Transfer, Proc. 9th Int. Conf. ICTERI 2013, vol. 1000, pp. 130–145. CEUR-WS (2013)
16. UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded Systems. OMG (2011), `http://www.omg.org/spec/MARTE/1.1/pdf/`
17. OMG Unified Modeling Language^{TM}(OMG UML), Infrastructure. OMG (2011), `http://www.omg.org/spec/UML/2.4.1/Infrastructure`
18. OMG Unified Modeling Language^{TM}(OMG UML), Superstructure. OMG (2011), `http://www.omg.org/spec/UML/2.4.1/Superstructure`

How Do Computer Science Students Use Distributed Version Control Systems?

Michael Cochez, Ville Isomöttönen, Ville Tirronen, and Jonne Itkonen

Department of Mathematical Information Technology,
University of Jyväskylä,
P.O. Box 35 (Agora), 40014, Jyväskylä, Finland
{michael.cochez,ville.isomottonen,ville.tirronen,jonne.itkonen}@jyu.fi

Abstract. The inclusion of version control systems into computing curricula enables educators to promote competences needed in real-life situations. The use of a version control system also has several potential benefits for the teacher. The teacher might, for instance, use the tool to monitor students' progress and to give feedback efficiently. This study analyzes how students used the distributed version control system Git in several computing courses. We analyzed students' commit log data in two advanced programming courses, a second-year introductory software engineering course, and two courses where students developed software products. This enables us to compare Git usage between introductory level and master's level students, and between exercise-driven and product-driven courses. We found out that students which are using the version control system in a software product development setting used it in a more graceful manner. The students which were further given introduction to branching in the system also used this to not have to wait until the practical session to commit their changes. We also found the amount of garbage in the repositories is strongly relayed to the students' awareness of the version control process and the need of keeping the workspace clean.

Keywords: Programming Education, Version Control System, Git.

1 Introduction

Version control systems (VCSs) have a decades-long history in professional software engineering with early systems like Source Code Control System (SCCS) and Revision Control System (RCS) developed in the seventies and eighties, respectively. These pioneering systems only supported storage of the versions on the file system, while later systems also allowed for remote and mostly centralized storage of the versions. The most well-known centralized systems are Concurrent Versions System (CVS) and Subversion (SVN). Currently, there is a trend towards the use of distributed version control systems (DVCS) where each user has a local copy of the repository which can be synchronized with other repositories. Systems such as Git and Mercurial exemplify this type of present-day decentralized technology. DVCSs enable flexible change tracking, reversibility, and manageable collaborative work, which are valuable properties for both small and large projects.

There are many arguments for incorporating VCSs into an educational setting. From a teacher's point of view, using VCSs increases possibilities for monitoring how students make progress with their assignments and eases the feedback process. The teacher could, for instance, include corrections and suggestions directly into the students' program code [2]. More generally, educators acknowledge that the use of VCSs relates to effective team work, and that it is a crucial skill to be taught to prepare a competent workforce for present-day distributed workplaces [3].

An educational concern of interest to us is how students actually use VCSs. This has been previously studied by Mierle et al. [4], who hoped to find a correlation between an effective use of VCSs and study success among second-year students. The authors found no clear patterns in the data, which they attributed to the fact that beginner students climb their learning curve at different rates; see [4, 5]. These authors call for more research on VCS usage patterns in particular in upper-year courses [5], which provides a key motivation for the present study.

We initially collected data about the students' use of the distributed version control system (Git) from three different courses: *Introduction to Software Engineering* (second-year bachelor), *Functional Programming* (master's level), and *Service oriented architectures and cloud computing for developers* (master's level). The latter two are advanced programming courses. The results of the analysis were published in ICTERI 2013, with the main conclusions being that if a VCS is adopted as a course management tool, the students are likely to use it simply as such a tool. This was indicated, for instance, by the fact that there tended to be one dominant committer in the groups instead of the effective group-wide use of the system. By comparing the advanced programming courses to the second-year SE course, where students were given an introduction to Git usage, it was also concluded that training can promote a more shared adoption and professional use of VSCs in group work settings.

The present study extends our ICTERI 2013 article by including data from two additional courses where students work for a software product instead of programming exercises. These are the *Bachelor Project* and *Design of Agent Based System*. Through the inclusion of data from this different setting, we aim to refine our previous findings and raise informed questions for this area where more research is needed. We principally study the student use of VCS quantitatively by exploring version control commit frequencies, commit sizes, and the activity of individual students, while we also evaluate student commit messages regarding how informative they are. We also look into the what kind of files students store into their repositories.

2 Version Control Systems in Education

Clifton et al. [6] summarize that in educational settings VCSs have been adopted to promote realistic software development experiences [7], to monitor or visualize team and individual contributions [8], and to manage non-code artifacts such as creative writing [9]. Clifton et al. themselves, as well as many others, use a VCS for course management purposes. Further, some authors regard VCSs as a valuable tool to monitor and understand *how* students develop code [10]. Unsurprisingly, one of the most usual educational targets appears to be project-based courses where VCSs both foster team work and facilitate course management tasks such as assessment and grading [11]. Milentijevic et al. [12] go as far as to propose a generalized model for the adoption of VCSs as support in a variety of project-based learning scenarios. All in all, we find that there is a general consensus of the benefits of VCSs as an integral part of computing curricula, with one key argument being that they measure up to the requirements of globally distributed workplaces [3].

There are also challenges in the educational use of VCSs. Reid and Wilson [5], who used the CVS system, report on the confusion in judging which of the students' assignment versions was the final one. Glassy [10] found that students tend to put off working on assignments for as long as possible, even though a VCS is proposed to them with the hope of iterative work processes. Issues of this kind relate to inefficient use of VCSs. Furthermore, Reid and Wilson [5] noticed that some students mixed the functionalities of the CVS check out and update commands, and that also teaching assistants encountered problems if they had not properly familiarized themselves with the tool. These issues were considered to be due to the lack of a mental model of the VCS system used. Yet another challenge Reid and Wilson [5] raise is increased teacher workload when repositories are initiated and managed by teachers. In a more recent study, Xu [11] points out that there can be a long and rough learning curve before students feel comfortable using Git. Accordingly, Milentijevic et al. [12], who used CVS, report that students find a VCS to be a useful tool after they are sufficiently familiar with it. In the paper by Glassy [10] and Xu [11], the value of informative commit log messages is raised as a topic to be emphasized to the students. Rocco and Lloyd [13] in turn observed that some student have difficulties in understanding what constitutes "a significant change" to be committed.

It is much more difficult to find systematic empirical studies on issues such as how frequently students make commits and how they share the work. Rocco and Lloyd [13] found in their data that over 80% of a CS1 course population could adopt an iterative work process with the Mercurial system (50.0% did 7–21 commits and 33.3% more than 21 commits). On another course the authors defined a minimum commit frequency for one assignment and no requirements for the assignment that followed. With the first assignment, 75% of the students obtained a reasonable commit frequency, while with the latter this was 81%, altogether indicating that informing students of proper VCS usage can have a positive effect on their work processes. The authors note that not only were the students able to grasp the basics of the VCS (Mercurial), but they tended to continue to take advantage of the tool later on.

The present study focusing on the students' usage patterns with the Git system in both bachelor's level and master's level courses complements the studies such as the ones by Rocco et al. [13] and Mierle et al. & Reid and Wilson [4, 5].

3 The Courses

This section describes five courses in which Git system was used. The first is a bachelor's level introductory course on software engineering, the following two are master's level programming courses, and the last two are courses where students work on a project assignment.

Introduction to Software Engineering (SE) is a 3-credit bachelor's level course consisting of lectures, a course assignment, and an end-of-course exam. The lectures introduce students to the basic concepts of software engineering, while the mandatory course assignment is the preparation of a project plan. The assignment is done in small groups and consists of four larger phases that need to be accepted by the lecturer. Mandatory supervision sessions on version control were arranged at the beginning of the course in order to encourage all the students to use the distributed version control system Git for the group assignment. The course had altogether 72 students in 33 groups (2.18 ± 0.76 students per group).

Functional programming (FP) is a 6-credit master's level course implemented without traditional lectures and exams. The course is run in week-long cycles such that each week a new set of exercises is announced for the students. Students work in small groups and all of their study time is devoted to programming the weekly exercises. Two contact sessions are held each week. The first one is devoted to supporting the students' work and answering their questions. During the second weekly contact session there is a review of the student-written code. Overall, the course emphasizes self-direction on the part of students, similar to recently discussed course models such as the flipped classroom; see more details in [14–16]. Git was proposed for students as their primary group work tool and all of the exercises had to be returned via it. Thirty-six students where active in the course divided over 13 groups. (2.77 ± 0.89 students per group).

The other master's level programming course studied, *Service oriented architectures and cloud computing for developers* (SOA&CC), introduces students to the use of digital services and the concept of cloud computing. A format similar to the FP course is used during the first (5 credits) part of the course. During that part of the course students undertake independent group work on a set of assignments each week. Two weekly sessions are arranged for the group work and one mandatory contact session focusing on reflective program review is arranged at the end of each week. An analysis of how the course model used in this course attempts to motivate students can be found in [17]. During the course, Git is not only used as a version control system; it is also used as a tool to deploy code to Platform as a Service (PaaS) providers. Nine groups of students were formed with altogether 36 students (4 ± 0.82 students per group).

Design of Agent-Based Systems (Agent) is a five-credit course in which students get acquainted with agent systems as a novel software development paradigm. This course follows after another one which gives the students a theoretical foundation in agent systems. During the course students learn what an agent-based system is by implementing its basic components, learn what their system is capable of by using it in a simple scenario, and get an overview of how agent-based systems compare and connect to other technologies. For the development of their platform, students use Git as a collaboration tool; both in the group and with the course teacher. At the beginning of the course, each students had to perform a preliminary task with the Git version control system in a temporary repository. Concretely, there was a simulated scenario where two development branches (their own and the teacher's) had diverged from each other. The students' task was then to merge these two lines of development into one. In this paper, we use data from the group repositories from the spring 2013 offering in which 2 groups of 3 and 2 groups of 4 students participated.

Bachelor project (Project) is a 5-credit course where students innovate and implement there own software product under the general theme of open data. The students work in the groups of four students with the main learning goal being the explicit recognition of group processes (see e.g. [1]) and software development processes. Moreover, the students reinforce their programming skills and gain an authentic experience of programming-in-the-many using a version control system. Only a starting lecture and one topic lecture on group processes and software processes are provided at the beginning of the course, while the remaining of the teaching resources are expended on intensive per-group dialog sessions arranged on weekly basis. Students are graded with pass/fail, self-evaluations and shared discussions providing the main pedagogic instruments. During the starting lecture, the students were asked if they were already familiar with Git, and thus whether they needed VCS training for their project. All the students agreed that they can grasp the basics and learn more within their group, and no specific training was arranged. So far all the student groups have used Git as their VCS. It should also be noted that in this project-based course VCSs are used purely as a group work tool as opposed to a submission system. The data we refer to in this paper originates from the spring 2013 offering including 4 four-person groups and 2 three-person groups.

The first four of the above-mentioned courses utilize the Faculty's *YouSource*[1] system. Similar to staff members, students can use their university credentials to log in to this system and create projects and Git repositories to manage collaborative work. The projects and repositories can be defined to be either private or public and collaborators can be added to them with a variety of permissions. This system has been in use at the department since mid-2010 and has been used in many courses and research projects. Already before the YouSource system was conceptualized the Git system was used without a centralized management interface. Regarding the Bachelor project course, students are guided to take advantage of their preferred Git environment, which has resulted in some

[1] `https://yousource.it.jyu.fi/`

of the groups using GitHub[2] or Bitbucket[3] as an alternative to or in addition to YouSource.

It should be noted that in the remainder of this paper we are specifically concerned with the Git version control system, which belongs to the third generation of version control systems (DVCSs). Students are free in their choice of environment for interacting with the version control system. Students can for instance use the *Git* command line tool, tools with a graphical user interface, or tools included in their integrated development environments.

4 Data Analysis

The Git repositories created by students or course teachers for the respective courses are the source of all data analysis in this paper. One limitation of this data source is that we cannot see any data related to branches which a student did not push to the central Git server. However, if work in one of these so-called local branches got merged into a branch synchronized with the central Git server, we are able to see its history as well. Further, this limitation is of minor importance since we are mainly interested in how students use the version control system in group work settings. Another limitation, which is inherent to the Git DVCS, is that we cannot know for sure whether time stamps on commits are truthful. It is technically possible to tamper with the date of the commits, but since there is no benefit for students to do so, we make the assumption that the time stamps are correct.

To study students' usage patterns, we will perform five different analyses, the first four of which are based on commits to the repository and the last one on the content of the repositories. For each commit we extracted the number of insertions and number of additions in accordance with the short status log of the commit[4]. We added these two numbers together to form what we will call the number of changes of the commit. The tools used in the analysis have been developed by the authors of the paper and consume output produced by the diverse Git commands.

For the first analysis, we will, for each course, look at the commit activity over the whole course. To be concrete, we will visualize the commit activity by plotting the estimated probability density function of the total number of changes, i.e. for all students, over the span of the course. The density is estimated via the standard kernel density estimator, using a Gaussian kernel with the bandwidth of 6 hours [18]. The height of the plot then shows the relative likelihood of a commit at a specific point in time. We also calculated how intensive the students used the system close to sessions, i.e., starting from three hours before until three hours after the session. What we measure is the amount of commits and total number of changes of commits during these periods in comparison to these amounts outside of the periods.

[2] `https://github.com/`
[3] `https://bitbucket.org`
[4] `http://www.kernel.org/pub/software/scm/git/docs/git-log.html`

The second analysis focuses on students' activity during the implementation sessions. This is done only for the FP, SOA&CC, Agent courses since the remaining two do not have distinct sessions during which students get time to implement their work. We use a similar method as in the first analysis, but accumulate all commits that were made during the implementation sessions in the same plot. This plot shows when the students commit their code during the contact sessions. In the figure the far left of the x-axis represents the start of the session and the far right 15 minutes after the end. This is done in order to account for commits right after the sessions. In this case we use a bandwidth which is one tenth of the total length of the session. It should be noted that the classrooms do not always have enough computers such that each student can work independently. However, we do not think that this factor contributes significantly since many students tend to bring their own laptop to the sessions or work from home. Further, the teachers noticed that, likely because of the group setting, many groups gathered around one computer. During most sessions there were computers unused in the classroom.

Thirdly, we perform an analysis of the commit messages in the different courses by classifying them in three categories: useful, trivial, and nonsense. A message is placed in the nonsense category if its content is not anyhow related to what is being committed. An example of this type of messages are these which contain only a couple of random letters, needed because the Git system does not allow for empty commit messages. A trivial message is one which has no information beyond what is immediately visible from the commit meta-data. This category includes, for instance, a message consisting of a list of changed files or one saying that a given commit is a merge of two branches. All other commits are classified as useful. It should be noted that being in the useful class does not directly imply that the message is of high quality. It only means that the message is not trivial or nonsense. The classification was done manually by the respective teachers of the courses. We do not try to make a comparison between the courses, because the bias caused by having different raters is difficult to estimate.

By the fourth analysis, we try to measure whether the version control system is used equally among the students in the group. If the system would be used by all the students in a group, we would expect that the most active student in a group of n students performs $(1/n) * 100\%$ of the commits. To represent this number for all groups in the different courses we first find the students with the highest number of commits in their respective groups. Then we calculate their individual share in the total number of commits of their group. We then create an overview of the obtained percentages where we show a chart for each group size since comparison among unequal group sizes would lead to biased results. It only makes sense to measure this for groups with more than one person. The SE course had a few single-person groups, hence only 28 groups from that course are included.

In the fifth and last analysis, we investigate the types of files which students put under version control. First, teachers of each of the courses listed the file types and limits which they would expect a normal repository to contain. We started out from the files included in the HEAD of the master branch. For all the courses we first determined the type of each file using the BSD *file*[5] command. However, the SE and Agent repositories required a manual analysis to decide the type of the files because the *file* command was unable to distinguish between the file types in use in the course. Then we counted the number of files of each file type. Then for each count, we compared it to the number of files expected by the teacher and any surplus was counted as garbage. The final number calculated for each group is the fraction of garbage in the total number of files.

5 Results

This section describes the results of our analyses. The first subsection shows the results for the analysis of the student activity during the whole course. In the second subsection, we focus on the implementation sessions only. The results of the analysis of the commit messages is shown in subsection three. Then we consider the activity distribution among students in the fourth subsection. Lastly, we look at the types of files which students submit to the version control system.

5.1 Commit Activity over the Whole Course

The student activity in the SE, FP, SOA&CC, Agent, and Project courses is presented in the Figures 1, 2, 3, 4, and 5, respectively. We will first discuss the results for the SE and Project course since both courses had a focus on long term planning. In the SE course, we draw thick vertical lines to indicate the end of each of the four phases of the course assignment, and in the Project course, they represent the deadlines given to the student groups. As can be seen in the graph for the SE course (Figure 1), there seems to be no correspondence between these deadlines and the student activity, whereas such correspondence is clearly visible in the graph for the Project course (Figure 5). In the SE course students appeared to commit rather evenly throughout the course, which we originally attributed to the VCS training given in the course. We also attributed the activity spike at the start of this course to the students trying out and getting familiar with the version control system, which was required of them in the early training sessions. Unlike in the SE course, in the Project course, activity spikes can be seen just prior to the deadlines of the course and especially at the (more important) second deadline. The last deadline was only for final bug fixes. Regardless of the spikes, the graphs show constant activity after the first third of the course, which seems to indicate proper use of the version control system. The less active period in the early part of the course indicates a period when the students are familiarizing themselves with their topics and planning the project work, and the fact that may students had difficulties in adapting to intensive practical work.

[5] `http://www.openbsd.org/cgi-bin/man.cgi?query=file`

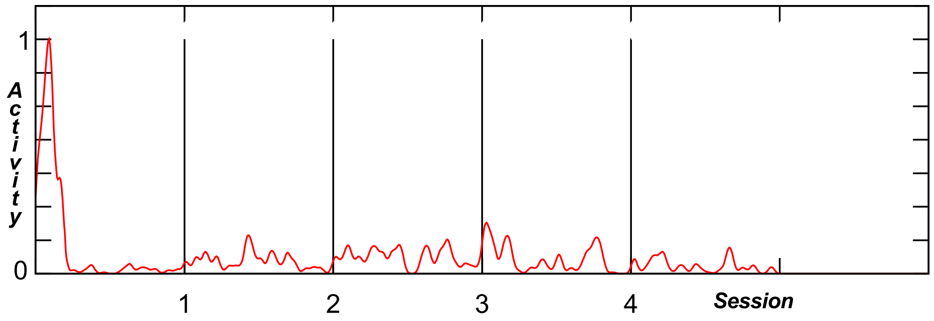


Fig. 1. Commit activity during the SE course

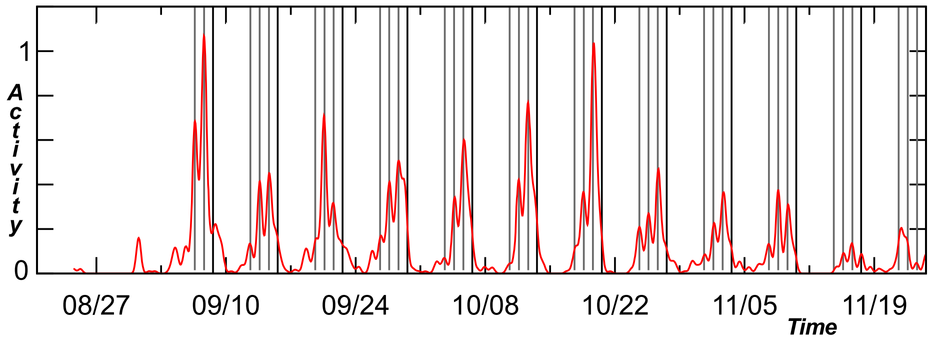


Fig. 2. Commit activity during the FP course

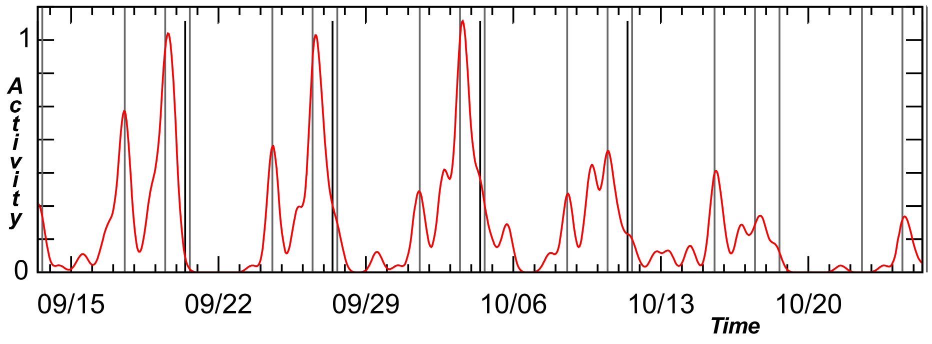


Fig. 3. Commit activity during the SOA&CC course

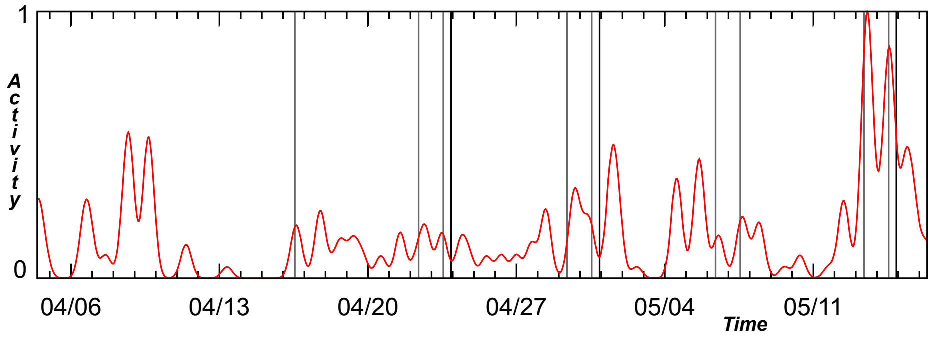


Fig. 4. Commit activity during the Agent course

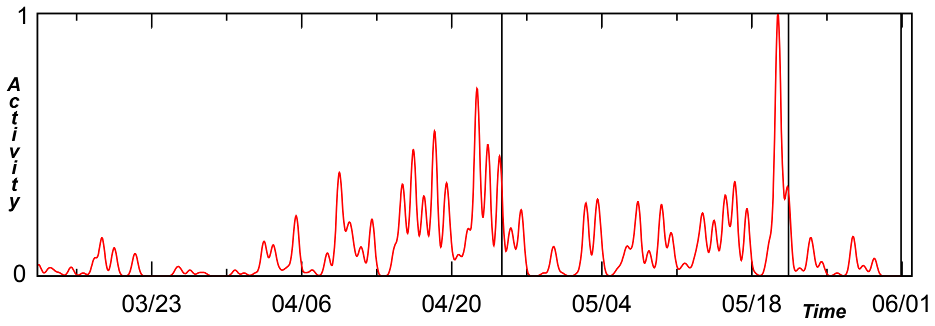


Fig. 5. Commit activity during the Project course

Table 1. Percentage of the commits and percentage of the total commit volume happening within three hours from the sessions

	FP	SOA&CC	Agent
number of sessions	35	20	9
total number of commits	1848	446	248
number of commits around session	916	192	49
percentage of commits around sessions	50%	43%	20%
total number of changes	165975	1061268	20109
number of changes around session	53941	309433	4186
percentage of changes around sessions	32%	29%	21%

In the graphs of the FP, SOA&CC and Agent courses (figure 2 and 3), we indicated with thin vertical lines the sessions during which students get time in the classroom to work on the assignments. The thicker vertical lines indicate deadlines for the weekly assignments. The dates of the sessions are displayed on the x-axis in a month/day format. In contrast to the SE course, these graphs show a closer correlation between student activity and the implementation sessions and the deadlines. The graphs suggest that a significant amount of the work was committed during the contact sessions, which again suggests that students bring their work to the sessions to be committed there. It is also clearly visible that the students have a very low activity during the weekends.

When we calculate the percentages of commits and the percentages of the total commit size in a period starting from 3 hours before and ending 3 hours after the sessions for the courses with sessions, we get the numbers as show in table 1. What we notice is that in all courses a significant amount of activity is focused around the sessions, which prompts us to study the student behavior during the sessions separately in the next section. We also notice from the table that the activity in the Agent course was, both in number of commits and number of changes, much less concentrated around the sessions as in the two other courses. There are at least potential factors which could explain this, namely the fact that the course had much less sessions as the other courses forcing students to do part of the work outside the sessions or the fact that students did get introduction exercises to the git version control system after which they were not afraid of making commits without others consent.

5.2 Commit Activity during the Sessions

In the FP and SOA&CC courses students were clearly more active during sessions than at other times. Also the graph for the Agent course (Figure 4) shows an activity peek during the sessions. The graphs in figures 6, 7, and 8, show the students activity during the sessions and 15 minutes after the session. We normalized the duration of the session (90 minutes) and the 15 minutes overtime between zero and one. Interestingly, we notice a similar behavior in all three courses. There seem to be three periods of higher activity. The first moment of higher activity is in the beginning of the session after about 10 minutes. The second one, which last longer, is between 20 and 40 minutes after the start of the session and lastly, the activity peaks that appear at the end, or shortly after the session.

The first period of activity is most likely because individual students have been implementing parts at home. These students then decide to commit only after receiving consent from other group members. This is an indication that students do not know how to use the version control system efficiently, as in principle they could have used a separate branch for their local development and merged their changes to their shared version of the exercises. Also, speculating based on student dialog, some students might have feared ‘losing face’ by making their preliminary versions visible to others, including the teacher.

During the second period of activity students are using the system as they are supposed to, committing changes regularly. Then the activity drops for quite some time before reviving shortly after the session. We attribute this last peak to those groups who have been working during the whole session without committing many changes. At the end of the session they want to store their work for later continuation and decide to put all their work in the system.

Notice that also the Agent course, despite the lower percentage of commits during the sessions, shows a similar pattern.

5.3 Commit Message Analysis

The classification of the commit messages was performed for all courses and yielded the results shown in table 2.

Table 2. Categorization of the commit messages per course

	useful	trivial	nonsense
SE	996 (67%)	430 (29%)	59 (4%)
FP	1422 (78%)	276 (15%)	129 (7%)
SOA&CC	289 (74%)	65 (17%)	37 (9%)
Agent	211 (85%)	36 (14%)	2 (1%)
Project	845 (72%)	303 (26%)	29 (1%)

In an ideal repository we would not find any trivial and nonsense messages. What we see from the table however is that in certain courses there is a significant amount of these types of messages.

As can be seen in the table, the amount of nonsense messages in the repositories of the courses in which a product is created, i.e., the Project and Agent courses, is significantly less than in the other courses. Concrete, the history of the repositories contain about 1% nonsensical messages whereas these account for between 4 and 9 percent in the other courses. From these higher figures, the lowest is software engineering where students where introduced to the proper use of the version control system. Interestingly there were no lectures devoted to the use of the version control system in the Project course but the amount of nonsense was still very low. This might be due to the fact that a significant amount of the students had earlier attended the SE course and likely picked up the habit of creating sensible messages. Another factor which we suspect to play a role is the fact that students have a stronger feeling of ownership over the repository and act hence more responsible.

It is not visible from the table, but the teachers performing the classification shared the opinion that the messages in the useful category where not all that descriptive. Some commit messages could be regarded as 'locally sensible',

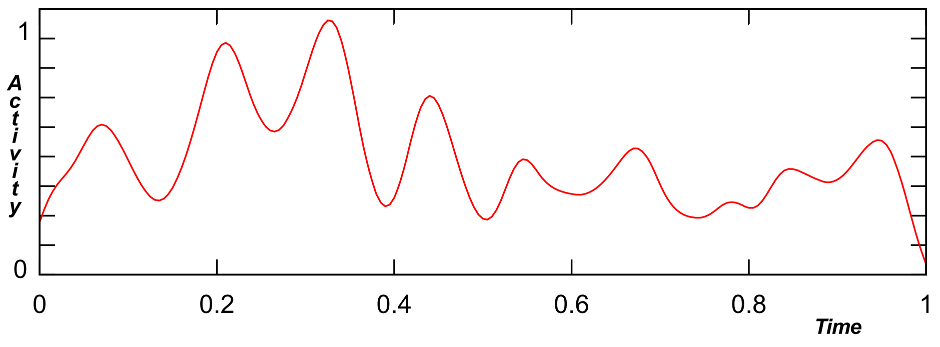


Fig. 6. Commit activity during the sessions of the FP course

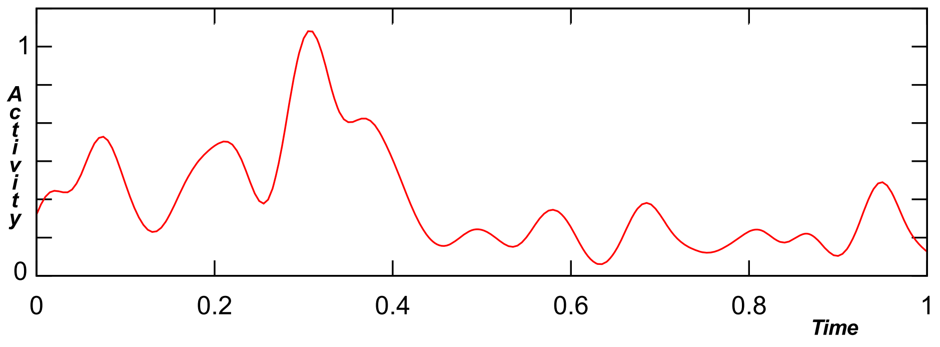


Fig. 7. Commit activity during the sessions of the SOA&CC course

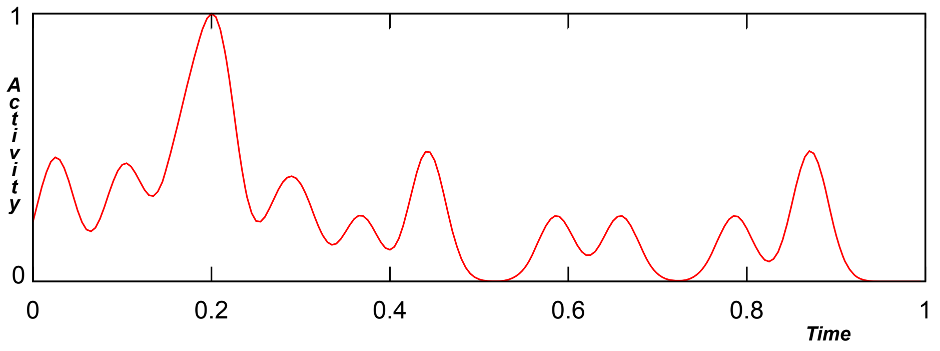


Fig. 8. Commit activity during the sessions of the Agent course

meaning that they could be useful for communication during a short time span, but offer not much for later inspection. Many of the commit messages are clear indicators that the students regard the system mainly as a submission system. Examples include “Answer for week 12” and “exercise 4a”. We also noticed some messages related to problems in using the Git system. The amount was however not as significant as the teacher had expected. It is also observed that the quality of the messages is depending on the group, indicating that some groups use the messages for communicating, while others do not. Moreover, in the Agent and Project course the teachers found regularly very long commit messages (more as 100 words) which explained in detail why certain design decisions were made. From these messages it became clear that in these groups the version control system was used also as a mean of communicating with each other.

5.4 Differences in Student Activity

To show the differences in activity among students we assembled the charts in figure 9. The figure contains one chart for each group size. In each chart there is a mark representing the percentage of commits performed by the most active committer for each group. The mark has a different shape depending on which course the group belongs to. A gray vertical bar indicates the expected percentage if each group member would commit an equal number of times. The height of the shapes on the vertical axis is only for representational purposes and hence not significant.

What we are measuring is not the activity level in the sense of how active a student is in a course. Instead, we measure how active the students are in using, or more precisely committing to, the version control system. One should be careful not to confuse the number of commits performed by students with their activity levels because the content of the commits can vary. Not even measuring the size of the commits would give a better measure for student activity since, for instance, fixing a bug might only induce a small change in the code, but require a significant amount of effort to find.

What we see from the charts is that the most active committer in a group, most of the times, commits significantly more often than the expected percentage. Put another way, the most active committer in each group is very often much more active user of the system as the average which one might expect. Interestingly, even in the Agent and Project courses we do see the same phenomenon. This can be due to that student having a dominating role in the group. In the FP, SOA&CC and Agent courses we tried to mitigate this effect by grouping students according to their skill level.[14, 16, 17] From the observations in the class we however think that the main differences are caused by a different level of familiarity with the version control system between the group members. The person with the most experience will commit more frequently or is given the task of submitting the work of others to the system.

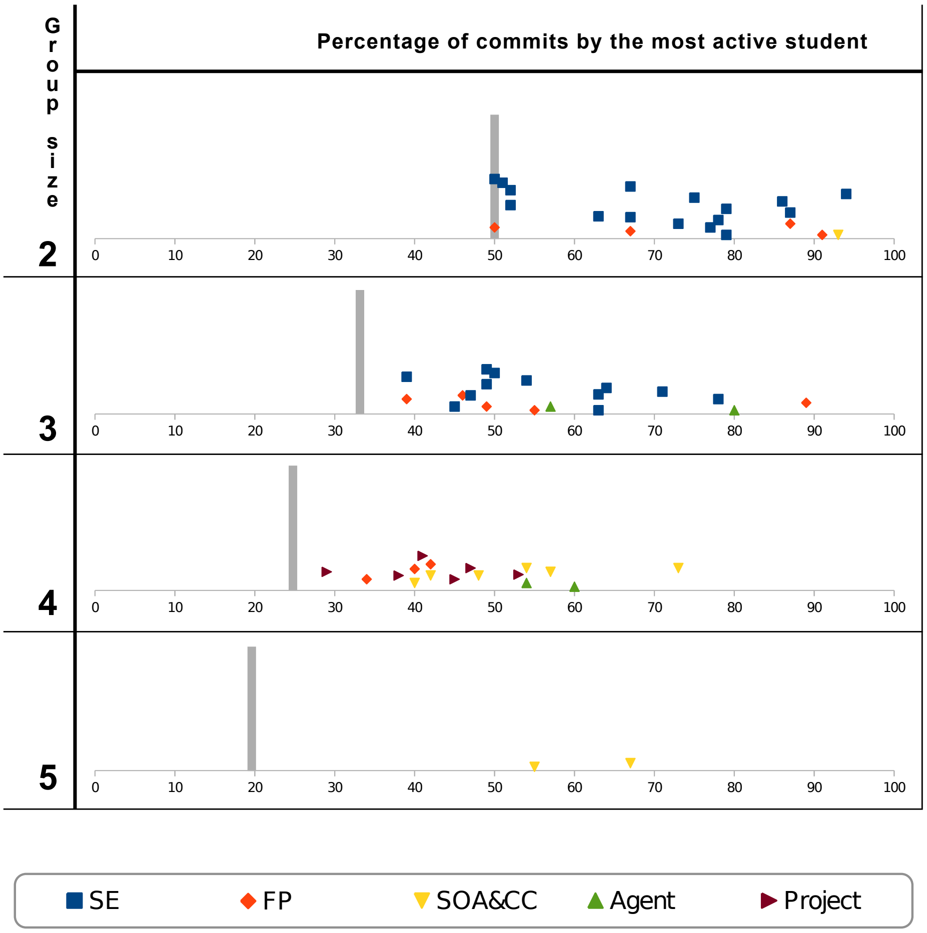


Fig. 9. Percentage of commits performed by the most actively committing student of each group. Each chart displays this percentage for groups of a given size. The gray bars indicate the percentage which would be obtained if every student in a group with the given size would commit the same number of times. The height of the shapes in the charts is not significant.

For example, consider the groups of three students in the Agent course, that is, the green upward triangles in the second plot. In one group the most active student performed 57% of the commits, while in the other group this was around 80%. If all students in the groups would have committed an equal number of times, we would have seen 33% for both, as indicated with the gray bar in the chart.

5.5 Which Files Did the Students Commit to VCS?

The presence of files which do not belong in the version control system, such as executable programs, temporary compilation files and copied documentation, suggests that the students used the VCS as plain file storage. Figure 10 shows the fraction of the groups with a given percentage of redundant files in their final repository. We see a big difference between the figures for the respective courses.

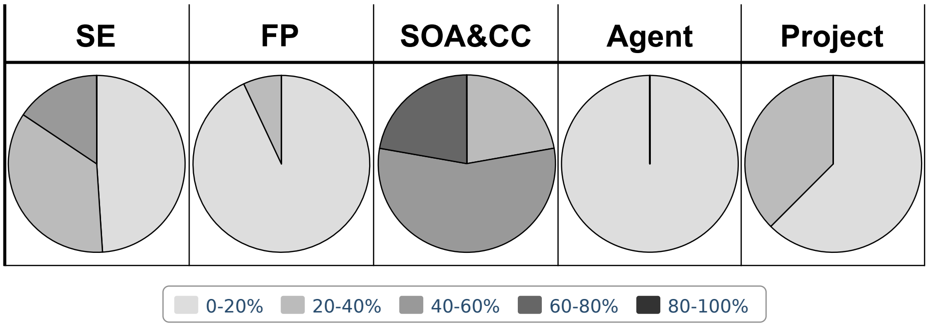


Fig. 10. Fractions of the groups with a given percentage of 'non-versionable' files in their repositories

In the chart for the SE course we see that most groups did not include many compiled files in their repositories, as was explicitly instructed in the course.

During the functional programming course, students can often test their code without actually compiling it, which could explain the low amount of garbage in the repositories. The garbage that is committed consists entirely of compiled binaries and other compiler generated files.

During the SOA&CC course many students use integrated development environments (IDE) which do the compilation automatically for the user. It seems like many students have included all files which the IDE produced to the version control system.

In the repositories of the Agent course there were close to no garbage files at all. Concrete, there was one group which had one backup file generated by an editor and a sub-repository which was checked into the normal repository.

In conclusion, the teachers of the respective courses came up with the following influential factors:

– When student are *explicitly instructed* to not include any redundant files in their repositories, they will most of the time pay attention. This explicit instruction was used in the SE and Agent course.
– When students work directly with the *Git command line tool* and not with a graphical user interface, they tend to include less garbage. This way of working was adapted in the FP and Agent courses.

- When students use an *integrated development environment* (IDE) then they will often include all files which are generated by the IDE. This effect was visible in the SOA&CC course where students often included a lot of compiled and other temporary files from the IDE.
- If the course has a *long term product development*, students will try to keep their repositories cleaner. The reason is that, in the other courses, when each week a new assignment is given, the old garbage which is in another folder will not bother the students in the weeks to come.

6 Conclusion

In this article, we focused on students' usage patterns in courses related to the computer science field while using the distributed version control system Git. We first looked at when students commit their work during the course and in more detail at their committing pattern during the implementation sessions. We concluded that most students commit changes regularly during the implementation sessions, but do not commit changes of work which they have been doing before the session itself. Some groups commit rarely during the session and make a big commit at the end of the session. These phenomena seem less pronounced in a course where students were introduced into the use of branching and the use of the version control at the beginning of the course. We did some effort in classifying commit messages and noticed that students do often write messages which are either trivial or even sheer nonsense. Nonsense messages appeared, however, much less frequent in courses where the students are creating a product over the whole span of the course. Further, we looked at how the usage of the system is divided inside groups and found out that the activity of the most active user in a group is significantly higher than what would be expected if each group member would use the system equally much. This behavior seems independent of the course type. As the last part of our analysis we considered the types of files which students put under version control. We concluded that there are several factors which reduce the amount of garbage in the repositories. First, students can be instructed explicitly to not put any files of the wrong type. Second, students can be advised to use the command line tools for version control and avoid integrated development environments. Last, students who create products over the whole run of the course will avoid garbage to keep their working space clean.

The hypothesis which put forward in our previous article seems to remain valid. Our findings suggest that using a VCS as a submission management tool may result in students adopting the tool as "just a required manner to return the assignments" — instead of a professional tool by which collaborative and distributed work is managed. Indications for this were that across all courses where the version control was not used to create a final product, the version control system was not too evenly used by the team members. It was also noticed that in these courses the quality of the commit messages was quite low, which we did not find in the project courses. While VCSs are pronounced as useful course management tools in the literature, we would like to note that professional use of

VCSs requires support, demonstration of their usefulness and student awareness of what is going on. Further, as noticed in the courses in which students did work on one product during the run of the course, it seems important that students get a feeling of ownership over the repository in order to use the system gracefully.

There are several aspects which could be analyzed in future research. One of the goals during every course is that students acquire new knowledge and skills. Hence, it would be interesting to observe how the metrics which we used in this paper are evolving during the course, e.g., how the quality of commit messages in the beginning of the course compares to the end. Another aspect which could be followed over time is the addition and deletion of garbage. Further, it might be possible to find prototypical students, i.e., these with a similar usage pattern and most likely attitude towards the use of this kind of systems.

We also came up with the idea of introducing a new type of reflective exercise in the class in the future. Student would, near the end of the course, get to analyze their own usage pattern in the repository. This task could increase the students' awareness and teach them better habits for their future work.

Promisingly, in our second-year Software Engineering and Master level Agent courses, training sessions were provided and the courses show a rather constant commit curve with peeks much less pronounced compared with the other courses. Further research could also point out how effective the students can use the system and how the organization of the group work influences the use of the system. It might be that some students know very well how to use the system, but do not see any reason to use their skills up to a full extent in the given settings.

Acknowledgments. We would like to thank the department of Mathematical Information Technology of the University of Jyväskylä for both financial and material support, without which this research would not have been possible. We would also like to thank the reviewers of the ICTERI 2013 conference for their useful corrections and suggestions, which greatly improved the quality of this paper.

References

1. Brown, R.: Group Processes: Dynamics within and between Groups. Oxford, UK, Basil Blackwell (1988)
2. Laadan, O., Nieh, J., Viennot, N.: Teaching operating systems using virtual appliances and distributed version control. In: Proceedings of the 41st ACM Technical Symposium on Computer Science Education, SIGCSE 2010, pp. 480–484. ACM, New York (2010)
3. Meneely, A., Williams, L.: On preparing students for distributed software development with a synchronous, collaborative development platform. In: Proceedings of the 40th ACM technical symposium on Computer science education, SIGCSE 2009, pp. 529–533. ACM, New York (2009)
4. Mierle, K.B., Roweis, S.T., Wilson, G.V.: CVS data extraction and analysis: A case study. Technical report utml tr 2004–002. Technical report (2004)

5. Reid, K.L., Wilson, G.V.: Learning by doing: Introducing version control as a way to manage student assignments. In: Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education, SIGCSE 2005, pp. 272–276. ACM, New York (2005)
6. Clifton, C., Kaczmarczyk, L.C., Mrozek, M.: Subverting the fundamentals sequence: Using version control to enhance course management. SIGCSE Bull. 39(1), 86–90 (2007)
7. Hartness, K.T.N.: Eclipse and CVS for group projects. J. Comput. Sci. Coll. 21(4), 217–222 (2006)
8. Liu, Y., Stroulia, E., Wong, K., German, D.: Using CVS historical information to understand how students develop software. In: MRS 2004: International Workshop on Mining Software Repositories (2004)
9. Lee, B.G., Chang, K.H., Narayanan, N.H.: An integrated approach to version control management in computer supported collaborative writing. In: Proceedings of the 36th Annual Southeast Regional Conference. ACM-SE 36, pp. 34–43. ACM, New York (1998)
10. Glassy, L.: Using version control to observe student software development processes. J. Comput. Sci. Coll. 21(3), 99–106 (2006)
11. Xu, Z.: Using Git to manage capstone software projects. 159–164 (2012)
12. Milentijevic, I., Ciric, V., Vojinovic, O.: Version control in project-based learning. Computers & Education 50(4), 1331–1338 (2008)
13. Rocco, D., Lloyd, W.: Distributed version control in the classroom. In: Proceedings of the 42nd ACM Technical Symposium on Computer Science Education, SIGCSE 2011, pp. 637–642. ACM, New York (2011)
14. Tirronen, V., Isomöttönen, V.: Making teaching of programming learning-oriented and learner-directed. In: Proceedings of the 11th Koli Calling International Conference on Computing Education Research, Koli Calling 2011, pp. 60–65. ACM, New York (2011)
15. Tirronen, V., Isomöttönen, V.: On the design of effective learning materials for supporting self-directed learning of programming. In: Proceedings of the 12th Koli Calling International Conference on Computing Education Research, Koli Calling 2012, pp. 74–82. ACM, New York (2012)
16. Isomöttönen, V., Tirronen, V.: Teaching programming by emphasizing self-direction: How did students react to active role required of them? ACM Trans. Comput. Educ. 13(2), 6:1–6:21 (2013)
17. Isomöttönen, V., Tirronen, V., Cochez, M.: Issues with a course that emphasizes self-direction. In: Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE 2013, pp. 111–116. ACM, New York (2013)
18. Parzen, E.: On estimation of a probability density function and mode. The Annals of Mathematical Statistics 33(3), 1065–1076 (1962)

Bridging the Generation Gap in ICT Education

Nataliya Kushnir, Anna Manzhula, and Nataliya Valko

Kherson State University, 27, 40 rokiv Zhovtnya St., Kherson, 73000 Ukraine
{kushnir,valko}@ksu.ks.ua, ilovetrees@mail.ru

Abstract. The main objective of this chapter is to describe the educational needs of modern students as representatives of the Net generation. We highlight the discrepancy between their characteristics and the traditional ways of teaching ICT disciplines. In this chapter we report our teaching experience and poll results at Kherson State University, Ukraine for three year period. We offer new teaching approaches to cope with the generation gap and ways to improve the quality of ICT teaching.

Keywords: Generation gap, Generation Net, teaching approaches, ICT discipline.

1 Introduction

The UNESCO report on Information Technology in Education [1] informs that Ukraine is on the way of "the rapid advancement of ICT in education that leads to constant updating of educational content and quality of ICT training". However, there are still many problems remain. Mainly, they are connected with the fact that teachers and their educational institutions are having hard time to make a transition to information society: “increased demands for flexibility, mobility and adaptability to the education management system, educational institutions and teachers in the context of rapid changes make it difficult to maintain and improve the quality of educational services."

2 Related Work

The general trends of students’ learning style and their behavior that were identified by our previous teaching experience and investigations on digital competence formation among future teachers have resulted in further research [17]. We attempted to find out the unknown factor that has significant impact on pedagogical process. We tried to understand the nature of this phenomenon by considering modern students as the representatives of a new generation.

To date, about 40 books and scores of articles and chapters have been written on this generation that report the results of international surveys and other research and describe new gens’ characteristics. Their impact itself on education at all levels has been interesting to researchers and educators too. There are about 10 terms to describe the current generation of students: Millennials (Howe and Strauss), Generation Y or Gen Y (Nader), Echo Boomers (Tapscott), Net Generation (Tapscott), Digital Aboriginals (Tarlow and Tarlow), Digital Natives (Prensky), Nexters (Raines, and Filipczak), Dot.Com Generation (Stein & Craig) [2].

The representatives of this generation are middle and high school students, college students, and posdocs. Roughly all young people in education born between 1982 and 2003. They have grown up in the world of high-tech, digital and mobile technologies, and used to be online 24/7.

The burgeoning technology itself has had a profound effect on this generation, unlike any previous one. In the classroom, they can chat on Skype or write SMS to their friends, take notes on Ipad, surf the Internet and read a book on ReadBook. This behavior cannot be fully appreciated by their teachers: it's considered that electronic instruments and digital devices distract students from the "real" study [2]. Most of today's teachers are representatives of the earlier generations. They use teaching methods and learning models that best fit them but not newcomers.

It was found that a representative of new generation inherent a range of characteristics that are defined a predisposition for becoming a successful educator [15]. They have potential, strong motivation and desire to make their society a better place. Some researches study and develop strategies for retention them in definite professional sphere including Education.

Moreover, the representatives of Net generation have solid moral values connected with a family and society as a whole. They are highly motivated to form open and tolerant society. It's important for new gens' educators to consolidate and maintain youths' system of values [15].

The results of the 3rd year students' polling showed the low level of professional awareness (Preschool and Elementary School faculty). Only 50% students have an intention to become teachers [16]. The absent of professional focus among students sets an important task for educators to make seeing themselves in teaching profession.

Among the characteristics of generation should be noted that new Gen workers are usually educationally focused and attribute their success to their educational capabilities, want to have a career promotion and a flexible work schedule, disregard the dress code, demand ICT equipped workplace.

Ronald A. Berk synthesized research evidence based on ten national and international surveys: EDUCAUSE [4], College Students‘ Perceptions of Libraries and Information Resources Survey, Greenberg Millennials Study [5], Education Research Institute (UCLA) [3] American Freshman Survey [11], National Center for Education Statistics [9], Net Generation Survey [8], The Net Generation: A Strategic Investigation [13], Nielsen Net View Audience Measurement Survey [2, 10], Pew Internet and American Life Project [6, 7] and Technological preparedness among entering freshman [12]. The research results from the surveys and aforementioned books have yielded twenty learner characteristics typical for most Net Geners: technology savvy, relaying on search engines for information, interested in multimedia, creation Internet content, operating at twitch speed, learning by inductive discovery, learning by trial and error, multitask on everything, short attention span, visual communication, craving social face-to-face interaction, emotionally open, embracing diversity and multiculturalism, preferring teamwork and collaboration, striving for lifestyle fit, feeling pressure to succeed, constantly seeking feedback, thriving on instant gratification, responding quickly and expecting rapid responses in return, preferring to type.

We have identified some teaching approaches that are contradictory to the contemporary students' needs. This gap is especially obvious in teaching computer

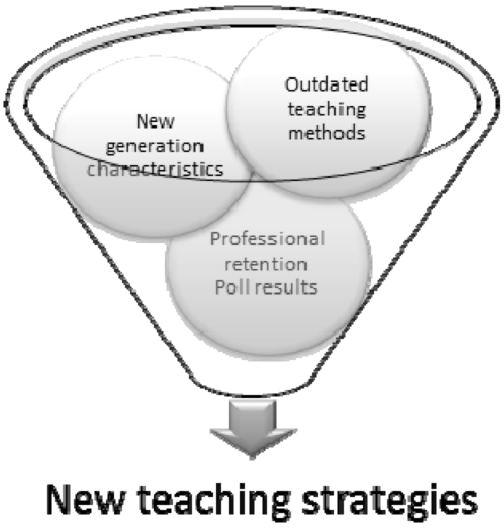

New generation characteristics
Outdated teaching methods
Professional retention Poll results
New teaching strategies

Fig. 1. Up-grading discipline background

disciplines. Complete comprehensive step-by-step instructions and exclusively individual learning are no longer efficient. This context led the authors of this chapter to a revision of present teaching strategies.

3 Setting Up the Pedagogical Experiment

The only discipline that is offered at Kherson State University and focuses on developing of ICT skills for future teachers of all specialties is “New Information Technologies and Technical Facilities of Education”. This year the discipline was renamed in "Information Technology" in most curricula.

The pedagogical experience of teaching the discipline "New Information Technologies and Technical Facilities of Education” in 2011 (109 students) and in 2012 (112 students) at Faculty of Pre-School and Elementary Education (FPEE) and results of polling students from different departments allowed us to identify problems that are mainly related to the mentioned disparity [17].

The poll of 60 students of non-computer (Faculty of Foreign Philology and Translation) and computer specialties (Faculty of Mathematics, Physics and Informatics) showed high level of access to ICT of Net Gens at Kherson State University. Most of obtained results are correlated not only among groups of computer and non-computer specialties, but also to the trend of global statistics.

100% of the respondents have their own laptop, Ipad or desktop, a mobile phone or a similar device. 84% of students have started using the computer for learning for 7 years or more. 29% of students spend 5-7 hours a day and 7% - more than 8 hours on the Internet on a weekday and only 1-2 hours with benefit. 84% of students spend more time online at weekends. Usually students spend about 2 hours a day speaking by telephone and at least 1-2 hours communicating through the Net.

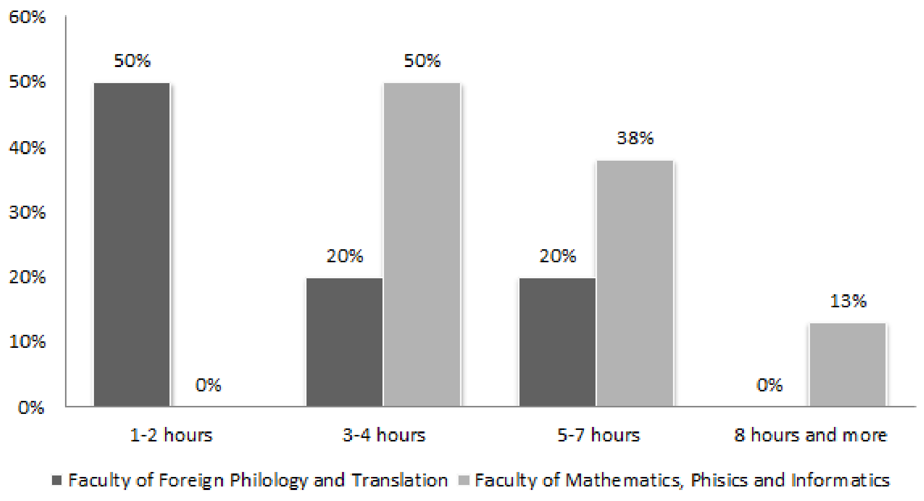


Fig. 2. Students usually spend at the computer on a weekday

Students use computer mainly for learning – 25.5%, entertainment (watching movies, playing games) – 21.5%, communication (social nets, chats, forums, instant messengers, etc.) – 18.5%, personal development (taking courses, reading books out of an academic program, etc.) – 14%, creativity (writing music, making video, etc.) – 13%, other purpose – 8.5 %.

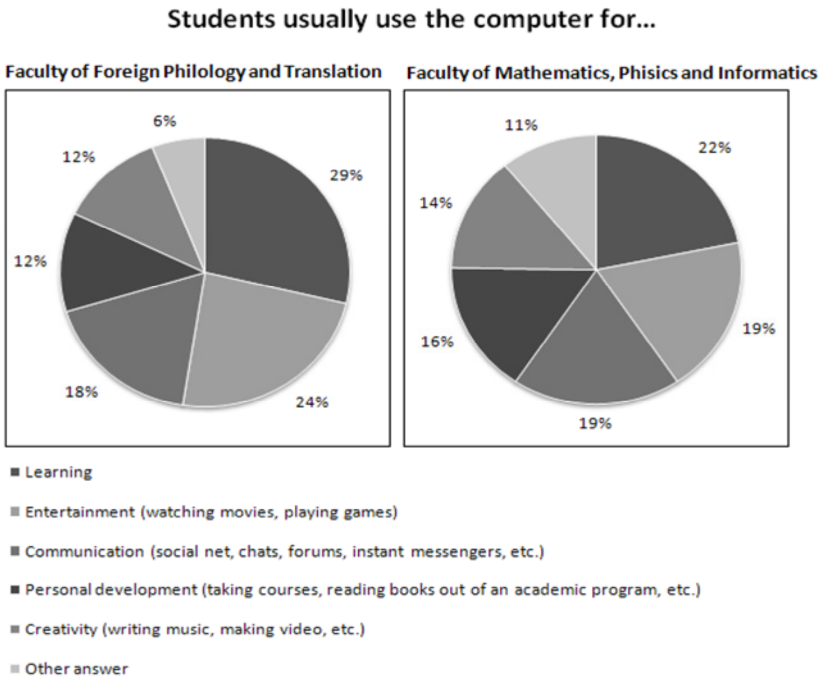


Fig. 3. Main objectives for using computer and the Internet by students

26% of students mentioned search engines as the most frequently used site on the Internet, 19% - social networks, wiki sites, 7.5% - forums, 8.5% - sites with ready-made home tasks, 7.5%, - educational portals, 7.5%, - e-mail and 1.5% - e-Library.

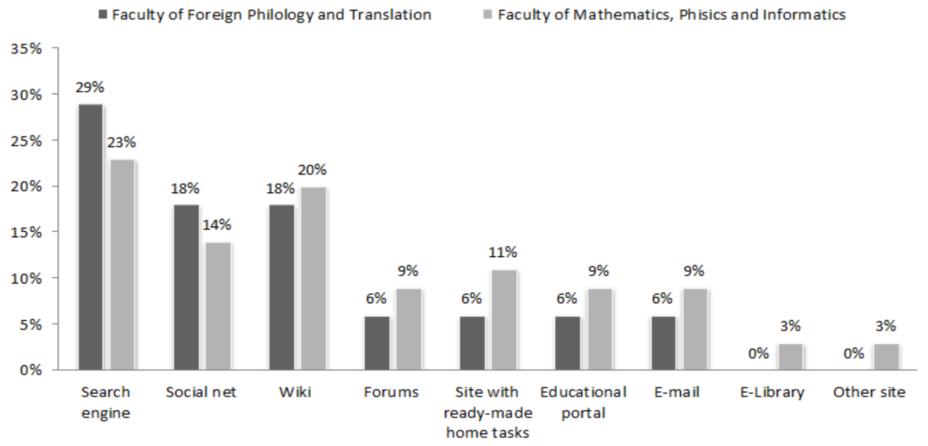


Fig. 4. The most frequently visited sites by students

Among the typical learning tasks at university that are associated with the use of ICT, students noted the following: searching information - 26%, writing reports - 17.5%, making multimedia presentations - 15.5%, translating text - 14.5%, filling tables and forms - 12%, watching movies - 8%, other - 6.5%.

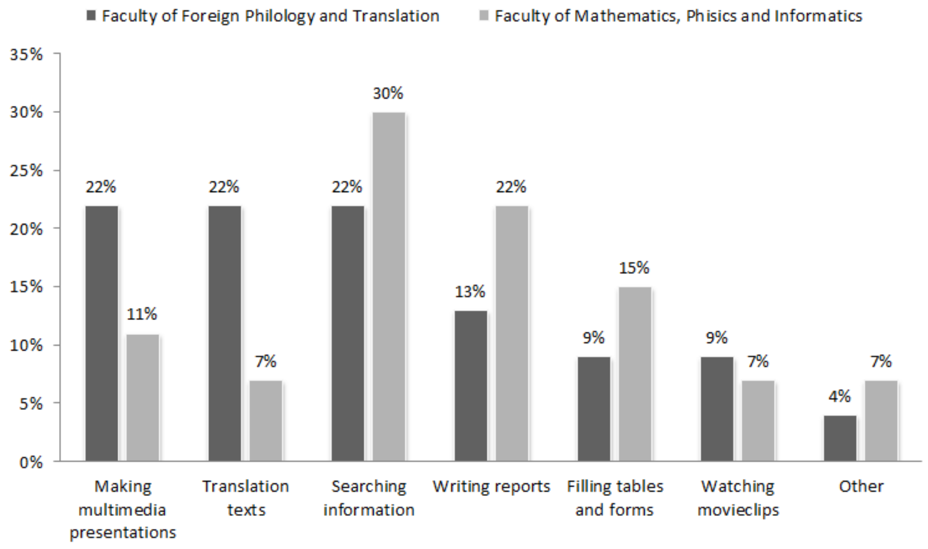


Fig. 5. Typical learning tasks in ICT use

Thus, the poll results indicate that the students at Kherson State University refers to a group of people with a high level access to ICT and inherent essential characteristics of the new generation.

Therefore, our students desire experience and experimentation in the classroom and extracurricular activities. They are enthusiastic, visually oriented, ready to cooperate, easily switch environment, emotionally open and spontaneous. However, these features cannot be seen by researchers, psychologists and educators as strong or weak sides of this generation. New generation are future adults, it is necessary to listen to the educational needs of our students, perceive them as independent and self-sufficient part of society, to make contact with them and start sharing positive experiences between generations.

To meet modern students' educational needs such as visual communication, constantly seeking feedback, thriving on instant gratification, preferring to type we have published the course on e-learning platform Moodle «KSU online». Its open access for authorized and unauthorized users allowed realizing element of distance learning. In addition to lessons and communication through KSU online, the teacher consulted students by e-mail or mobile phone, so at least one quarter of the students asked for advice by e-mail, 1/8 of the students contacted by a mobile phone.

Students were offered to use instructions published online, but they tried not to use them at all. It was preferable for them to consult with the teacher or with colleagues. The advantage of short instructions (instead of step-by-step instructions) is a short time students spend to understand a plot of a task ("what should I get as a result?"), increasing level of concentration on the task and, accordingly, the level of satisfaction after its execution, intensification of communication in a group and with the teacher, searching for additional information and taking independent decisions. It's interesting that students found sometimes unexpected ways to solve the same problems, using convenient for them online resources.

Instructions are saturated with images that provided quick perception of information by students, and their desire to show high quality of design, accuracy in their works.

Tasks provided freedom in the form of organizing work – individual, in pairs or groups. Only 1 out of 8 tasks "Creating a website" excluded other forms of work except individual and one task "Creating a Memory Card" is supposed exclusively group work, other tasks provided the freedom in choosing coworkers. As a result, 80 of 84 works were performed in pairs or small groups. The composition of groups has changed during the course but slightly.

Complex and problem tasks caused increasing communication in groups and between groups, a desire to work in a team and to distribute work.

The result of each practical lesson was an original creative work created by students (website, email, memory card, etc.). It's common when students take success of their creative works as their personal success; they consider their works as expression of their individuality. The majority of students acts competitive and tries to do their best. All the students were proud of their results, teachers and colleagues' appreciation strongly influenced on students' attitude.

Students' works were published on the course webpage. The views statistics of web page with the students' works two times exceeds quantity of views of the task page, which demonstrates the interest in the evaluation and comparison of creative works from students' point.

During the course we observed the absence of a psychological barrier to ICT that we faced with students earlier, competitive atmosphere, students acted with dedication, resourcefulness, creativity, aesthetics, aggressiveness, ambition and confidence.

We have analyzed and matched traditional ICT teaching approaches, characteristics of Net generation, empiric data (polls, page statistics) and presented contradictions in the table below.

Table 1. The contradictions of the present ICT teaching approaches to the generation Net characteristics

ICT teaching approach	Generation Net characteristic	Poll results
ICT teaching "from scratch" disregards the actual level of the student's ICT skills. As a result, school ICT discipline assignments are duplicated; lab manuals are detailed and tend to be comprehensive.	Tech savvy	84% of respondents have started to use the computer for learning 7 years ago or earlier.
Teaching materials are not interactive and updating.	Relying on search engines for information	About 26% of the students mentioned search engines as the most frequently used site on the Internet.
Weak level of visualization of teaching materials, lack of interactivity including hypertext	Interest in multimedia, "visual" communication	Movies and computer games have the second position among the purposes of using computer ranked by students.
Step-by-step manuals that presuppose learning by copying the sample, the absence of the original product as a result of the students' work	Creation of Internet content	Social nets, wiki-sites and forums have the third position among the most frequently visited sites on the Internet. All of them are a platform for a creation of people's own content, posting their opinion, sharing things made by them. 13% of respondents mentioned the creative activity as a major purpose for using the computer.

Table 1. (*Continued.*)

Students are constrained by one plotline in learning process, the absence of immersion, problem-solving and decision-making tasks and enough freedom for actions in realization students' learning trajectory	Multitasks on everything	The sum of hours for different everyday life activities informed by student is about 28 hours a-day.
Weakly realized person-centered approach	Emotionally open	Social nets were ranked as the second position by students.
Individual fulfillment and performance of learning results	Teamwork and cooperation	Using computer for communication among students is placed on the third position after learning and entertainment purposes by students.

As for our course content and type of its presentation, results of interviewing students of Faculty of Pre-School and Elementary Education in 2012-2013 academic year have confirmed last year statistics: quality of teaching materials on KSU Online was highly appreciated by students, average score was 9.29 in 2011 and 9.16 in 2012 out of 10.

The results of the entrance poll of different faculties in 2011-2013 academic years showed the following tends:

- In 2011 89% of the respondents owned a computer or a laptop, this academic year 100% of students are owners of computer or similar device regardless of the faculty.
- 70% (2011), 68% (2012) and 69% (2013) students have an access to the Internet outside the university – thereby, this rate stays the same.
- However, amount of students who recognized themselves addicted to the Internet has grown from 24% in 2011 to 32% in 2013.
- Interesting results were found about visiting University website by students from different faculties. 17% of students of the third year said they had never visited the university site at faculties where teachers hardly use ICT in teaching, while the rate has tended to 0 (since 2011 to date) at faculties where most of the teachers regularly use ICT.

The number of students who have a positive attitude towards the use of ICT in education (inter alia, at lectures) has increased (see Figure 2).

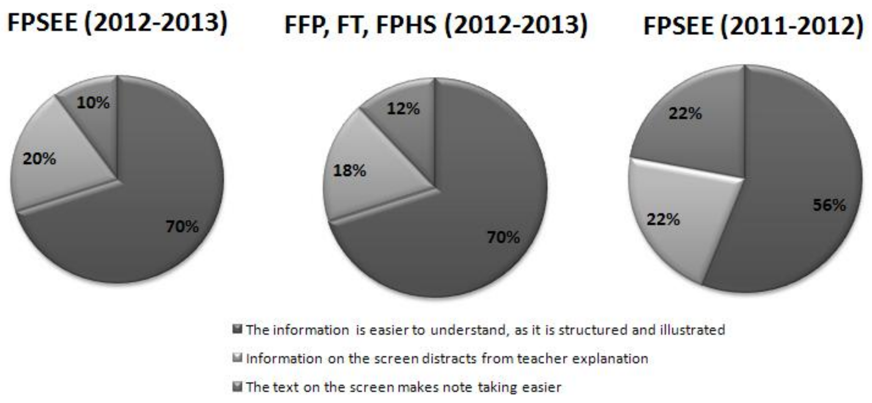


Fig. 6. Students' Attitude towards the Use ICT during Lectures

The purposes of using the computer by students have almost the same rating, regardless of year and faculty (see Figure 3).

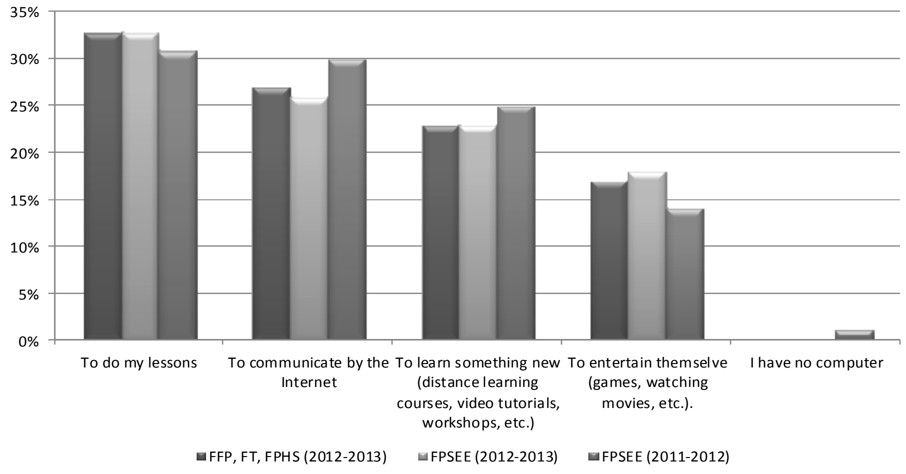


Fig. 7. The Purpose for Using a Computer and Internet by Students

Social net is becoming more popular among services for communication (see Figure), IME and email are also frequently used.

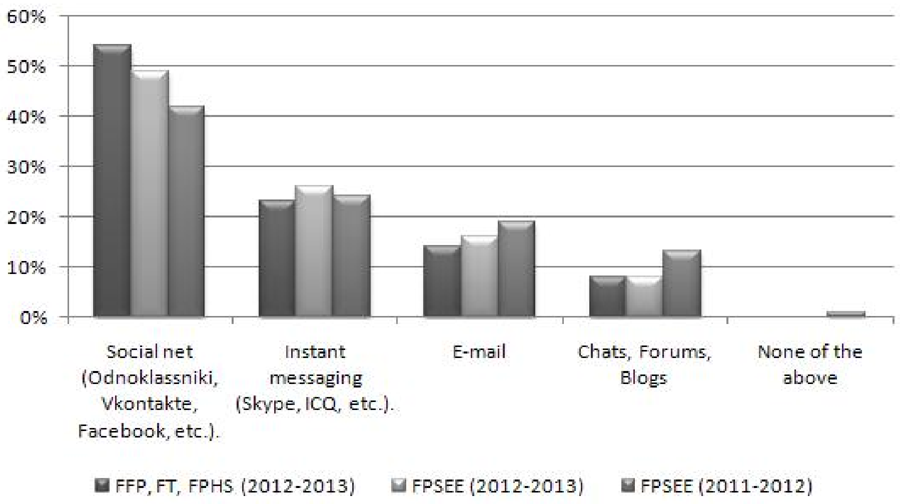
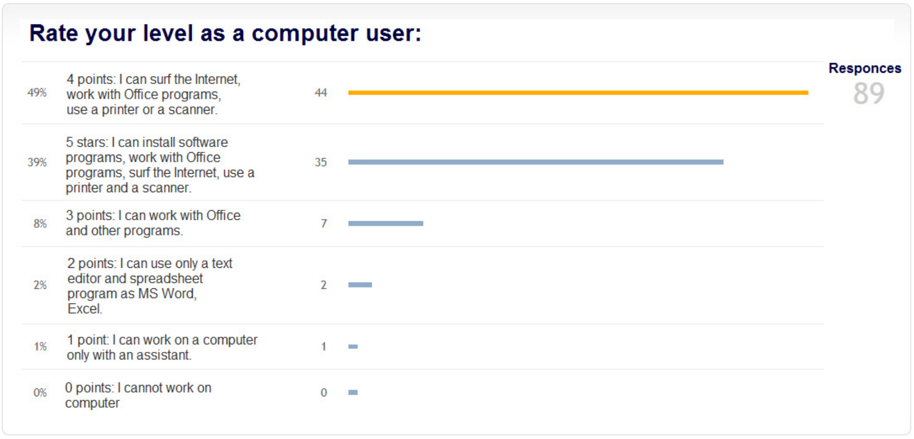


Fig. 8. Using Communication Services by Students

Modern students overestimate their skills for online searching according to Berk. The results of our research have proved this fact. The results of entrance poll and test in Informatics in our course are contradictory. The majority of students estimated their level of ICT proficiency as excellent and good. The rate is depicted below.



Rate your level as a computer user:

Responces 89

49%	4 points: I can surf the Internet, work with Office programs, use a printer or a scanner.	44
39%	5 stars: I can install software programs, work with Office programs, surf the Internet, use a printer and a scanner.	35
8%	3 points: I can work with Office and other programs.	7
2%	2 points: I can use only a text editor and spreadsheet program as MS Word, Excel.	2
1%	1 point: I can work on a computer only with an assistant.	1
0%	0 points: I cannot work on computer	0

Fig. 9. Results of Students' Self-appraisal as a Computer User (FFP, FT FPHS, 2013)

The entrance test contains 45 comprehension questions and is designed by analogy with ECDL test. The aim of testing is to check residual knowledge of school computer discipline. Its results showed that only 4% of students had a score higher than 4, 31% had a score more than 3 but less that 4 points, 43% had from 2 to 3 points and 22% scored less than 2 points.

During the practical lessons following problems that are related to organizational issues and students attitudes have appeared:

- Students have no skills to manage their time and efficiently work without strict teacher's control, neglect deadlines. Only some students uploaded their works on time.
- Not all students use email regularly. Many of them chronically forget their email user names and passwords. Students are confused if use other browsers or other version applications. As a result, login process for email or distance learning system regularly takes up to 7 minutes.
- There is no clear understanding of the terms among students.
- It is necessary to consider different pace of students' work while planning lessons. Some of students do not regularly use computers, so typing, mouse control, searching for a command cause a delay.
- Plagiarism. Some students prefer not "to waste" their time to do lessons. They easily copy their colleagues' results or download similar ones from the net. It is important at the first lesson to highlight the value of students their own creative work and punish the students' attempts to copy someone else's work. Under the strict control the number of plagiarism cases decreases but does not disappear. It is growing according to the task's complicity. The reason for such situation is the priority for many students to get a good grade and not knowledge.
- The students do not like reading, especially READING OF long instructions. Usually they prefer to ask a teacher or colleagues than to read a manual step-by-step. It is also associated with different systems prevailing perception among students. Feature of the course is that most of the teaching materials are stored in digital form. Creating e-learning course does not require additional financial costs, as opposed to hard copy. It is faster and easier to edit and to update. However, reading from the screen does not give effective results. Perceived only about 40% of the information.
- Lack of collaborative activities, such as work in pairs, groups and teams, social assessment, problem-solving and decision-making tasks. A student presents results of an individual work only to one person– his teacher, and feels subordinated under such conditions. In addition, it is the reason of the lack of the critical view on his own work. This results in the students' frustration and dissatisfaction to any criticism. Teacher comments are usually perceived in a negative or indifferent way.
- Educators should pay special attention to students' feedback about effectiveness of teaching activities and the level of their satisfaction. Such monitoring and evaluation allow teacher to obtain information about teaching effectiveness and quality, get information to improve or change his work. It should reflect the dynamics of the level of one student's knowledge.

The presented requirements to discipline designed in previous work on digital literacy formation [17] have been adapted to Berk's pedagogical strategies. As a result, we have formulated new approaches considering educational needs of the Net students' generation (see Table 2).

Table 2. The approaches of teaching ICT discipline to meet educational needs of generation Net students

The teaching element	Requirements
Discipline content	– A discipline content should reflect current research and encourage students to use new approaches and technologies, highlight current trends in ICT development. – Tasks presuppose creative activity, form skills of self-learning and motivate to further development. Examples are inspiriting. – Result of task execution are useful, valuable and applicable product in professional practice. – All elements of the discipline are focused on future professional activities.
Students motivation	– All elements of the course (assignments, surveys, etc.) help students to see themselves in their future profession, particularly in teaching profession. – It is important to emphasize interest to the students' opinion, to make possible them to contribute to collective projects, for example, to create a bank of teacher's materials; – Student wants to get a feedback from his colleagues about his work. Any result of creative task should go through following stages: creation, publication and social assessment with definite criteria. – Student wants to evaluate teacher's work too, so a teacher should organize the ways for students to impact on the discipline development, to express their wishes.
Organizational issues	– Tasks presuppose collaboration, facilitate communication and interaction to develop personal aspect of student and help to realize his individuality. – Clear structure, planning, to do lists and deadline system. – Active use of formative assessment techniques. – Ice-breaking, team-building and communicative exercises at the beginning and at the end of the lesson to form communicative skills, values, relationship and positive atmosphere. – Use no more than two new computer environments at lesson. – Teacher should consider in the selection of services to work at lesson: – registration absence or its simplicity, – functionality, – easiness, – necessity to install additional software, – potentiality to use in profession.

The main aim of updating the discipline was to help future teachers to create and organize their own learning space in the Internet. Therefore online interactive services that can be used for communication and teaching pupils were included (creating word clouds, mind map, open online documents, site, etc.). We also considered Berk's strategies while designing the discipline. The updated version of "Information Technology" discipline was taught at the Faculty of Foreign Philology (FFP) - 104 students, Faculty of Translation (FT) – 61 students, Faculty of Psychology, History and Sociology (FPHS) – 28 students.

Table 3. The implementation of Berk's teaching strategies in the course "Information Technology"

Characteristic of Net Gen student	The implementation of educational strategy
Tech savvy	The virtual discipline environment was created with the system of distance learning ksuonline.ksu.ks.ua located on a MOODLE platform. This system allows developing a course with such elements as glossaries, wikis, multimedia clips, presentations, tests, blogs, forums, etc.
Teamwork and cooperation	Some tasks are complex and presuppose collaboration, while their execution reinforces the cognitive interpersonal communication and interaction of all participants. An important stage is to prepare a group of students to work together. To ensure about students' readiness for cooperation a teacher should arrange Ice braking, team-building and communicative exercise (up to 5 minutes) at the beginning and at the end of lesson. Teacher, who works in the computer classroom, usually recognizes his students primarily "from the back." Therefore, such exercises facilitate personal contact with the group, which is especially important while teaching 1 or 2 credit disciplines.
Interested in multimedia, "visual" communication	The discipline content was developed and designed considering students' interest in "visual communication" and multimedia. Videoclips and presentations were added to each theme. The tasks and manuals contain minimum of text information and maximum of graphics and illustrations. Teachers included such elements as blogs and forums.
Rely on search engines. Multitasks on everything	A teacher encourages students to use search engines during the course and work with Google services as Google.Drive, Google.sites, etc. This satisfies the interest in creating online content and allows simultaneously to work with several documents. It's also an efficient tool of organization of group activities.

Table 3. (*Continued.*)

Retention in the profession	Organization of students' activity is a process of constant modeling of professional activity of specialists under learning conditions. The students of teaching specialties create educational games, tests, documents, tables that are useful in their professional practice. The results of the work are evaluated not only technical element, but also educational one. Also such task as creating a presentation "My choice" (students aimed to describe situation of their professional choice, analyze their present achievements and capacities, goals and dreams, ways to attain them) and collective writing of mind map "Modern teacher" were added.
Emotionally open	Tasks are person-centered and presuppose creation unique results that will describe student's own lifestyle, attitude, etc. This approach helps to decrease the quantity of plagiarisms and contributes to the development of their creative abilities. Each student's work passes through formative assessment: self-appraisal, social assessment, teacher's assessment.

Strategies described above are in agreement with learning strategies that have been formulated in the UNESCO recommendations and the requirements presented to ICT competence of teachers - UNESCO's ICT Competency Framework for Teachers [19].The selection of tasks should be aimed to strengthen professional orientation of future teachers (creation of interactive games, information booklets, working with serial documents, recording and analysis of quantitative data in a spreadsheet), to reduce the proportion of reproductive work (complex creative tasks), and to focus the training process at development and production of knowledge.

Tasks such as making word clouds, mind maps, learning games are relevant to the professional and educational discipline orientation. In addition, they simulate creative activities and can be easily adapted to a group work.

Poll Results of 2012-2013 academic years at Faculty of Pre-School and Elementary Education are comparable with poll results published one year earlier. The quality of teaching materials on KSUOnline highly appreciated by students, average score was 9.29 in 2011 and 9.16 in 2012-2013 out of 10. The novelty of lectures was assessed as 7.59 in 2011 and 6.84 in 2012-2013, the novelty of practical tasks had 7.64 and 7.39 respectively.

The students' preferences about the most interesting and useful for future professional tasks have changed. In both of these categories the task "Creating didactic game" leads. Making a poster "It's interesting to know," creation of "Site Class" and creation of online poll follow in the rating. These tasks we preserved in the updated version of the discipline "Information Technology". As a result, new version of "Information Technology" discipline has the following structure:

The student preferences on most interesting and useful for future professional tasks have changed. The task "Creating didactic game" leads in both these categories. Making a poster "It's interesting to know," creation of "Site Class" and a creation of online poll follow in the rating. These tasks we preserved in the updated version of the discipline "Information Technology". As a result, new version of "Information Technology" discipline has the following structure:

Lesson 1. Registration on KSUOnline, taking an entrance poll and test. Presentation "My choice" with the following structure:

1. My strengths and weaknesses (academic, creative, personal, individual aspects).
2. My goals, objectives and deadlines (to 2020 year).
3. My completed tasks.
4. A letter from You yourself in 2020.

Lesson 2. Creating a class site on Google Sites (Students add, delete pages, edit them and insert pictures and text, change site's design). Taking a poll "NET Gener profile Scale".

Lesson 3. Creating a poll on Google Drive and inserting it to the site.

Lesson 4. Creating a cloud of words (Tagxedo service), publishing pictures with a cloud of words and writing an assignment for the pupils on the "Homework" site page. Changing assignments with the word clouds and its execution.

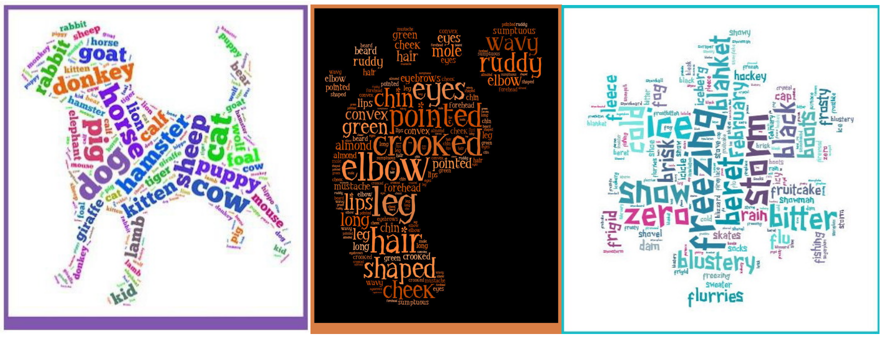


Fig. 10. Ipyna *Rebenok*. Animals (5 grade). Task: Sort the words into categories: domestic animals and wild animals. *Alena Kudina*. Winter holidays (7 grade). Task: Write a story about winter holidays using words from the wordcloud. *Yana Yanyk*. Appearance (7 grade). Task: Make several word combination on the topic "Appearance".

Lesson 5. Creation educational games with triggers or hyperlinks in MS PowerPoint.

Lesson 6. Social expertise of the games (organized with Google Drive – a spreadsheet). Watching the movie "The image of the modern student" by Michael Wesch (Kansas State University, 2007). Creating a collective essay "How I like to learn" in online document on Google drive.

Lesson 7. Co-creation of mind map "Modern teacher." Taking final poll and test.

Making a poster “It’s interesting to know” and contributing to wiki page “Communication in the network” are for independent work.

Returning to the problem of assessment it should be noted that in a presentation "My Choice", 68 students described their school successes and achievements considering them as the next step in their professional development, but some students evaluate these achievements as formal, not real:

> "*To the moment of leaving school I had about 60 letters of honor in my* "*collection*"*. But I understand they only have formed my ability to learn and now mean nothing*".

Ice Breaking. We believe that training and game technologies are vital elements in teaching students, especially students of pedagogical specialties. They help contact the auditory rapidly, create friendly atmosphere and get in tune to work and, as a result, to improve relations among students. Besides, it’s favorably, if a prospective teacher has an opportunity not only theoretically explore innovative techniques but test them oneself as a student. Current students who were taught traditionally would teach differently in several years.

Obviously, a success of training approach strongly depends on a teacher personality and his skills in training and teambuilding. Therefore, the authors of this chapter have selected games and exercises that do not require special teacher’s skills and knowledge or any additional devices.

Despite the fact that the third-year students have attended a number of courses on psychological and pedagogical technologies, the participation in training and communication games was a discomfort for them. During the course, we’ve found that the most third-year students have lack of communication. For example, at the first lesson after the game "Introduction", we asked all students to tell about their three small victories. Such "victories" were often met with utter surprise in the group. It was uncovered that the students' ability to learn is directly connected to their sense of self-confidence and psychological atmosphere.

As a closing activity, we frequently use games to develop students’ awareness about their course progress. In our opinion, the most successful were the games “Medal for me” after creating the class site – the most difficult task, and “Nobel Prize” for the last lesson of the course.

The plot of the first one is the following. The teacher gives every student a round piece of paper to write his name and his/her brightest achievement during the lesson (e.g., students have dedicated the medals “for patience”, “for the first website in my life” and etc.). Students, one by one, come up to the teacher who is ready to award a medal with cheers and all the honors. It’s important all students are standing in a round at this stage of the game. If students are sitting, they feel like spectaculars but not like participants.

The game “Nobel prize” is similar, but the level of psychological comfort is lower. The students are asked to choose his highest achievement during the course. Then every student in turn stands up on the chair – “the stage” – and try to act as a famous scientist, starting his Nobel speech with the phrase “The core of my discover is …”, then tells about his/her best results, thanks the people who helped him/her, etc. Then the group should give the student a standing ovation. The most difficult thing for teacher in this game is to help some students not to be afraid to climb “the stage”.

Understanding students’ progress and success themselves is an important factor that significantly affects their motivation to learn and improves general attitude to the subject. Therefore, summing up the lessons, the teacher should emphasize new knowledge and skills gained by students.

Training elements in students’ groups of different teachers’ specialties and years of study had different reaction. Some students express stunning and rejection: "Why should I do that? It is not serious." despite the fact that they had finished Innovative Teaching Methods and Technologies discipline, Training exercise was an extraordinary situation. One teacher said describing her experience: "Third-year students look so serious that sometimes I feel scared to come to the classroom. It is like teachers and students have switched their roles...”. We find it particularly important to use the elements of training and communicative techniques working with students of teaching specialties. Present students are future teachers who are taught traditionally. In several years, they’ll adapt one of teaching styles they have seen. It is important to implement innovative techniques and give the students a chance to test them in practice.

The social aspect is a priority for a new generation life, they tend to have a relationship regardless of the field of interaction, and education is not an exception. If a teacher does not build relationship consciously, it usually takes the worst form. A teacher should know that a mark is no longer a sufficient motivation for the positive attitude of a student to the discipline, especially in the pass-fail system. It is clear that students often associate the discipline with teacher’s personality. The high quality executed tasks show not only an attempt to get a high mark but sympathy to a teacher. In computer disciplines social aspect is often nearly absent or poorly developed: most of the time students are working individually on the computers. Times of detached teacher have passed. Today’s successful teacher can easily contact the audience in a short time, acts as a partner, a friend and a leader.

Social Expertise. It’s time to change assessment system, to shift the emphasis from control to formative function to strengthen students’ motivation. In addition, it’s necessary to vary teacher’s assessment with self-assessment and social assessment due to teacher’s criteria. Students can determine the quality of the work due to criteria and assess technical realization, structure, relevance to age of pupils, subject, design, quality of illustrations, literacy, etc. This strategy forced students to think about their works from independent standpoint.

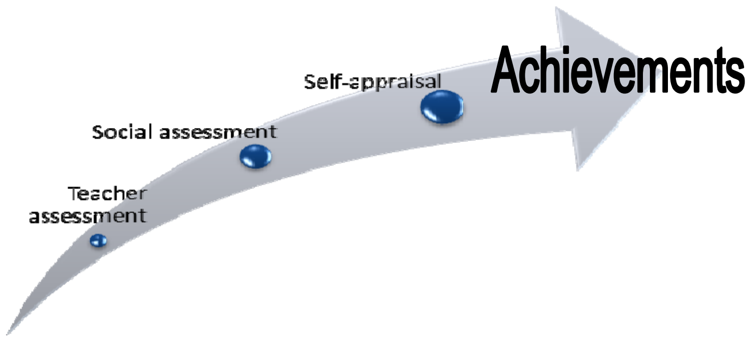


Fig. 11. Increasing Students' Achievements by the Method of Formative Assessment

The specific of a teacher profession is that teaching materials should be adopted for unique group of children and pedagogical situation. Therefore, our course includes many tasks, results of which are students' individual or group creative works. However, we have faced a problem with an evaluation of such pieces: a student's work may be technically perfect, but generally does not produce the impression of integrity and completeness. We consider a social expertise as the best method of evaluation in such situation. Its aim is to establish the extent for quality of work, as well as the preparation of proposals for achieving compliance. An assessment of works by students allows us to solve the problem of content of work, the quality of its performance, to identify weaknesses and make appropriate adjustments, to make recommendations for improvement. The method of social expertise helps to develop critical evaluation skills and change social value orientations in a group of students.

Criteria for evaluating students' works on each topic and shared document were published by the course developers to implement social expertise. Each student has an available document with criteria and fields to give other students' marks. In our point of view, each work should go through following stages: creation, publication (presentation) and an assessment. Students could try a role of expert, while the teacher's role was to coordinate and guide. A student with low self-esteem at first is afraid that his groupmates will assess not his work but his person, but it has never happened, all students are aimed to be objective and independent judges.The results of social expertise provided adequate evaluation of works. It also solved the problem of students' misunderstanding and distrust about results of teacher's evaluation and ignoring proposed criteria during creation student's own work. It also encouraged students to create neat and bright works that can be interesting for other students and almost entirely help to escape the problem of plagiarism.

Social expertise has stimulating, diagnostic and formational functions. A wish to get social appreciation motivates to create high-quality individual works. Such way of assessment forms critical perception of the information.

Teamwork in the Network. We used online service GoogleDrive to meet the new generation needs in communication in social networks and multimedia. GoogleDrive lets share documents for collective editing. In the course we used:

- Google spreadsheets for social expertise;
- Google Text Document “The way I like to learn…” to learn more about a comfortable learning environment for modern students;
- Memory-card “An image of a modern teacher” to reinforce retaining students in profession.

Use of shared documents in asynchronous collaborative writing was inspired by a set of collaborative learning theories that use Piaget’s model of equilibration to describe how the cognitive conflicts generated by the heterogeneity of the participants motivates students to contribute to a document and learn from their interactions [18].

Scientists give preference asynchronous types of interactions that allow evaluating work of those students who were absent at traditional lesson too. However, there is a problem that not all services have option for monitoring the contribution of each student. Using collaborative writing allows the teacher to monitor the progress of student learning better, creates a sense of responsibility among students and motivates them to cooperate.

Creation of opened for collective editing documents has aroused interest to collaboration. Students said they liked not only to express their opinions, but be able to see what others are doing and react to it.

Collective work should be preceded by a problem situation. For example, for the setting a discussion about an efficiency of modern education we used the video "The image of the modern student," which presents the results of a study conducted by Michael Welsh, the University of Kansas (USA). It indicates problems in the modern system of education from students’ point of view. Collective writing the shared document “The way I like to learn” is the next stage of work on this topic. The result of students’ discussion should be presented in "the tree of changes" with ideas how to improve the system of education. The next step is to create mind maps "The image of the modern teacher," in which students make a portrait of the characteristics of the modern teacher. The result of these activities is a formation of social skills and social consciousness.

The process of group learning is opposed to traditional front-to-individual and characterized by such basic features as

1. Participation. Group work reinforces an information field extension of individuals and a group as a whole. They learn to discuss problems together, make collective decisions and develop mental potential.

2. Socialization. Students learn to ask questions, listen to their colleagues, to monitor presentations of other students and interpret heard. Gradually understanding of the need of active participation in a work group and responsibility for one’s contribution in collaboration are coming. Students are given the opportunity to "try on" different social roles.

3. Communication. Students need to know how and when to ask the question of how to organize the discussion and how to manage it, how to motivate participants, how to speak, how to avoid conflicts, etc.

4. Reflection. Students should develop skills of inner reflection and self-analysis, understand how to assess the results of their own activities, individual and group participation, and the process itself.

5. Interactions for self-development. Students should realize that the success of their learning activities depends on the success of each individual student. They should to help each other, to support each other, to foster development, because in cooperative learning model it is a necessary "win-win" process. In this case, everyone is responsible for everyone, for all, for the entire educational process.

The analysis of the benefits of teamwork and experience in the use of the learning process, set up new questions:

– choice of instruments for students' interaction (eg, forums, knowledge map, wikis, online services);
– scenario (a sequence of actions in students' interaction and solving a problem);
– pedagogical aspects (a learning objective, a creation of a motivational moment, methods of estimation of the group and a contribution of each student, etc.).

It has defined the vector of our forward research: computer-supported collaborative learning (CSCL) to improve traditional and distance learning.

Another important improvement in the organization of the discipline was using of open online document on GoogleDrive among teachers. Each of three teachers has contributed to planning, selection of training exercise and other notes. After each lesson, teachers made a record in "a discipline online diary" to describe in free form results of the lesson, such as the most common students' mistakes, teaching and technical problems, positive situations, questions from the students, etc. Keeping a discipline diary had a great success. Teaching has become more effective and convenient, the level of awareness and collaboration among teachers has risen, so now teachers create and use it at other disciplines.

4 Results and Discussion

Developed approaches allowed us to improve a range of disciplines including "Information Technology", which is taught for students of all teacher specialties. We also applied them to following disciplines: "Introduction to Information Technology" (for future teachers of elementary school and Computer Science, the 1st year of study), "Fundamentals of Computer Science and Applied Linguistics" (translators, the 2nd year of study), and "Office Computer Technology" (programmers, the 1st year of studies).

Students expressed their positive attitude verbally in the classroom and several students sent e-mails with gratitude after finishing the discipline.

The results of the final poll confirmed their appreciation.

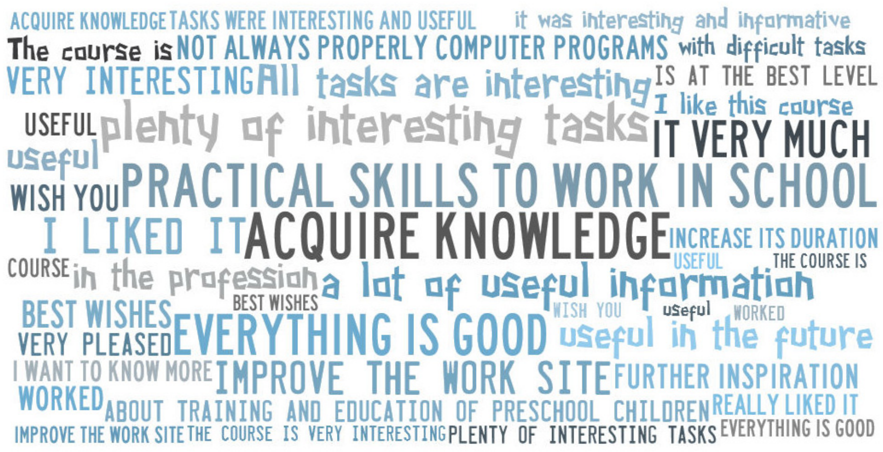


Fig. 12. Word cloud composed of students' comments about IT discipline

5 Conclusions and Outlook

Thus, we have obtained the following results:

1. To eliminate mismatching vision on learning format and bridge the gap between generations we discussed and analyzed characteristics of Net generation students that should be concerned while developing and teaching ICT disciplines. Following characteristics are directly connected with teaching ICT disciplines: technology savvy, rely on search engines for information, interested in multimedia, create Internet content, operate at —twitch speed, learning by inductive discovery, learn by trial and error, multitask on everything, short attention span, visual communication visually, crave social face-to face interaction, emotionally open, constantly seek feedback, prefer to type.
2. We highlighted the discrepancy between the characteristic of today's students and traditional ways of teaching ICT disciplines. The characteristics of modern students are different to traditionally explicit step-by-step instructions to labs, lack of interactivity of didactical materials, weak level of visualization, individual fulfilment and defense of the results, absent of personal centered approach and creative element in tasks.
3. We have found new approaches to resolve these discrepancies and implemented them in teaching practice. Such elements as discipline content, student motivation, and organizational issues were brought up and reconsidered.
4. We have made a revision of the courses to form professional awareness of future teachers, change learning environment and students' attitude to position of socially active, open-minded, creative specialist.

In the future, we plan to expand the list of pedagogical specialties to which the updated version of the course "Information Technology" will be taught and improve other disciplines according to the proposed approaches.

We have defined following problems that need further research:

- methods of reinforcing students' motivation while group work;
- assessment of every participant's contribution in a group project;
- organization of discussion and collaboration between future teachers and practicing teachers;
- sharing experience among other university teachers, organization of trainings, workshops, seminars, round tables, etc.

References

1. ICTs in higher education in CIS and Baltic States: state-of-the-art, challenges and prospects for development. Analytical survey. GUAP, St.Petersburg, 160 p. (2009)
2. Berk, R.A.: Teaching strategies for the net generation. Transformative Dialogues: Teaching & Learning Journal 3(2), 1–23 (2009)
3. Cashmore, P.: Stats confirm it: Teens don‘t tweet (Nielsen NetView Audience Measurement Survey (July 2009), `http://mashable.com/2009/08/05/teens-dont-tweet`
4. DeAngelo, L., Hurtado, S.H., Pryor, J.H., Kelly, K.R., Santos, J.L., Korn, W.S.: The American college teacher: National norms for the 2007–2008 HERI faculty survey. Higher Education Research Institute, UCLA, Los Angeles (2009)
5. Frand, J.L.: The information-age mindset: Changes in students and implications for higher education. EDUCAUSE Review 35, 15–24 (2000)
6. Greenberg, E.H., Weber, K.: Generation we: How millennial youth are taking over America and changing our world forever, Emeryville, CA, Pachatusan (2008)
7. Horrigan, J.B.: Home broadband adoption. Pew Internet and American Life Project, Washington, DC (2006)
8. Horrigan, J.B., Rainie, L.: Internet: The mainstreaming of online life. Pew Internet and American Life Project, Washington, DC (2005)
9. Junco, R., Mastrodicasa, J.: Connecting to the net.generation: What higher education professionals need to know about today's students. Student Affairs Administrators in Higher Education (NASPA), Washington, DC (2007)
10. National Center for Education Statistics (NCES), Kridl, B.: The condition of education. Washington, DC: U.S. Department of Education, Office of Educational Research and Improvement. National Center for Education Statistics (2002)
11. Ostrow, A.: Stats: Facebook traffic up 117%, Veoh soars 346% (Nielsen Net Ratings, August 2007) (2007), `http://mashable.com/2007/09/13/nielsen-august`
12. Pryor, J.H., Hurtado, S., DeAngelo, L., Sharkness, J., Romero, L.C., Korn, W.S., Tran, S.: The American freshman: National norms for fall 2008. Higher Education Research Institute, UCLA, Los Angeles (2009)
13. Sax, L.J., Ceja, M., Terenishi, R.T.: Technological preparedness among entering freshman: The role of race, class, and gender. Journal of Educational Computing Research 24(4), 363–383 (2001)
14. Tapscott, D.: Growing up digital: How the net generation is changing your world. McGraw-Hill, NY (2009)
15. Behrstok, E., Clifford, M.: Leading Gen Y Teachers: emerging Strategies for school leaders, Washington, DC, USA. TQ Research&Policy BRIEF, p. 18 (2009)

16. Petuhova, L.E.: Theoretical and methodological background of informational competencies formation of elementary school teachers. In: Doctoral dissertation, specialty 13.00.04 - Theory and Methods of Professional Education, p. 539. The South Ukrainian National Pedagogical University named after K.D. Ushynsky, Odesa (2009)
17. Kushnir, N., Manzhula, A.: Formation of Digital Competence of Future Teachers of Elementary School. In: Ermolayev, V., Mayr, H.C., Nikitchenko, M., Spivakovsky, A., Zholtkevych, G. (eds.) ICTERI 2012. CCIS, vol. 347, pp. 230–243. Springer, Heidelberg (1989)
18. Khandaker, N., Soh, L.-K., Miller, L.D., Eck, A.: Lessons Learned From Comprehensive Deployments of Multiagent CSCL Applications I-MINDS and ClassroomWiki. IEEE Transactions on Learning Technologies 4(1), 47–58 (2011)
19. ICT Competency Standards for Teachers, `http://www.unesco.org/en/competency-standards-teachers`

Three-Subjective Didactic Model

Aleksander Spivakovsky, Lyubov Petukhova, Evgeniya Spivakovska, Vera Kotkova, and Hennadiy Kravtsov

Kherson State University, 27, 40 Rokiv Zhovtnya St., Kherson, 73000, Ukraine
{spivakovsky,petuhova,spivakovska,veras,kgm}@ksu.ks.ua

Abstract. The article shows the transformation of modern didactic model into three-subject one (Student - Teacher - Information and communication pedagogical environment). The new subject's active components that are the most evident in learning process are analyzed. The requirements set to the information and communication pedagogical environment as an educational subject is described. The comparative characteristic of traditional and innovative teaching system components is presented. The authors have made the comparative description of the main university studies forms in different didactic models: object-subject, subject-subject and three-subject training. The experience of the information and communication pedagogical environment creating in the University is described. It is carried out the comparative subjects' behavior modeling during different training forms according to subject-subject or three-subject didactic system. The measurement of each three educational subjects' cogency and their significance in the process of major study operations (collection, processing, storage, transmission) during various training forms as lecture, practice and individual work is presented.

Keywords: Didactics, information society, information and communication pedagogical environment, three-subjective didactics, forms of training organization at university, WebQuest.

1 Introduction

Education is an institute of social experience transmission and human socialization in society. Naturally it depends on the level of social development and labor market needs. Mankind experience increases every day, thus respectively changing the education.

According to UNESCO experts the crisis phenomena in modern education are related with the global mankind problems such as different countries development non-uniformity in the conditions of globalization, persistence of educational systems and knowledge updating permanently growing.

UNESCO has defined the goals in educational system development in the 21st century. According to these goals person's right for education is guaranteed without the dependence on quality education conditions access, such as poverty, gender, location, and ethnic accessory or inability. Therefore one of the primary goals is the creation of information society based on open access to educational and scientific information, educational resources and platforms [1].

The historical retrospective shows the educational system changes at different stages of society development.

Throughout many millennia human speech was the only way for information transfer. The word epoch was characterized by the absence of knowledge accumulation, so a person as the basic source was the information transfer institute.

The information amount increase became the background of writing nascence. Writing unlike the sound speech turned to be the knowledge transfer technology.

The writing invention allowed to transmit voice information to unlimited distance and broadened extremely its existence in time. Beyond dispute, the writing appearance created new additional conditions and opportunities to realize the human culture potential. But at the same time, writing leads to the limitation and narrowing of informational speech content. Writing is a sign system and as a sign it is just the signified representative, so it reproduces only a half of properties and meanings of what is meant. In this case, a word transfers only a part of the properties and meanings contained in the "live" speech. Thus, actually written language loses completely the so-called prosodic information contained in the "live" speech that sounds. The graphic is losing the information that is expressed and transmitted in direct speech.

Year 1450 AD (500 years ago) is marked by a new information technology appearance, the third one in a row. Only then printing technology appeared and we consider it a knowledge distribution technology. This phase is called an era of books. Definitely the appearance of books allowed the creation of an effective and mass educational system, to organize public libraries, to ensure universities development. The appearance of books as a mean of knowledge transmitting, promoted the humankind's achievement of those heights which it has now.

The important consequence of the definite social development turned to be the understanding of purposeful activity for social skills transfer from one generation to another as a connection between two organized activities – teaching and learning, and their concrete reflection in education. The humanity cumulative teaching experience has found expression in didactics, one of pedagogy's sections that examines general theory of education and training. It is believed that the term was introduced by German pedagogue Ratke in his lectures "Rahitiy's summary of didactics and art education", and meant the scientific discipline which studies teaching theory and practice [2].

The efforts to make educational process organized intelligently and purposeful are presented in many Jan Komenskiy's works, especially in "Great didactics" which covers almost all the issues that present the subject of modern pedagogy. Jan Komenskiy was the first one who developed didactics as a system of scientific knowledge, giving the reasoned exposition of principles and rules for children's education. He examined the most important learning theory problems: educational content, teaching principles as visualization, sequence and so on, class-and-lesson system organization, etc.

The object-subject educational relations between the teacher and the student of that time were the most prolonged in pedagogy. The subject is the teacher who educates actively the student who is his object of influence through informative-educational

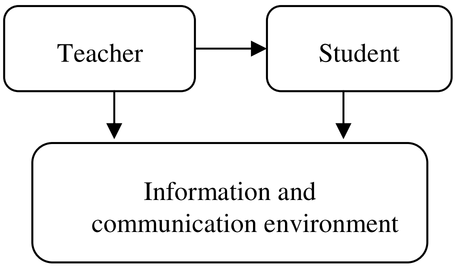


Fig. 1. Schematic model of the object-subject relations

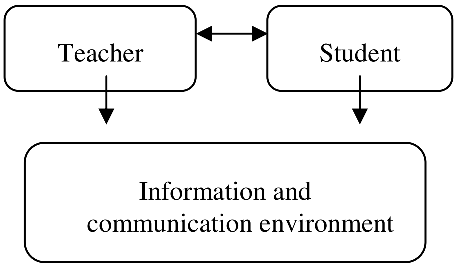


Fig. 2. Schematic model of the subject-subject relations

environment (word, book, equipment). Schematically such didactic relationship is depicted in Fig.1.

Gradually developing the mankind experience has increased to such an extent that a person using only natural abilities was not capable to learn and operate with informational resources. As the result, the person begins to use the technological tools to optimize working process with information. According to it, labor market no longer corresponds to specialists' "conveyor" training, as received knowledge amount becomes quickly obsolete; an employee should make a "decision" in unusual situations. Naturally, the student becomes an equal subject of the educational process. The transformation from the object of educational process into the subject is the result of education democratization, teaching differentiation and individualization distribution. Schematically, the new relationship is depicted in Fig. 2.

The information processes intensification, introduced into science, economics, production, requires the development of a new educational model and the variety of information and communication environments in which people could reveal fully their creativity, develop skills and cultivate self-improvement necessity and the responsibility for their education and development.

The traditional paradigm considered education as younger generation training for work and life by consuming material things created in other areas. The new paradigm foresees education as independent value.

2 Innovative Methodical System

The purpose of creating a new education paradigm is to provide conditions for education, training and development for independent, smart person to satisfy the requirements on a market economy, capable to improve continuously his own level of knowledge and culture, integrated into the global informative space.

Thus, today, we are talking about innovative methodical system which unlike traditional one, corresponds to professional education demands in an informative society. The comparison of the main components of these two systems is presented in Table 1 [3].

Table 1. Comparative characterization of traditional and innovative teaching systems

Name of component	**Traditional methodical system**	**Innovative methodological system**
Learning objectives	Seizure and adoption of educational material. Provide students with knowledge, skills and practice.	Provide students with knowledge, skills and practice. Creation of modern information and communication teaching environment. Purposeful development of creative self-sufficing person. Formation of professional competence, leadership skills, ability to work in group.
Principles of learning	The scientific character principle. The principle of systematic and consistency. The principle of visibility. The principle of studying direction in accordance with issues of education, training and development.	The principle of the activity learning environment. The principle of organic unity between the changing requirements at labor market and conserved features of the educational system. The principle of necessity for continual self-study.
Contents of training	Classical learning, technocratic one.	An integrated approach to fundamental and applied activity aspects of a specialist-to-be.
Study methods	Reproductive, explanatory, illustrative.	Problem-search, research.
Study means	Visual tools. The teacher's word - for knowledge transfer, books, movies, tape, training devices, pictures, maps, tables, machines, devices, models, collections, tools, and historical schemes, charts, diagrams, etc. Technology. Video-recordings, radio and television, filmstrips, slides, transparencies, projectors, televisions.	Facilities. Information and communication technologies. Hypertext, multimedia training materials. Databases for educational purposes. Networking means for videoconferencing and video lecture. An effective system of monitoring training activities. Remote devices for self-work. Computer testing in on-and off-line modes.
Study forms	Lectures, seminars and practical lessons.	Dispute, seminars, conferences, "round table", symposium, debates, colloquium, distance learning, teaching and business games, role-play game.
Control forms	External process operations control within strictly defined rules is dominated. A teacher assessment result (flow, final control) is dominated. Lack of balance between control and self-control. Lack of effective control for individual learning methods of each student.	Strict current control of individual learning of each student by means of testing in on-and off-line mode. Rating control knowledge. Creating an effective environment according to Jean Piaget for easy convenient self-organization that motivates students in learning activities.

In addition, society today has faced the phenomena which require answers:

1. Teacher has lost the monopoly on knowledge;
2. Students have unlimited access to information resources;
3. Availability of qualitatively and quantitatively different ICT competencies of young and older generations.

For that matter, an educational paradigm transforms, which is characterized by the following principles:

- Globalization of knowledge, free access to educational resources;
- Integration of learning resources;
- Organization of global educational audiences;
- WEB-multimedia presentation of educational resources;
- Multilingual educational space;
- Asynchrony of modern models for learning management;
- Harmonization of social and educational environment;
- Formation of social identity of information system;
- Divergence in the implementation of their own educational way.

Thus, an evolution of modern education, information studies, mass computerization of educational establishments, constant upgrade of hardware, and development of computer networks, expanding of personal computerization of society, increasing of software products designed for use in an educational process – these are conditions that create new *information and communication pedagogical environment* (ICPE). This environment constantly and aggressively increases student's motivation to consume content that circulates in it, creating a new didactic model – *three-subjective relations*, which include three subjects of study - students, teachers and an environment.

The notion of "information and communication pedagogical environment" was preceded by such terms as "educational environment", "information environment", "information-learning environment".

The Educational environment is considered as a set of objective external conditions and factors of social objects, which are required for successful functioning of education. This is a system of influences and conditions of personality's formation, as well as possibilities for his development, which are contained in the social and spatial-disciplinary environment.

The Information environment is a set of technical and software storage, processing and transmission of information, as well as political, economic and cultural conditions of realization informatization processes. According to it the Information-learning environment should be considered as a complex of system adapted information influences modeling of the impact of natural sources of information environment of the relevant subject area, and are aimed to create the competencies, that are necessary for self-interaction with the information environment of the subject area. In pedagogical dictionary we find the following definition of it: this is a set of conditions conducive to the emergence and development of the processes of informational and educational interaction between the students, the teacher and means of new information technologies, as well as the formation of cognitive activity of the student on conditions of filling components of the environment (different types of training, demonstration equipment, software, teaching and visual AIDS, etc.) with the substance of a particular training course.

Information educational environment is a systematically organized set of tools data, information resources, interaction protocols, hardware and software and organizational and methodological support, focused on meeting its users' needs.

Thus, the analysis and generalization of the considered definitions allowed making our own interpretation of the notion of "information and communication pedagogical environment", which we understand as a set of knowledge, technological and mental entities that in synchronous integration provide quality mastering the system of relevant knowledge.

Essence we understand as a constant that is stored in the phenomenon in its various variations, including temporary. The subject is impossible to be thought without this inherent quality. The essence is the main thing; this is internal base items of subjects, which determines their deep connections and trends that stand out and are recognized at the level of theoretical thinking.

It should be noted that knowledge entity means the presence of achieved and generated knowledge. The technological entity is the technical, software networking tools of receipt, storage, processing and representation of information. As for the mental entity, it is a set of mental, intellectual, ideological, religious, aesthetic or other characteristics of a nation.

Thus, the information and communication pedagogical environment is a phenomenon that includes human, scientific, technical resources and its content can be implemented in various forms: distance courses, open educational resources, etc.

However, is it legitimate to consider ICPE a possessing equal rights subject for learning along with a teacher and a student?

3 Model of Three-Subjective Relations

Consideration of information and communication teaching environment as a subject, in our opinion, is possible because its components are not only technology but human resources as well, which continuously update them at the constantly growing speed. In this sense, it is necessary to point out an existing qualitatively new learning environment as opposed to which one that was 15-20 years ago. The question deals with the obtaining of today's educational environment the status of an equal partner. Sir Ken Robinson in The Third Teacher (2010) says, "The physical environment of the building is critically important in terms of curriculum" [4].

Within this approach, we implement an important target triangle: a natural integration of teaching, research and labor market needs. After all, ignoring the environment as a subject of education, we will prepare specialists for inadequate reality.

The inevitability of the transition of the education system to consider three-subjective relationship is reflected in the following three stages of didactic changes:

Stage I – the subject-object instruction (a teacher provides students with knowledge). Characterized by one-dimensional linear model, the volume of processed data – megabytes;

Stage II – subject-subject didactics (a teacher and a student are equal competent training partners). Characterized by two-dimensional polylinear model, the volume of processed data – gigabytes;

Stage III – three-subjective pedagogy (Teacher –Student – ICPE). The interaction of all subjects of the learning process (Teacher –Student – ICPE) obeys to the common goal which is formation of a competitive specialist and is characterized by a three-dimensional nonlinear model, the volume of processed data – terabytes.

Thus, we have the right to talk about three-subjective didactics as one of the areas of pedagogical science of the most general regularities, principles and means for organization of studying, providing a firm and conscious assimilation of knowledge and skills within peer relations pupil (Student), teacher (Teacher) and information and communication teaching environment.

It is important to underline that in this process, status and general condition of those who learn and teach and ICPE are constantly changing. In this context, we understand these learning activities with assimilation of knowledge and skills, and teaching – the knowledge message or source of knowledge to students, as well as instruction on ways and methods of work, coordinating training activities, particularly organization of active forms (discussion, round table, project activities, etc.) and monitoring of students mastering knowledge, skills and experience obtaining. Unlike traditional views, we consider, that it's necessary to introduce the one who teaches into the learning process, the changes that are ICPE (for example, by means of publishing of educational materials in the Internet). We have to mention, that new innovative forms of teaching activity are connected remotely or, as they say, distance management software training activities, both in time and space.

Within this definition naturally occurring three-subject relations, which we understand as the continuous and constant (both in space and time) interactions between students, teachers and information and communication pedagogical environment directed for satisfaction of students educational needs (Fig. 3).

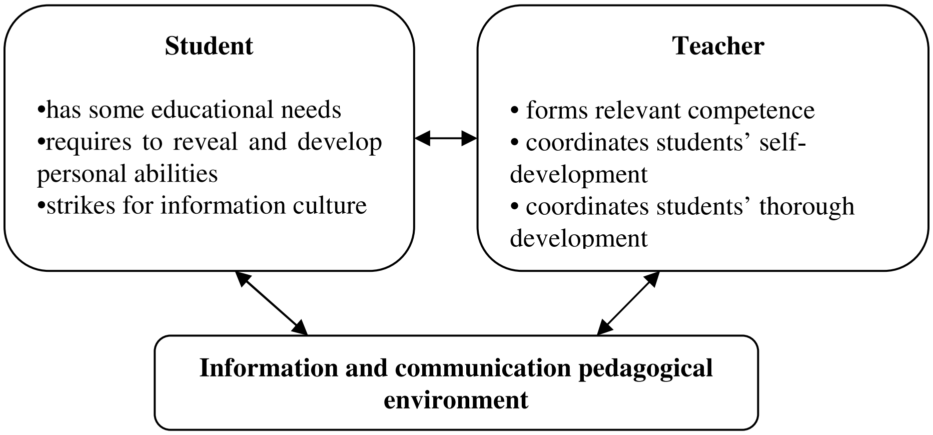

Student
•has some educational needs
•requires to reveal and develop personal abilities
•strikes for information culture
Teacher
• forms relevant competence
• coordinates students' self-development
• coordinates students' thorough development
Information and communication pedagogical environment

Fig. 3. Schematic model of three-subjective relations

As we are examining an environment as a separate element, it's important to mention its operation characteristics, which are the most evident in the learning process:

- environment constantly and more aggressively increases motivation of the younger generation for content use that circulates in it;
- environment provides access to resources at any convenient time;
- environment has comfortable, flexible, friendly, intelligent service, that helps people to find informational resources, data or knowledge they need;
- environment is not a negative emotional one, it corresponds people's demands as much as it is needed;
- environment permanently is filled up with information, data, knowledge with a constantly increasing speed;
- environment offers an opportunity to organize~~,~~ time convenient contacts between any number of people to provide suitable and flexible information exchange between them;
- environment, step by step, standardize, and then integrates the functionality of all previous, so-called traditional ways of receiving, storing, processing and presenting of the required information, data and knowledge to mankind;
- environment undertakes more and more routine operations connected with humans operating activities (which is one of the greatest challenges for humans to expect in the future –"the more commissions - the more responsibility - the greater risks remain without resources");
- environment receives more and more control over the data, and operational mankind's activities [3].

Due to our three-subjective didactics, we can answer the above-listed vital questions connected to modern educational system:

- the teacher's role and place in the new didactic model;
- correlation between virtual and visual forms of subjects' relationships in didactic system;
- development of technological management providing rights for subjects of didactic system to login informational resources;
- organization of modern control systems over learning activities;
- assurance of the organic unity between changing requirements of labor market and conservative educational system potential;
- organization of modern, and most importantly, systematic and constantly active system of re-training and professional qualification upgrade of teachers.

Step by step, it is getting clear that technology that produces modern industry, today, not only affects the technology transfer of knowledge, but in fact, it determines qualitatively new forms of its organization for their mastering. At this stage, we can see the following problems:

1. Heterogeneity of distribution of computer and communication facilities;
2. Huge differences in the process of training and constant re-training of staff, both academic and administrative ones;

3. Inertia of education system;
4. Constantly growing volume of technological renewal of learning environment that includes all the tools, both for teacher and learners;
5. Imposing of different learning paradigms that make substantial confusion in the teachers' presentation of their new role in the process of knowledge transmission, development of abilities and skills;
6. Stereotype of the philistine attitude to pedagogy in whole, as a descriptive section of human knowledge, in which every citizen is a knowledgeable expert.
7. Absence of formal systems which describe different models of learning [3].

Active learning environment contains the following units, which are procedural, substantive and control. The environment begins to play a more important role and assumes some part of teacher's functions. There is no doubt that, certain requirements must be done in the process of setting up a proper learning environment, which will provide active learning environment. Working with the program, both a student and a teacher will be limited by a system of actions, which was laid out in the program, that's why development of the system requirements of ICPE is very important. According to our research, information and communication learning environment can serve as the subject of the educational process if it meets the following group requirements:

1. Hardware requirements: multimedia computers in classrooms are networked with the obligatory access to the Internet resources. In addition, an important aspect creates opportunities to access educational electronic resources (Wi-Fi technology) for students in any convenient place, for example, library, dormitory, canteen etc.
2. Software requirements: software environment should resolve security issues (registration, personalization, delineation of access rights to get to resources), to be integrated (all educational components should be in its natural form), easy for exploitation, filling and modification, to provide opportunities for interaction, communication, monitoring for learning process, to contain an output mode out of the complicated situations (expert mode), to offer opportunities for distance learning (on-line and off- modes).
3. Academic requirements refer to methods of filling information and communication teaching environment.
4. Social demands. Special attention should be paid to a specified group of claims which, in our opinion, contains cultural, ethical and legal aspects, because users of information and communication pedagogical environment create some community. First of all, it is about the rules of communication in the network and use of the reworks of other authors.
5. Requirements to Human Resources. Construction of the educational process on the basis of information and communication technologies implies specialists- programmers and accordingly well-trained teachers.

ICPE correspondence to these requirements can be achieved by using management system of the quality of educational information resources [5].

One of the variants for the construction of above mentioned environment in the University was the development of a comprehensive informatization program in Kherson State University. The program included:

- improvement of control processes;
- perfection of educational process;
- development of information structure (technical and programmatic aspects), including establishment of a system of Internet access from any place in the University using Wi-Fi technology, which creates the conditions for a comfortable learning and significantly expands the possibilities of teachers professional activity organization;
- improvement of personnel qualification.

Expected results of program execution:

1. Adequate qualifications of the staff in the field of information and communication technologies (in terms of the International computer driving license).
2. Adequate qualifications of the graduates in the field of information and communication technology.
3. Automated control of the educational process on the levels of University - Department.
4. Quality access to internal and external e-mail educational and methodological resources for the teachers and the students.
5. Teachers and students' automated control of their own work.
6. Licensing of the used software.

An important area is the improvement of the educational process, because introduction of new subject of learning process naturally transforms existing elements of training, including forms of teaching at higher educational establishments. Today, educational resources are open and distance forms for studying are actively developing and integrating into traditional forms of teaching: lectures, workshops, laboratory classes, independent, individual work of students, forms of control. Let's try to analyze basic traditional forms of training organization in different didactic models (Table 2).

To summarize the experience of teacher's educational work organization with ICT we had conducted observation of training sessions, question students and teachers, analyzed the advanced pedagogical experience of universities.

The learning environment created on different distant platforms (E-Learning, Moodle, Stellus etc) is most widespread. Such resources have many educational and developmental opportunities: the transfer of public-pedagogical, didactic and methodological knowledge, monitoring of students' knowledge and skills. Therefore they can operate distantly as well as be easily integrated in traditional training.

Multimedia presentation creates support for teacher's conducting lecture. It allows the lecturer present clearly and logically teaching material, its key aspects. Due to such form of teaching lectures the student has the opportunity to get the necessary information before the lecture, during it, or preparing independent work within the

Table 2. Forms of learning in didactic models

Subject-object study	Subject-subject study	Three-subject study
Teacher		
The source of educational information is a teacher; students are forced to put down a limited amount of information, static visibility is used additionally.	A teacher presents difficult educational material, students selectively put down the information that is necessary for each personally, use additional sources, including the Internet. Dynamic visibility is dominative.	A teacher and students in the debate form discuss problematic issues due to free access to open lecture and other information sources. Students write down the required information at will.
Practice		
Reproductive methods of teaching material development are used.	Part-search training methods prevail.	Search and creative methods are directed at forming experience of training materials, particularly under unusual circumstances.
Independent work		
It consists of lectures, practical exercises execution.	Studying unwrought amount of teaching material.	The main part of teaching material is studied individually.
Forms of control		
A control requires presence of a teacher who relates a student's knowledge with the volume of lectures material.	Students' readiness to use received knowledge in condition of life situation is also under control.	Monitoring can be conducted without teacher's presence, and the result – unconventional approach and creative thinking of students are estimated.

tasks provided by the teacher. Attention should be paid to the fact that the presentations allow the lecturer to focus on the main issues that need to be addressed to. Thus, traditional model, where the lecturer reads and the student writes down, transforms onto the model, where the lecturer discusses and organizes discussions around key questions during the classes.

An innovative algorithm of getting knowledge and skills is formulated due to natural integration of such traditional forms of training as lecture and individual work:

1. At the stage of preparing before the lecture the student through the corresponding website studies teaching material that the lecture contains. Student's main task at this stage is to prepare the questions which he will formulate during the classes.
2. The algorithm of the lecture consists of a discussion of the key issues that the teacher considers important and discussion around the questions that students prepared. The logical ending of the lecture is formulation creative tasks for further independent students' work.

3. Independent students' work after the lecture primarily dedicated to the search and processing of information necessary for solution of the problems which were aroused during the lecture [3].

Thus, the innovative algorithm not only naturally combines different forms of educational process organization, but also attracts the students with active forms of interaction: with each other, with the teacher and with the environment, that can provide them with necessary resources of managing professional competencies.

Electronic testing provides the most complete and objective real-time monitoring of student's knowledge.

Thus, the organization of traditional education with the involvement of the information and communication pedagogical environment has the following features:

- it intensifies students' work through the use of ICT;
- it increases students' interest in education, implements the knowledge control as during classes, so after finishing of each topic of the course;
- it provides students' access to full complex of training materials and tasks of independent work with the use of distance learning technology.

The combination of remote and traditional forms of training modifies the main stages of learning.

1. The student gets acquainted with the educational materials of the course at convenient time, because the access is opened via the Internet. The student looks through the text of the lecture highlights the theoretical issues requiring refinement. He gets acquainted with the practical tasks and by necessity studies the material for self-learning.
2. The lecturer conducts the lecture with discussion of the key issues that he considers important and the questions that students prepared.
3. During the practical training the students and the teacher evaluate advanced tasks. The students send practical tasks to the teacher by e-mail. So, during practical classes are formed students skills of applying theoretical knowledge in practical activities.

The defining element of the educational process is consultation, where the students remotely receive teacher's answers on specific questions or clarifications or methodical recommendations for managing and improving practical tasks.

Thus, the combination of traditional and distance forms of professional education helps to enhance students' motivation to achieve new knowledge and skills, to be systematic and strength during assimilating significant volume of learning material. It also releases training time for testing the practical skills during classes.

The use of ICT gives ground for innovations, which influences the improvement of the quality of independent work and ensures the quality of specialists' training.

Development of network technologies and telecommunications, especially the Web-technologies have contributed to the development of project learning technologies. Among them a WebQuest as research activity is widely used.

WebQuest is an inquiry-oriented lesson format in which most or all the information that learners work with comes from the web. The model was developed by Bernie

Dodge at San Diego State University in February, 1995 with early input from SDSU/Pacific Bell Fellow Tom March, the Educational Technology staff at San Diego Unified School District, and waves of participants each summer at the Teach the Teachers Consortium.

Since those beginning days, tens of thousands of teachers have embraced WebQuests as a way to make good use of the internet while engaging their students in the kinds of thinking that the 21st century requires. The model has spread around the world, with special enthusiasm in Brazil, Spain, China, Australia and Holland [6].

WebQuests of either short or long duration are deliberately designed to make the best use of a learner's time. There is questionable educational benefit in having learners surfing the net without a clear task in mind, and most schools must ration student connect time severely. To achieve that efficiency and clarity of purpose, WebQuests should contain at least the following parts:

1. An introduction that sets the stage and provides some background information.
2. A task that is doable and interesting.
3. A set of information sources needed to complete the task. Many (though not necessarily all) of the resources are embedded in the WebQuest document itself as anchors pointing to information on the World Wide Web. Information sources might include web documents, experts available via e-mail or real-time conferencing, searchable databases on the net, and books and other documents physically available in the learner's setting. Because pointers to resources are included, the learner is not left to wander through webspace completely adrift.
4. A description of the process the learners should go through in accomplishing the task. The process should be broken out into clearly described steps.
5. Some guidance on how to organize the information acquired. This can take the form of guiding questions, or directions to complete organizational frameworks such as timelines, concept maps, or cause-and-effect diagrams.
6. A conclusion that brings closure to the quest, reminds the learners about what they've learned, and perhaps encourages them to extend the experience into other domains.

Some other non-critical attributes of a WebQuest include these:

1. WebQuests are most likely to be group activities, although one could imagine solo quests that might be applicable in distance education or library settings.
2. WebQuests might be enhanced by wrapping motivational elements around the basic structure by giving the learners a role to play (e.g., scientist, detective, reporter), simulated personae to interact with via e-mail, and a scenario to work within (e.g., you've been asked by the Secretary General of the UN to brief him on what's happening in sub-Saharan Africa this week.)
3. WebQuests can be designed within a single discipline or they can be interdisciplinary. Given that designing effective interdisciplinary instruction is more of a challenge than designing for a single content area, WebQuest creators should probably start with the latter until they are comfortable with the format.

Longer term WebQuests can be thought about in at least two ways: what thinking process is required to create them, and what form they take once created.

Thinking skills that a longer term WebQuest activity might require include these (:

1. Comparing: identifying and articulating similarities and differences between things.
2. Classifying: grouping things into definable categories on the basis of their attributes.
3. Inducing: inferring unknown generalizations or principles from observations or analysis.
4. Deducing: inferring unstated consequences and conditions from given principles and generalizations.
5. Analyzing errors: identifying and articulating errors in one's own or others' thinking.
6. Constructing support: constructing a system of support or proof for an assertion.
7. Abstraction: identifying and articulating the underlying theme or general pattern of information.
8. Analyzing perspectives: identifying and articulating personal perspectives about issues [6].

The forms that a longer term WebQuest might take are open to the imagination, since these are some ideas:

1. A searchable database in which the categories in each field were created by the learners.
2. A microworld that users can navigate through that represents a physical space.
3. An interactive story or case study created by learners.
4. A document that describes an analysis of a controversial situation, takes a stand, and invites users to add to or disagree with that stand.
5. A simulated person who can be interviewed on-line. The questions and answers would be generated by learners who have deeply studied the person being simulated.

Putting the results of their thinking process back out onto the internet serves three purposes: it focuses the learners on a tangible and hi-tech task; it gives them an audience to create for; and it opens up the possibility of getting feedback from that distant audience via an embedded e-mail form.

One of the advantages of a web quest is to save student's time, taking into account the fact that the teacher himself directs the student's activities providing a list of Internet addresses from which the students receive needed data to implement their project. The result of the WebQuest can be a presentation or web page that can be posted on the Internet and provide an opportunity for all to keep their opinions, suggestions, feedback and even appropriate changes.

Web quest is a complex task, and therefore assessment of its implementation should be based on several criteria, which are focused on the task type and the form of result presentation.

B. Dodge recommends using from 4 to 8 criteria, which can include assessment of the following aspects:

- research and creative work;
- argumentation;
- the originality of the work;

- working skills in the micro group;
- oral presentation;
- multimedia presentation;
- written text and other [6].

The use of Web technologies, including Web-quests in the educational process, in our opinion, plays a significant role in the development of students' cognitive activity, quality of their knowledge, promotes the development of independent learning skills, and provides the learners except the core knowledge with competencies.

Thus, the experience in organization teachers' work proves the integration of learning through a combination of traditional and distance forms of professional training organization. So, Web technologies have changed the educational paradigm and effect significantly on the techniques, methods, forms and means of University training. This integrated training modifies the role of the teacher who becomes coordinator of student's personal professional growth through filling ICPE with educational and methodological materials, carrying out students' consultations by e-mail, controlling their activity and performance.

4 Measurement of the Importance of Training Subjects

However, is ICPE a significant, important subject of learning in practice of University operation? The theoretical conjectures study was conducted at the Faculty of preschool and primary education of Kherson State University in order to confirm or refute it. The research required a questionnaire of future primary education teachers, as they acquire an integrated system of philology, humanities, exact, natural and artistic sciences, which, in our opinion, reduce a risk of results' obtaining only from certain cycle of training. The main task of the questionnaire is to evaluate the significance of subjects of modern educational process, including ICPE.

Determining the validity of each of the three subjects of the educational process was made by expert evaluation method. 27 qualified experts (university teachers, graduate students, methodologists) joined the independent expert committee.

To define a point of evaluation for each subject Delphi method (for members of the expert committee conditions for an independent individual work were created) was used [7]. Maximum and minimum estimates depended on a number of subjects, in which, there are three. Thus, minimum score for one of three components –1 point, an average score – 2 points and maximum – 3 points. Then, the statistical processing of the results, which were presented to experts for final approval, had been conducted. The cycle of expertise was repeated three times.

We present the results of independent expert Commission work. 12 experts put 1 point for the subject "teacher", 8 experts – 2 points, 7 experts – 3 points. The total number of points for the given subject is:

$$\sum_1 = 1\times 12+2\times 8+3\times 7 = 49$$

The total number of points for the subject "student" is:

$$\sum_2 = 1\times 2+2\times 10+3\times 15 = 67$$

The number of points for the subject “ICPE” is estimated similarly:

$$\sum_3 = 1\times12+2\times11+3\times4 = 46$$

The total number of points for three subjects is 162 points.

The appropriate number of subject’s points is divided into the total number of points to define the significance (*V*) of each subject:

$$V_1 = \frac{49}{162} = 0{,}30; \qquad V_2 = \frac{67}{162} = 0{,}41; \qquad V_3 = \frac{46}{162} = 0{,}29.$$

Below are the results of an independent expert committee (Table 3).

Table 3. Determination of cogency of training subjects (*V*)

Subjects of the educational process	Number of points			$\sum$	*V*
	1	2	3		
Teacher	12	8	7	49	0,30
Student	2	10	15	67	0,41
ICPE	12	11	4	46	0,29

According to the results of expert reviews cogency *V* (in fractions of a unit) for each of these three specified subjects, according to experts, is approximately the same, with a slight advantage “student” (0,11 larger compared to “ICPE” and 0,12 larger compared to the “teacher”).

Results summarizing the data are shown in Fig. 4.

214 students, as the most significant subject of didactic system according to experts’ definition, were asked to rate on a 10-point scale the importance of the three subjects of the educational process: students, a teacher and ICPE – in the process of operating with information (collecting, processing, storing, transmission) in various forms of training organization: lectures, practical classes and independent work. To do this, students were asked to determine the importance of each component of didactic models: Student – Teacher – ICPE a five-point scale (1, ..., 5).

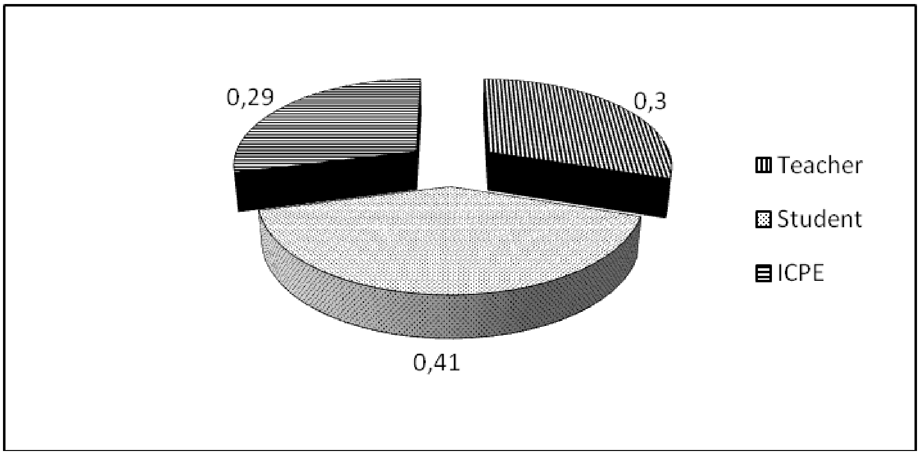


Fig. 4. The importance of the subjects of the educational process

Results of the student's questionnaire are shown in Table 4.

Assessment E_{ij} ($i, j = 1,2,3$) for each i-component of the didactic system operations in terms of j-forms of training organization is given by (1):

$$E_{ij} = K_{ij1} + K_{ij2} + K_{ij3} + K_{ij4}, \tag{1}$$

Where E_{ij} – total score in terms of transactions weight, K_{ijk} – i-score weighting components didactic system, j-teaching forms and k-rate transactions, %.

Table 4. The results of the student's questionnaire

Components		Student		Teacher		ICPE	
Form of training organization	Indicators						
		Points	%	points	%	points	%
Lecture	Collecting	1182	5,5	2187	10,1	334	1,5
	Processing	2151	10,0	2630	12,2	2411	11,2
	Storing	2625	12,2	1455	6,7	2402	11,1
	Transmission	830	3,9	2004	9,3	1355	6,3
	∑	6788		8276		6502	
E			31,6		38,3		30,1
Practice	Collecting	1674	7,7	2006	9,2	1235	5,7
	Processing	1885	8,7	3121	14,3	815	3,8
	Storing	1241	5,7	1663	7,6	2421	11,1
	Transmission	2178	10,0	1198	5,5	2322	10,7
	∑	6978		7988		6793	
E			32,1		36,6		31,3
Independent work	Collecting	2214	9,7	2119	9,3	2033	8,9
	Processing	2366	10,4	2882	12,6	1938	8,5
	Storing	2154	9,4	2007	8,8	2013	8,8
	Transmission	994	4,4	384	1,7	1703	7,5
	∑	7728		7392		7687	
	E		33,9		32,4		33,7

The overall assessment of V_i ($i = 1,2,3$) for each component of the didactic system is given by (2):

$$V_i = (E_{i1} + E_{i2} + E_{i3})/3. \tag{2}$$

Let's analyze the results.

It is generally known that, lecture – is the main form of teaching, prepared for the adoption of theoretical material. Table 4 shows that while gathering information during lectures (17,1%), the most significant entity of the educational process is a teacher (10,1%), 5,5% of operation is performed by a student, 1,5% – ICPE. It's explained by identification of the content and material of lectures, in its selection, the main role is occupied by a teacher, but the lecture provides not passive acceptance of students' knowledge but their active involvement into the learning process, preparation for lectures, which is provided with ICPE use. We should note that active cognitive activity of students during lectures is possible for basic training, which includes familiarization with the theme of the lecture and its plan, the main content of the theme for the tutorial, content repetition of the previous themes etc.

According to the survey results, information processing on the lecture (33,4%) subjects' contribution is approximately the same: 12,2% – Teacher, 11,2% – ICPE, 10% – Student. This is because the teacher coordinates educational information processing, and an active entity involved in this process may be a student. ICPE activity due to a shift in emphasis onto the use of methods and means of processing students – from note-taking information material: full or theses synopsis for computer processing of the information received.

Storing educational information of the lecture (30,0%) between the subjects of the educational process was distributed: 6,7% – Teacher, 11,1% – ICPE, 12,2% – Student.

According to received questioning results, transfer of educational information of the lecture (19,5%) is implemented by a teacher (9,3%) and ICPE (6,3%), although students (3,9%) provide additional information, interesting facts and problematic issues. The task of the teacher, at this stage, is to transfer the adapted information disclosing a nature of scientific concepts, genesis of scientific theories, ideas, etc.; aggregated information is transmitted using pedagogical software, e-presentations, etc.

The results of the distribution of three-subjects training, in the process of operation with information during a lecture are presented in Fig. 5.

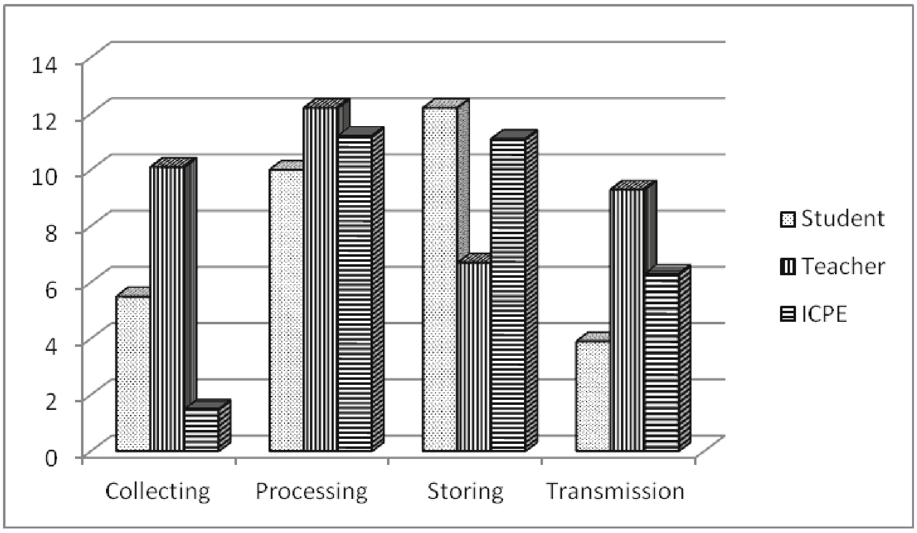


Fig. 5. The significance of the subjects of the educational process during operations with information on the lecture

Thus, the most important subject of the lecture organization according to students is a teacher (38,0%), a student (31,6%) and ICPE (29,0%) provide processing and preservation of educational information. In general, during the lecture principal place of work with information take processing operations (33,44%) and storing (30,0%), followed by transfer (19,5%) and collection (17,1%).

Let's analyze the data from Table 4 according to the importance of the subjects of study during the preparation and conduction of practice. As it is known, practical lesson is a class that involves organizing teacher's detailed study of individual theoretical positions discipline and development of skills in their practical application by individual performance to related tasks.

Analyzing the data in Table 4 concerning the collection of information (22,6%) during the practical sessions was revealed that students' contribution is 7,7%, but teacher's and ICPE respectively 9,2% and 5,7%. Comparing with a lecture, students' activity increased by 2,2%, due to test theoretical knowledge of students, development of skills based on acquired knowledge and, as a result, a detailed collection of information for further processing.

In the process of collection (22,6%) and processing (26,8%) of the information during preparation and practice, according to students, a teacher and a student have the greatest significance, which is confirmed by received data: respectively (9,2% and 14,3%), (7,7% and 8,7%). This is due to students interest in learning, deepening and refinement of knowledge, developing skills, primary accumulation of experience, professional motivation and, consequently, activity in learning. Significance of ICPE is gradually increasing, as it is evidenced by statistics data, in storing and information transmission: 11,1% and 10,7%.

Visually, the results are presented in Fig. 6.

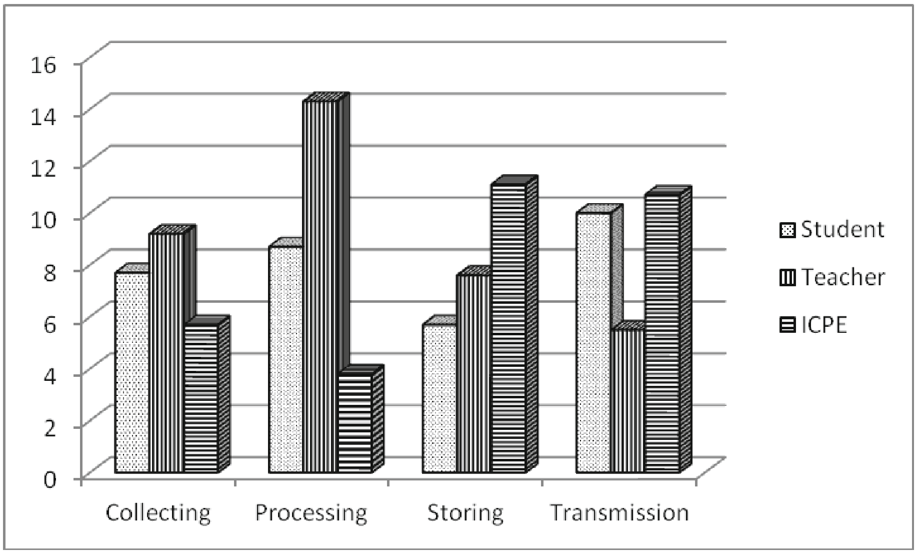


Fig. 6. The significance of the subjects of the educational process during operations with information during the practice

Thus, a teacher is the most important subject of practical training organization (36,6%), although a student (32,1%) and ICPE (31,3%) are equal subjects. Generally during practical classes the importance of operations with information is as follows: processing (26,8%), transmission (26,2%), storing (24,4%) and collecting (22,6%).

Let's analyze the importance of training subjects in the process of student's individual work organization. As you know, independent work of a student is a primary mean to master academic material at a time, free from mandatory training sessions.

Leading role in collection, processing, storing and transmission of material belongs to a student (33,9%), according to the relevant data: 9,7% 10,4% 9,4% 4,4%. This is primarily due to the students' understanding of the importance of having theoretical knowledge, development of skills, and accumulation of their own professional experience and, as a result, operations with the information according to the educational goals. Practice has proved that the most active in independent work will be a student who is more motivated to master for his future profession. The result of questioning is the importance of teachers is on average 32,4%, ICPE – 33,7%, and an independent educational-cognitive students' work is task-teacher, under his leadership, but without his direct involvement but widespread use of information and communication teaching environment. The importance of business education in independent work is shown in Figure 7.

So, according to students' definition, important subjects of independent work are all three components of the didactic system – Student (33,9%) and Teacher(32,4%), and ICPE (33,7%). During the independent work with information, transaction processing occupies a principal place (31,5%), then collection (27,9%), followed by storing (27,0%) and afterwards transmission (13,6%).

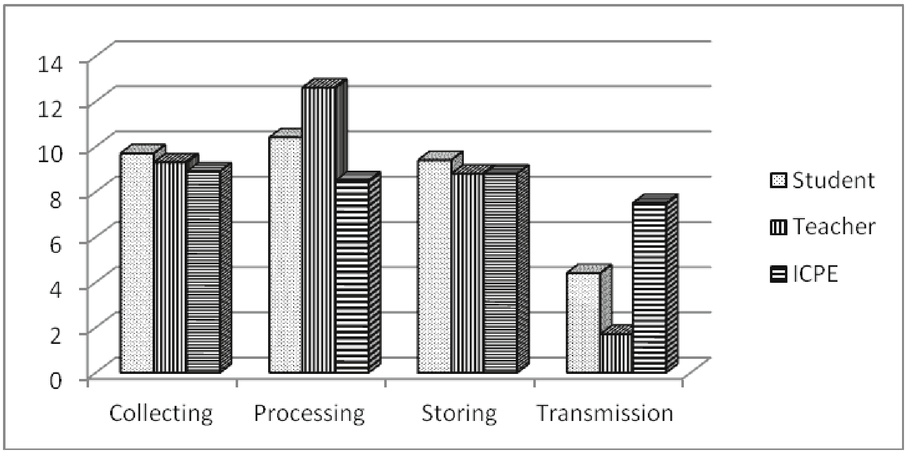


Fig. 7. The significance of the subjects of the educational process during information operations in class work

The analysis of the results of the survey showed that according to students' all three constituents are important and significant components of the didactic Student – Teacher – ICPE system. This is the statistics of indicators weight components: Student

(V_1 = 32,5%), Teacher(V_2 = 35,8%), ICPE (V_3 = 31,7%). It's important to underline that received students' survey data correlate well with similar data of experts' assessment of component's importance of the didactic system. The student is a significant subject of a teaching process at the University as learning outcomes largely depend on its intended acquisition of trade. Proof that serve high levels of significance to students in maintaining and processing information during lectures, collecting, processing and transmitting information during practice, collecting, processing, storing information during independent work. Major indicators of the importance of the teacher can be seen during collection, processing and transmission of information during the lecture, which is determined by specifics of this type of training sessions – teaching theoretical material. During practical sessions and independent students' work the teacher has the greatest indicators of the importance of information processing. ICPE's significance is high during operations with information and practical lessons in the process of information preserving, as for in-class, ICPE acts as an equal-right subject of the educational process. This is because ICPE provides access to informational resources at any convenient time, quickly and easily enables to find all necessary information, provides flexible and convenient information sharing between students. However, according to average data ICPE significance inferior teachers' importance, as a teacher manages the studying-cognitive students' activity, coordinates their independent improvement of knowledge, skills and abilities. It should be noted, that due to ICPE systematical involving in learning process the role of it as new subject will improve gradually because of improving learning outcomes.

Summary results of the survey are presented in Fig. 8.

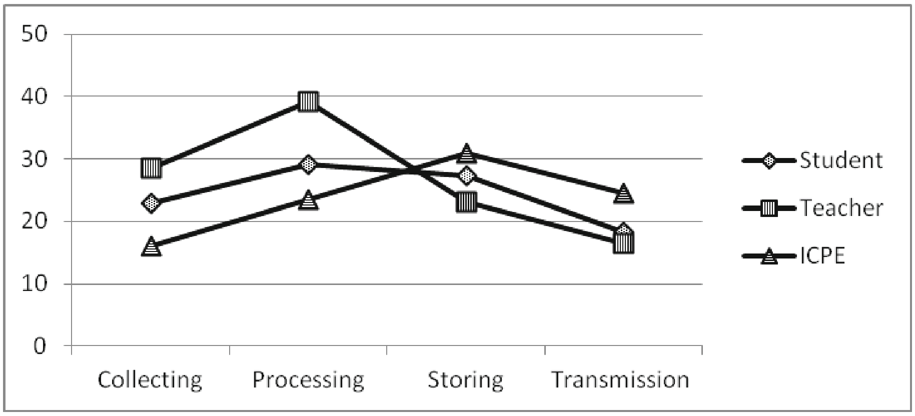


Fig. 8. The importance of the subjects of the educational process during operations with information

5 Conclusions

Thus, the analysis of the scientific literature, theoretical and experimental study on transforming learning into different didactic models showed that information and communication pedagogical environment is an important subject in the process of learning at the University; ICPE transforms traditional subject-subject model of study into three-subject one, directly affecting and slightly changing the role and function of other subjects of study, partly fingering their functions itself, particularly in transient conditions while performing operations with information on various forms of training.

References

1. Building Knowledge Societies, `http://en.unesco.org/themes/building-knowledge-societies`
2. Harvey, L., Kendall, R.A.: Sugimoto: The Didactic Theory of Wolfgang Ratke. California State University (1976)
3. Petukhova, L.E.: Theoretical bases for training primary school teachers in information and communication teaching environment: scientific monograph. Ayilant, Kherson (2007)
4. The Third Teacher: 79 Ways You Can Use Design to Transform Teaching & Learning by OWP/P Architects, VS Furniture, Bruce Mau Design, Abrams (2010)
5. Kravtsov, H.M.: Design and Implementation of a Quality Management System for Electronic Training Information Resources /. In: Ermolayev, V., et al. (eds.) Proc. 7-th Int. Conf. ICTERI 2011, Kherson, Ukraine, May 4-7, vol. 716, pp. 88–98. CEUR-WS.org (2011) ISSN 1613-0073, CEUR-WS.org/Vol-716/ICTERI 2011-CEUR-WS-paper-6-p-88-98.pdf
6. Dodge, B.: A WebQuest about WebQuests. In: Byles, B., Brooks, S. (eds.), `http://www.memphis-schools.k12.tn.us/admin/tlapages/wq_wq2.htm`
7. Rowe, G., Wright, G.: Expert opinions in Forecasting: The Role of the Delphi Technique. In: Armstrong, J.S. (ed.) Principles of Forecasting - A Handbook for Researchers and Practitioners, pp. 125–144. Kluwer Academic Publishers, Boston (2001)

Emerging Technologies for Training of ICT-Skilled Educational Personnel

Mariya Shyshkina

Institute of Information Technologies and Learning Tools
of the National Academy of Pedagogical Sciences of Ukraine
marple@ukr.net

Abstract. The article intends to explore and estimate the possible pedagogical advantages and potential of emerging learning technologies application for enhancing ICT competence of educational personnel. The notion of cloud-based learning technology of open education is considered. The state of the art of cloud computing services application at the educational institutions in Ukraine is described. The notion and main categories of ICT for learning educational personnel are outlined. The problems of ICT-skilled educational personnel training in particular for public administration are considered. Holistic approach to personnel training within the cloud-based learning environment is proposed and perspective ways of its application for educational institutions are estimated.

Keywords: Learning technologies, personnel training, cloud computing, holistic approach.

1 Introduction

The Age of IT has brought progress to the area of ICT and network technology that gives new insights into the problems of learning development, showing a need to train adult professionals in advanced ICT, especially to train teachers, lectures and management stuff of educational institutions. There is a need in developing and estimating learning technologies supported by emerging ICT, such as cloud computing, mobile tools and services, network infrastructures.

Cloud computing technology (CC) is to create a high-tech learning environment of educational institution, enhancing multiple access and joint use of educational resources at different levels and domains. On this basis it is possible to combine corporate resources of the university and other on-line tools, adapted to learning needs, within a unite framework. For this aim a set of instrumentation tools for cloud-based learning resources collection, elaboration and design, holistic models of learning environment and a specialist, and a system of methodological and technological support for the learning process development within the cloud-based settings should be created.

The *purpose of the article* is to identify innovative pedagogical approaches and conceptual models of ICT-skilled educational personnel training arising within the high-tech learning environment.

2 Problem Statement

The problem of training of qualified educational and management personnel as well as teachers oriented on ICT based learning can nowadays hardly be taken independently from the processes of the innovative development of educational space formed within the school, region or globally [1]. In this regard, there is a need for fundamental research focusing on the possible ways of environment of educational institutions formation. It should take into account, the trends of improving ICT facilities while searching for new engineering technological decisions and new pedagogical and organizational models [1], [2]. The main focus is on shifting from mass introduction of separate software products, to an integrated and combined framework which supports distributed network services and cross-platform solutions.

Emerging technologies of information and communication networks give a way for development of innovative learning technologies for personnel education. This is a promising direction for rising of a field's human potential. So the organization and development of learning environment, search for new models for specialist training becomes a matter of interest [11].

There is **a problem** of formation of educational, social, professional and personal competencies of a professional that enable him (her) to live and work in a high-tech environment, to use learning resources and services, to achieve with its use the best pedagogical effect and to gain maximum learning potential of ICT. This may be achieved if to take up innovative ways of resources delivery, in particular, by means of cloud computing [2], [8], [12]. **The idea** is simply to explore pedagogical approaches and new learning opportunities appearing by virtue of cloud-based tools and services.

3 Cloud-Based Learning Technologies of Open Education

Cloud computing (CC) is an important trend of development of tools and services of modern open learning [3, 8, 12]. It gives new opportunities for individualization and differentiation of the educational process, its flexible adaptation to individual characteristics of learners, while changing the entire notion of e-learning organization [1, 2]. Obviously, under intensive development of information society the technological platform of learning organization processes in most educational systems has to be based on the leading tools and services of the very information society among which there are just cloud computing technologies as a new stage of development of information and communication networks [3]. This leads to the notion of cloud-based learning technology of open education

The *cloud-based learning technology* is a computer based part of learning technology, aimed at delivery of different didactic tasks. It reflects the model of the learning methodic structure (being a set of relations between the participants of the learning process and the elements of content and other components of computer based learning environment), supposing application of computer-based learning tools, information communication networks and electronic resources mostly and principally based on cloud computing [1].

Unfortunately, the dissemination and application of modern methods and tools of e-learning is characterized by a number of negative trends [11], including such as:

- Deepening the gap between the level of development of modern information technological platforms of e-learning and current supply of educational institutions with the facilities and services of information communication networks;
- Deepening the gap between the needs of modern society in improving the quality of education and outdated technologies to train personnel and to supply educational services.

In this respect just human resources of the university need the most intent attention. It requires new types of skills and competencies which graduates often lack of. These skills include leadership, ability to approach a problem holistically, and the ability to critically evaluate achievement and self-assessment [9], [11]. Just the shortage of highly qualified stuff leads to lack of strategic approach to technological infrastructure design and system solutions that are among the reasons of technological lag and insufficient quality of educational services.

There is a branch of pedagogical sciences and practices dealing with theoretical and methodological problems of ICT in education use, psychological and pedagogical substantiation of these processes, elaboration of ICT tools and resources for providing functioning and development of educational systems. There should be specialized personnel to insure the processes of implementation, introduction and development of ICT-based learning technologies. So the notion of ICT for learning educational personnel namely personnel oriented for wide adoption of ICT arises.

By *ICT for learning education personnel* we mean those who are concerned to organizational-normative, social-economic, educational-methodical, scientific-technical, production and management processes, aimed to satisfy information, processing and telecommunicate needs (other needs, connected to ICT methods and tools facilities) of the learning process participants, and also those, who support and manage this process [1]. The key categories of such personnel are teachers, lectures, management stuff (among them the leaders of ICT-departments) and public administrators aimed at wide spread and use of ICT for learning. ICT skills of such personnel become the central point of their training in view of dealing with emerging technologies [11].

So as to examine CC as e-learning platform for personnel training, it is necessary to take into account some didactic, methodical, technological, organizational and other application aspects, to make good decisions as for its educational benefits and most fruitful trends of application. For this aim the Joint laboratory "Cloud Computing in Education" (CCELab) was created on the basis of Kryvyi Rig National University and Institute of Information Technologies and Learning Tools of the National Academy of Educational Sciences of Ukraine in 2012, http://cc.ktu.edu.ua/. The main purpose of the CCELab are methodological and experimental research of emerging e-learning technologies and exploration of different aspects of cloud computing application for education and personnel training. The virtual laboratory CCELab is available at http://www.ccelab.ho.ua to reflect currents state of research and exchange opinions.

4 State of the Art of Cloud Computing Technologies Application

To reveal the state of the art of cloud based learning environment development and the rate of cloud-based services use by educational personnel in Ukraine the survey was made within the framework of the International internet-seminar “Cloud Computing in Education” held by CCELab in December, 2012, http://cc.ktu.edu.ua/report.html. At this seminar there were 127 participants from the 54 educational institutions from 22 cities of 18 regions of Ukraine. As the participants were those, concerned with the problems of CC, so they were those, being well aquatinted to the modern trends of technological development, and their institutions were well equipped and oriented for the use of advanced ICT.

To the question: **“How can you characterize the learning environment of your educational institution?”** the responses showed that 48% of participants considered it to be computer oriented (COE), 36 % - to be computer integrated (CIE); and 14% - to be personalized, i.e. cloud-based (PE). The results are presented at **Fig. 1.** (the entries are excluded).

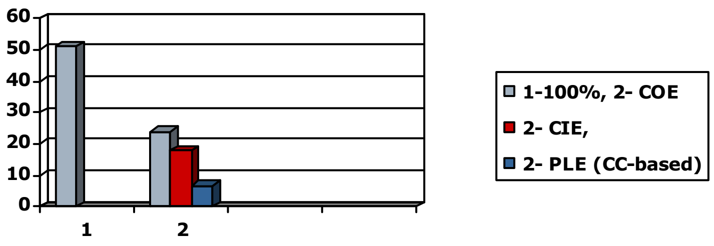


Fig. 1. The results of the survey for CC application at the institutions of higher education in Ukraine

For the question: **“For what activity types do you use cloud services?”** the results were the next, (**Fig. 2,** the entries are not excluding):

Organization of collaborative learning – 50 %
Learning resources management and delivery - 42 %
Electronic document processing – 30 %
Office applications – 24 %
Learning, professional communities – 28 %
Web-conferences, webinars – 34 %
Electronic libraries – 18 %
Data retrieving – 13 %

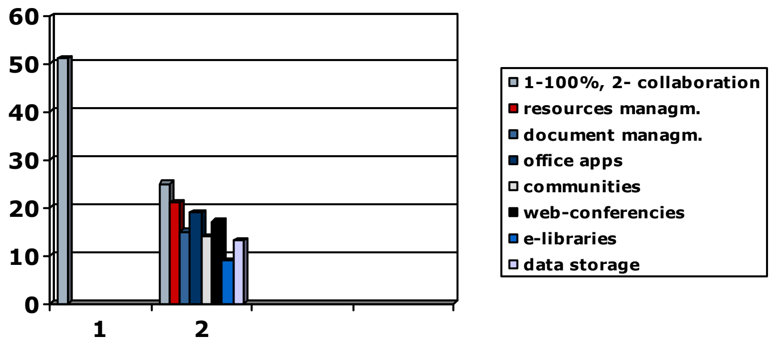


Fig. 2. Application of cloud-based services at the educational institutions in Ukraine

As it appears from the survey, the cloud-based services are widely used in educational institutions still its application is not systematic, it is not organized into the unite system, it is not consciously and purposely oriented to pedagogical aims. Thus there is a current need for upgrading of ICT competence of educational personnel, mainly those engaged with providing educational systems with emerging ICT, in particular, public administration personnel.

5 Education and Training of Public Administration Personnel

Public administrators are public servants working in public institutions, departments and agencies [10]. Specifically, they are concerned with "planning, organizing, directing, coordinating, and controlling government operations" [6]. Specific sphere are public servants for education management. For such personnel to be efficiently trained, the need to develop novel approaches arises, as this sphere is mostly concerned with multi-disciplinary knowledge and requires skills on the merge of training, learning and management. Due to the fact that most pedagogical innovations are also based on ICT the need in the sphere of education management also arises.

There are significant needs in IT competent specialists in the sphere of public administration. Without ICT competence or competence in ICT for learning, problems with their adaptation at the workplace arise, as do problems with the necessity of additional and often profound training almost immediately after hiring. In some cases, a vague idea of future graduates about the real problems and conditions of work with innovative ICT infrastructures and ICT-based tools leads to lack of commitment to practical solutions of work situations thus to a low level of innovative inclusion.

Formation of the innovative institution's ICT infrastructure could solve some of the aforementioned problems [11]. Namely, it would bridge the gap between the process of training and the level of demand for their product. An environment that would bring together the learning resources of educational and industrial projects would be created, and it would cover different levels of training.

According therefore to the high rates of development of both the global ICT market for the education sector, and the IT market of learning tools, the problem of training professional staff for the domestic public administration and the IT-oriented sector of education management; personnel which are primarily prepared within higher and post graduate schools (e.g. universities and advanced training schools) being continuous, and oriented for modern approaches to design of educational systems are a key point.

It is unlikely that the current state of training of management and public administration educational personnel could be regarded as fully satisfactory for the needs of innovative development of ICT-based learning, for the required number of qualified professionals with appropriate structure and quality of skills and competencies. Nowadays content-technological process regarding the creation and use of ICT products, and in particular the electronic learning resources, requires fundamental background knowledge in both ICT and pedagogy. The approaches however for training personnel today, do not sufficiently take into account the recent years' innovative changes in the ICT industry, nor the real needs regarding the extent of such training.

These problems should be considered within the context of development of an institution's and a region's innovative environment as well as on national and international level [3], [11]. Thus the developmental need of new models and approaches to personnel training arises, which will account for the modernization of ICT infrastructure and integrate resources of different levels and use.

A mean for provision of users with relevant services of cloud computing technology is considered to be outsourcing; i.e. a service required in a specific system to implement its core functions being offered and sold by another system external to this [3]. Outsourcing plays an important role in enhancing the scientific and technical level of ICT-systems of an educational institution as well as efficiency of its operation and development. It is a market mechanism incorporating the latest advances in the ICT sector to satisfy user demand [2], [3].

The main problem in educational practice is the contradiction between on the one hand the objective need for a continuous improvement of software and hardware power of training computer complexes, and on the other hand, the lack of personnel's ability (in both qualitative and quantitative manners) to maintain, manage and develop their ICT systems appropriately. Hence informatization of educational institution in terms of cloud computing and ICT outsourcing, will offer realistic solutions for both deepening and improvement of educational performance with ICT and use of information resources [2], [3].

The basic principles of such introduction should be: tight relationship of learning with training and methodological support for tutors, focus on a specific educational task; modularity of learning; continuity of learning, sharing experience and formation and participation in professional association activities (including electronic) [2], [3]. In this process electronic distance learning systems should be actively used, based on the principles of open education, with the maximum possible benefit of CC technology and outsourcing [2].

6 What are the Advantages of Cloud Computing Solution?

6.1 It Is a Cost Effective Solution

Being cost-effective is for the user to be able to get (buy) products and services proposed by the virtual supermarket of ICT according to their needs (individual or group, collective, corporate), “they may pay only for what has been bought (e -transport, e-content, e-services, virtual e-tools, a generic and subject software applications, network platforms - full range of cloud services along with services for the design and implementation of ICT systems and their fragments ordered by the users, their warranty and post warranty service, maintain, upgrade and improvement, etc.) and only for the actual time of use of the purchased product” [3, c.23]. This will allow users to avoid regular updating and upgrading of powerful general system software and hardware tools of their own ICT systems, avoiding a potential surplus of ICT products used from time to time; fragmentary, not fully, as well as spare parts, reduction of requirements for information security of their own ICT systems, reduction of the number of their ICT services and requirements for professional competence of their employees and as a result, significantly reduce overall costs to support operation and development of their ICT systems, to increase their social and economic return, their efficiency [2], [3].

6.2 This Is a Flexible Solution of ICT Infrastructure

It is designed for increased flexibility and effective access to learning resources so as to build a unified and mobile infrastructure.

On the basis of CC infrastructure all main aspects of learner interactions may be comprehended on the unite basis. Along to the approach introduced in [1], among them there were interactions between a learner and other learners; a learner and a teacher; a learner and a learning tool; a learner and educational institution; a learner and society. This will lead to environment of learning organization on the unite base, where collaboration between learners and a tutor, free and flexible resource access, learning activity within social inclusion into environment of educational system and society as a whole will be enabled. The ICT support of learning is realized by means of cloud services. It is designed for adaptation to the rapidly changing external/internal environment, changing of task/competence requirements and development of modern pedagogical approaches.

Due to the principles of open education [1], there is a need to create innovative learning environment to form and develop necessary professional skills. Among them there are leader skills, collaborative skills, critical thinking, and the ability to view a problem in a holistic manner. These skills refer for example to the sphere of public administration of education, since the process of innovative development is involved. This may be achieved on the basis of a holistic approach to specialist training when the planning, design and resource management, learning activity and its monitoring, may be represented on a unite basis. It will be achieved through complex development of different competencies: professional, fundamental, personal and technological.

7 A Holistic Model of a Specialist

Holistic approach to education deals with the learning processes to be taken as a unity of all main aspects of personality development, for example such as mental, emotional and volitional. This is in tune with the term meaning “holistic” as completeness, being impossible with disregard of some of its components.

There are innumerous investigations devoted to the problems of holistic learning development in different areas such as learning and teaching interaction, collaboration processes, engagement of both aspects of theory and practice to gain comprehensive view of a subject and others [7], [9]. There are also new trends of research development in concern to modern ICT. For example, holistic view is to approach learning environment structure. Thus, the model of learning environment, developed in [1] revealing main components and types of interactions within the different learning process settings taking it as a whole contributes to this trend.

The notion of holistic learning occurs in relation to personnel training, concerning to different components and interactions within educational organization. It may touch upon certain types of activity, collaboration and resource management processes, engaging thus the entire organization at all levels and developing a performance culture of personnel. There are different ways to approach peculiarities of specialist competencies formation, namely in the aspect of personal or professional features. That concerning to modeling of professional competencies [5], especially in the sphere of educational management. Another aspect is about holistic models to develop leader skills [4], [9], which are more to traits of a personality.

The proposed approach is based on holistic model of a specialist in the sphere of ICT for education presented in Fig.3. It concerns to Domain Competencies which occupy fundamental knowledge of educational management and modern learning technologies and also ICT skills and ability to use e-learning tools. There are also

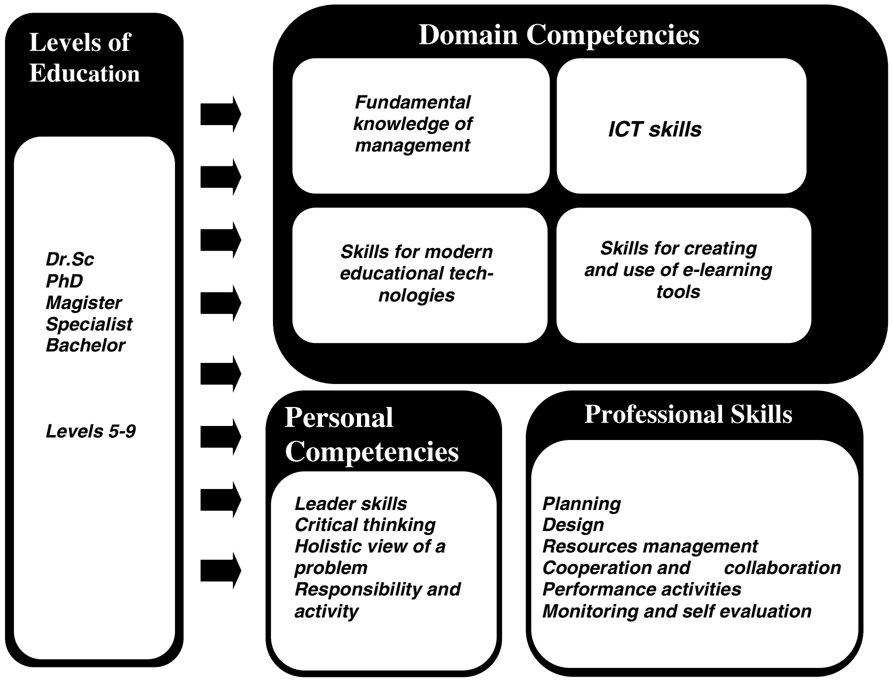

Levels of Education
Dr.Sc
PhD
Magister
Specialist
Bachelor
Levels 5-9
Domain Competencies
Fundamental knowledge of management
ICT skills
Skills for modern educational technologies
Skills for creating and use of e-learning tools
Personal Competencies
Leader skills
Critical thinking
Holistic view of a problem
Responsibility and activity
Professional Skills
Planning
Design
Resources management
Cooperation and collaboration
Performance activities
Monitoring and self evaluation

Fig. 3. A holistic model of a specialist

Personal Competencies, such as leader skills, critical thinking, and capability to holistic view of a problem, responsibility and activity of an individual. As for professional skills there are planning, design, resources management, cooperation and collaboration skills, performance skills and ability for monitoring and self evaluation.

All the components of a specialist's competencies, skills and knowledge are consistently formed within the main levels of education which corresponds to National qualification framework (levels 5-9).

The focus of the proposed model is ICT oriented educational personnel. The main advantage of a holistic approach is a uniform framework which is adapted for the different levels and spheres of learning. At the present case the environment consolidates resources for learners of different levels and also resources for training and management stuff aiming at enhancing ICT skills and competencies needed for operation within the cloud-based learning environment.

8 Analysis and Estimation of Perspective Ways of Development

The important step to wider application and further introduction of new learning approaches should be achieved through modernization and upgrading of ICT learning environment of educational institutions, increasing of overall level of e-learning.

Due to development of cloud computing technologies, functionality and access to collections of electronic learning resources has significantly increased. In this regard, cloud computing is a promising direction of development of electronic resources' collections, as it allows the creation of a unified methodology for a single platform, a framework for development and testing, and for improvement and elaboration of integrated assessment methods' quality. This gives an added value to available recourses [2], [11].

The social results will help to increase educational potential of ICT and add value to the best examples of available learning resources due to their flexible and learner-adaptive access.

The cloud-based learning infrastructure is to give the opportunities:

- To combine the processes of development and use of electronic resources to support learner competencies
- To insure holistic approach to specialist education and training, combining both technological and social competences, development of critical skills of a learner
- To integrate the processes of training, retraining and advanced training, at different levels of education by providing access to electronic resources of a unite learning environment
- To solve or significantly mitigate the problems of association of electronic resources of the institution into unite framework
- To access to the best examples of electronic resources and services to those units or institutions, where there is no strong ICT support services for e-learning
- To provide of invariant access to learning resources within the unified educational environment, depending on the purpose of study or educational level of the student, enabling person-oriented approach to learning

- To make conditions for a higher level of harmonization, standardization and quality of electronic resources, which may lead to emergence of the better examples of learning resources and to more massive use them

The result of instrumentation for cloud-based learning resources collection elaboration, and development of cloud-based learning environment of educational institution might be used within different learning and organizational educational structures.

9 Conclusion

There are real advantages of cloud based learning technologies to assure more flexible and cost-effective access to educational resources as within the university environment and also in environment of the whole region, national and international scale. This is an advantage so as to ensure the joint use of resources and widening participation in the learning process for learners from different institutions while necessary services are substantiated and supported. As if cloud-based learning technologies are already used in education so the challenge is to transfer this experience into the wider context.

The project is implemented within the framework of the Joint research laboratory of Cloud computing in education of the Institute of Information Technologies and Learning Tools of NAPS of Ukraine (Kiev) and the Krivoy Rog State University (Krivoy Rog), www.ccelab.ho.ua.

References

1. Bykov, V.: Models of Organizational Systems of Open Education. Atika, Kyiv (2009) (in Ukrainian)
2. Bykov, V.: Cloud Computing Technologies, ICT Outsourcing, and New Functions of ICT Departments of Educational and Research Institutions. Information Technologies in Education 10, 8–23 (2011) (in Ukrainian)
3. Bykov, V., Shyshkina, M.: Innovative Models of Education and Training of Skilled Personnel for High Tech Industries in Ukraine. Information Technologies in Education 15, 19–29 (2013)
4. Candis Best, K.: Holistic Leadership: a Model for Leader-Member Engagement and Development. The Journal of Values Based Leadership 4(1) (2011)
5. Cheetham, G., Chivers, G.: Towards a Holistic Model of Professional Competence. Journal of European Industrial Training 20(5), 20–30 (1996)
6. Chapman, B., Mosher, F.C., Page, E.C.: Public Administration. Encyclopedia Britannica, `http://www.britannica.com/EBchecked/topic/482290/public-administration`
7. Forbes, S.H., Martin, R.A.: What Holistic Education Claims About Itself: an Analysis of Holistic Schools' Literature. In: Proc. Annual Conf. American Education Research Association, San Diego, California (2004)
8. Zhang, Q., Cheng, L., Boutaba, R.: Cloud Computing: State-of-the-Art and Research Challenges. J. Internet Serv. Appl. 1, 7–18 (2010)

9. Quatro, S.A., Waldman, D.A., Galvin, B.M.: Developing Holistic Leaders: Four Domains for Leadership Development and Practice. Human Resource Management Review 17, 427–441 (2007)
10. Kettl, D., Fessler, J.: The Politics of the Administrative Process. CQ Press, Washington D.C. (2009)
11. Shyshkina, M.: Innovative Technologies for Development of Learning Research Space of Educational Institution. Information Technologies and Society 1 (2013) (in Russian) , http://ifets.ieee.org/russian/depository/v16_i1/pdf/15.pdf
12. Sultan, N.: Cloud Computing for Education: A New Dawn? Int. J. of Information Management 30, 109–116 (2010)

Improving the Efficiency of Synchronized Product with Infinite Transition Systems

Yuliia Romenska[1,2] and Frédéric Mallet[1]

[1] Univ. Nice Sophia-Antipolis, CNRS, I3S, UMR 7271, INRIA,
06900 Sophia Antipolis, France
[2] V.N.Karazin Kharkiv National University, 61022 Kharkiv, Ukraine
`Frederic.Mallet@unice.fr`

Abstract. Embedded System Design is becoming a field of choice for Model-Driven Engineering techniques. On the engineering side, models bring an abstraction of the code that can then be generated (and regenerated) at will. On the semantic side, they bring a reasoning framework to guarantee or verify properties on the generated code. We focus here on the Clock Constraint Specification Language, initially defined as a companion language of the UML Profile for MARTE. More specifically, we propose a state-based data-structure inspired by lazy evaluation technique to represent the unbounded CCSL operators. Lazy evaluation allows for an *intentional* representation of infinite transition systems. We provide an algorithm to compute the synchronized product of such transition systems. Even though the transition systems are infinite, the result of the composition may become finite, in which case the (semi)algorithm terminates and exhaustive analysis becomes possible. We also study the time complexity and show that it is exponential in the number of clocks. We then explore several solutions where this worst-case estimation is not attained and where analyses have a much better complexity in practice.

Keywords: Multiform logical time, synchronized product, lazy evaluation, MARTE CCSL.

1 Introduction - Context and Goal of the Project

In embedded systems the application and the execution platform are usually developed and refined concurrently but not independently. The application brings a set of functional and sometimes non-functional constraints. When mapped (or allocated) onto the execution platform, new non-functional constraints are introduced and must be satisfied. The representation of requirements and constraints in this context becomes an important issue, as they guide the search for optimal solutions inside the range of possible allocations. One of the important aspects of embedded system modeling is to capture the functional and non-functional requirements as well as the constraints (functional and non-functional) imposed by the execution platform through the allocation.

Model-driven engineering brings the technology to capture all these aspects. The Unified Modeling Language (UML) offers a set of constructs to capture both the structural and the behavioral aspects. Its extension, the UML profile for Modeling and Analysis of Real-Time and Embedded systems (MARTE) [1], offers a set of constructs to model resources and capture the functional and non-functional constraints. MARTE comes with a companion language, called the Clock Constraint Specification Language (CCSL) [2,3], dedicated to the specification of causal (functional) and temporal (non-functional) constraints and based on multiform logical time.

Multiform logical time is a flexible notion of time suitable for both functional and extra-functional properties that supports an iterative refinement process. Logical time considers time bases that can be generated from sequences of events not necessarily regular in physical time (as the usual meaning suggest). CCSL has arisen from different models in an attempt to abstract away the data and the algorithm and to focus on events and control. Even though CCSL was initially defined as the time model of the UML profile for MARTE, it has now become a full fledged domain-specific modeling language for capturing chronological, causal and timed relationships and is now developed independently. It combines constructs from the general net theory and from the synchronous languages [4]. It is based on the notion of clocks which is a general name to denote a (totally ordered) sequence of event occurrences. It defines a set of clock relations and expressions. Some CCSL operators are bounded, others are unbounded. The bounded operators are those that can be represented with finite Boolean transition systems. Unbounded operators require a specific symbolic representation.

TimeSquare [5] is an environment dedicated to the analysis of CCSL specifications. Its clock calculus engine performs a step by step exhaustive analysis up to a given finite number of steps. This *simulation* can arrange itself with a finite approximation. Other attempts to perform exhaustive analyses of CCSL [6, 7] were only considering bounded CCSL operators. In [8] there was an attempt to use finite state machines extended with integer variables to represent symbolically the unbounded operators. This paper proposes a new data structure based on lazy evaluation to represent symbolically the unbounded CCSL operators. It proposes an algorithm to perform the synchronized product of unbounded operators. When the composition is finite, the algorithms terminates. When the composition has an infinite number of reachable states, then the (semi)algorithm does not terminate. When the composition is finite, the complexity of the algorithm is analyzed and is shown to be exponential in the number of clocks in the worst case. Solutions are explored to achieve better results in some frequently encountered cases.

The paper is structured as follows. Section 2 summarizes the contribution. Section 3 describes related works and sources of inspirations for our own work. Section 4 introduces the Clock Constraint Specification Language. Section 5 gives the algorithms to perform the synchronized product with our lazy data structure, its theoretical time complexity is briefly studied and implementation details are mentioned. Section 6 discusses some improvements of the algorithm.

2 Contribution

The main contribution is to propose an encoding based on lazy evaluation to represent CCSL unbounded operators. The second contribution is to propose an algorithm to build the synchronized product of such automata. The (semi)algorithm terminates when the composition of unbounded automata becomes bounded.

In this work, the main operators of the clock constraint language were considered. For each basic expression and each relation of the kernel language, a transition system is proposed. Each transition is labeled by the set of clocks that must tick for the transition to be taken. Clocks can stall, tick or be dead. When a clock is dead, it cannot ever tick anymore. A path in the automaton is then an infinite word on the alphabet of powersets of clock names.

Automata representing the unbounded CCSL operators consist of an infinite number of states and therefore transitions (even though each state has a finite number of outgoing transitions). For those operators, a data structure inspired by lazy evaluation technique [9] is used. It allows postponing the construction of the state of an unbounded automaton to the moment when it is actually needed. In very frequent cases, the specification becomes bounded and the intentional infinite representation is never actually expanded.

On these transition systems, we apply the classical synchronized product of transition systems [10]. In the worst case (when automata are independent) the composition is very costly, exponential in the number of clocks. In some (frequent) cases, the cost of composition is much better. We have identified some precise cases where the composition is tractable. This paper extends our previous contribution [11] by identifying a set of frequently encountered cases and detailing some performance results.

3 Related Work and Inspirations

3.1 Timed Automata

The formalism of timed automata [12] has been designed to allow the specification and verification of real-time systems. It extends the concept of finite ω-automata by establishing time constraints. One of the main notions of the timed automata theory is a clock (not to be confused with the notion of clocks introduced in CCSL). In a timed transition the selection of the next state depends on an input symbol and the time reading of the symbol. For this purpose each transition is associated with the set of real-valued clocks. A clock can be set to zero simultaneously with the execution of one of the transition. At any instant the values of such clocks are equal to the time elapsed since the last time they were reset. The transition can be executed *only if* the value of the clock satisfies the constraint associated with this transition. Therefore in the theory of the timed automata a clock is an entity intended for determination of time which elapsed since the last execution of the transition and the setting the value of the clock to zero.

To compare Timed Automata to CCSL, the definition of a clock in CCSL time model must be considered. In CCSL, a clock is a set of ordered instants. Each clock has a lifetime limited by birth and death instants. Formally, a *Clock* c is a tuple $\langle \mathcal{I}_c, \prec_c, c^{\uparrow}, c^{\downarrow}, \equiv_{c\downarrow} \rangle$ where $\mathcal{I}_c$ is a sequence of instants (it can be infinite), $c^{\uparrow}, c^{\downarrow}$ are birth and death instants respectively such that $\mathcal{I}_c \cap \{c^{\uparrow}, c^{\downarrow}\} = \oslash$, $\equiv_{c\downarrow}$ is a coincidence relation and $\prec_c$ is a quasi-order relation on $\mathcal{I}_c \cup \{c\uparrow, c\downarrow\}$. All instants of clocks are strictly ordered, the birth instant precedes all the other instants of the clock and every instant precedes the death. If the set of instants $\mathcal{I}_c$ is infinite then there is no death instant. $\mathcal{I}_c$ represents the occurrences or ticks of the clock c.

Thus we can see that the notions of clock in timed automata and CCSL are radically different. Timed Automata clocks captured physical real-valued time properties, whose value is within a dense time interval. All time clocks evolve at the same rate (without drift). In CCSL, clocks represent logical time properties. The unbounded nature precisely comes from the relative (unbounded) drifts between the clocks that evolve at their own independent rhythm. CCSL also supports the use of physical clocks but offers no specific support for them and treats all the clocks as logical entities. [13] provides a framework where clocks from timed automata and CCSL clocks can be combined to benefit from the two constructs in a single analysis environment.

Lazy data structures were already proposed to capture dense real-time [14] and more recently Timed Automata [15]. We only consider here a discrete abstraction of time.

3.2 Synchronous Data Flow, Marked Graphs

Synchronous Data Flow (SDF) [16,17] is a special case of the dataflow model of computation. Its main characteristic is that the flow of control is completely predictable at compile time. The main components of SDF are the actors, tokens, and arcs. Production and consumption of tokens by actors allow modeling of relative rates of events. In synchronous dataflow graphs, numbers of consumed and produced tokens are constant throughout the execution. To avoid the overflow of resources and to maintain a balanced system, the scheduler must fire the source and destination components at different rates. Such systems can then be used to capture the relative drifts between CCSL clocks.

Safety analysis on SDF graphs is a way to determine whether the system remains bounded or not. Such techniques could be used to study boundness issues for CCSL specifications [18]. However, this is not the concern of this paper. We assume that the composition is bounded and propose an algorithm to build the synchronized product.

3.3 Synchronized Product of Transition Systems

When CCSL operators are expressed as transition systems, their parallel composition simply is the synchronized product of the transition systems [10,19,20].

Synchronization vectors are used to decide which transition systems must synchronize on which transitions. Synchronization vectors allows the specification of purely asynchronous compositions (where only one single system is fired at each step) to purely synchronous compositions (where all the automata must fire one transition at each step), and all the intermediate synchronization schemes. The main difference here is that the number of states may be infinite and we use lazy evaluation to dynamically expand the states whenever they are required to build a new composite state. The composition algorithm terminates only when the synchronized product becomes finite. In [8], there was an initial attempt to build the synchronized product of unbounded CCSL operators. In that work, the automata were folded using *extended automata* (with unbounded integer variables) rather than lazy evaluation. Therefore, the algorithm to compute the synchronized product was always guaranteed to terminate. However, deciding whether the result was finite or not would then require using integer linear programming techniques.

4 The Clock Constraint Specification Language

This section briefly introduces the logical time model of the *Clock Constraint Specification Language* (CCSL). A technical report [2] describes the syntax and the semantics of a kernel set of CCSL constraints.

A clock c is a totally ordered set of instants, I_c. In the following, i and j are instants. A time structure is a set of clocks C and a set of relations on instants $I = \bigcup_{c \in C} I_c$. CCSL considers two kinds of relations: causal and temporal ones. The basic causal relation is causality/dependency, a binary relation on $I : \preccurlyeq \subset I \times I$. $i \preccurlyeq j$ means i causes j or j depends on i. $\preccurlyeq$ is a pre-order on I, i.e., it is reflexive and transitive. The basic temporal relations are precedence ($\prec$), coincidence ($\equiv$), and exclusion ($\#$), three binary relations on I. For any pair of instants $(i, j) \in I \times I$ in a time structure, $i \prec j$ means that the only acceptable execution traces are those where i occurs strictly before j (i precedes j). $\prec$ is transitive and asymmetric (reflexive and asymmetric). $i \equiv j$ imposes instants i and j to be coincident, i.e., they must occur at the same execution step, both of them or none of them. $\equiv$ is an equivalence relation, i.e., it is reflexive, symmetric and transitive. $i \# j$ forbids the coincidence of the two instants, i.e., they cannot occur at the same execution step. $\#$ is irreflexive and symmetric. A consistency rule is enforced between causal and temporal relations. $i \preccurlyeq j$ can be refined either as $i \pi j$ or $i \equiv j$, but j can never precede i. We consider here discrete sets of instants only, so that the instants of a clock can be indexed by natural numbers. For a clock $c \in C$, and for any $k \in N_{>0}$, $c[k]$ denotes the k^{th} instant of c.

Specifying a full time structure using only instant relations is not realistic since clocks are usually infinite sets of instants. Thus, an enumerative specification of instant relations is forbidden. The Clock Constraint Specification Language

(CCSL) defines a set of time patterns between clocks that apply to infinitely many instant relations.

4.1 The Kernel Relations

Table 1 gives a full list of the basic clock relations provided in the CCSL kernel. Each of them can be described as a transition system. We only show the ones that are the most important for the discussion.. It is supposed that each automaton can fire only if one of the participative clocks in the CCSL operator ticks.

Table 1. Basic relations defined in the CCSL kernel (a and b are clocks, not instants)

Ref	Name	Kind of relation	Notation
R1	Subclocking	Synchronous	$a \boxed{\subset} b$
R2	Coincidence	Synchronous	$a \boxed{=} b$
R3	Cause	Asynchronous, unbounded	$a \boxed{\preccurlyeq} b$
R4	Precedence	Asynchronous, unbounded	$a \boxed{\prec} b$
R5	Exclusion	Asynchronous	$a \boxed{\#} b$

Coincidence. According to the considered relation the clocks a and b always tick simultaneously ($a \boxed{=} b$), it is defined as $(\forall k \in \mathbb{N}^\star)(a[k] \equiv b[k])$.

Subclocking. $a \boxed{\subset} b$ defines a as being a subclock of its superclock b (Fig. 1). Every instant of the subclock occurs synchronously with one of the instants of the superclock: $(\forall k \in \mathbb{N}^\star)(\exists i \in \mathbb{N}^\star)(a[k] \equiv b[i])$.

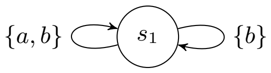


Fig. 1. The automaton for subclocking relation: $a \boxed{\subset} b$

Exclusion. $a \boxed{\#} b$. The clocks connected with this relation cannot have coincidence instants: $(\forall k \in \mathbb{N}^\star)(\forall i \in \mathbb{N}^\star)(a[k] \# b[i])$.

Cause. $a \boxed{\preccurlyeq} b$ is the index-dependent relation. This operator is unbounded (Fig. 2). Every instant of clock a has to precede the instant of clock b with the same index $(\forall k \in \mathbb{N}^\star)(a[k] \preccurlyeq b[k])$.

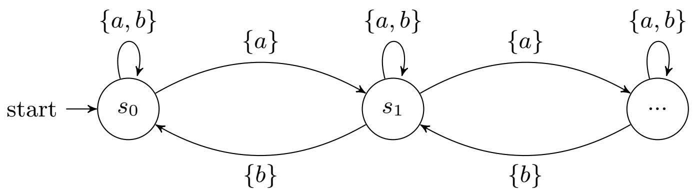


Fig. 2. The automaton for causality relation: $a \boxed{\preccurlyeq} b$

Precedence. $a \boxed{\prec} b$. This relation is a severer version of the previous one in the sense that the instants of the clocks a, b with the same indices cannot be equal: $(\forall k \in \mathbb{N}^{\star})(a[k] \prec b[k])$.

4.2 The Kernel Expressions

A CCSL specification consists of clock declarations and conjunctions of clock relations between clock expressions. A clock expression defines a set of new clocks from existing ones. Most expressions deterministically define one single clock. Table 2 gives a list of the CCSL kernel expressions.

Table 2. Basic expressions defined in the CCSL kernel

Ref	Name	Kind of expression	Notation	Textual form
E1	Inf	Mixed, unbounded	$a \wedge b$	*a glb b*
E2	Sup	Mixed, unbounded	$a \vee b$	*a lub b*
E3	Defer	Mixed	$a \ (\ ns\ ) \rightsquigarrow b$	*a deferred b for ns*
E4	Sampling	Mixed	$a \mapsto b$	*a sampling b*
E5	Strict sampling	Mixed	$a \rightarrow b$	*a strictlySampled b*
E6	Intersection	Synchronous	$a * b$	*a clockInter b*
E7	Union	Synchronous	$a + b$	*a clockUnion b*
E8	Concatenation	Synchronous	$a \bullet b$	*a followedBy b*
E9	Waiting	Synchronous	$a \curlywedge_n b$	*a wait n b*
E10	Preemption (UpTo)	Synchronous	$a \lightning b$	*a upto b*

Sampling. $a \mapsto b$. The sampling expression ticks in coincidence with the tick of the base clock immediately following a tick of the trigger clock and after it dies. In the considered case, the trigger clock is b and the base clock is a. The textual syntax of this expression is represented as $c = a$ *sampling* b. In Figure 3 the automaton is given, where input symbol c is equal to the result clock of the expression. The notation $\{a, b, c\}$ denotes that the automaton remains in state s_2 if a, b and c tick all together simultaneously. If b and c tick simultaneously without a then, the automaton goes back to state s_1. If a ticks alone, it stays in s_2. All other cases are forbidden by the semantics of the operator.

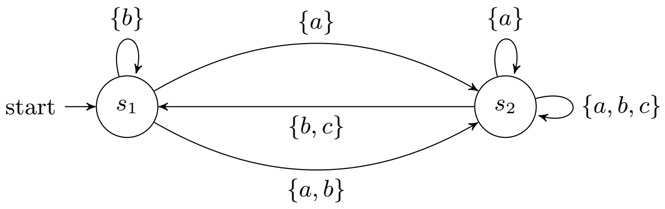


Fig. 3. The automaton for sampling expression: $c = a \mapsto b$

Strict Sampling. $a \rightarrow b$. The expression is a strict version of the previous one where c is emitted when the automaton is in state s_1, and a and b tick simultaneously.

Waiting. $a \curlywedge_n b$. The resulting clock ticks only once after a special number given as a parameter of the base clock, and then the resulting clock dies. $c = a$ *wait* n b, where n is a given parameter (it is a natural number).

Preemption (UpTo). $a \lightning b$. The resulting clock ticks in coincidence with a, it dies as soon as b starts to tick: $c = a$ *upto* b.

Union. This expression is non-terminating and index-independent. Its result is a clock with set of instants which is a union of the instants sets of the clocks-parameters that participate in the expression: $c = a + b$.

Intersection. The result of this index-independent expression is the clock which ticks each time when the clocks-parameters tick simultaneously: $c = a * b$.

Concatenation. $a \bullet b$. The expression is terminating. The resulting clock ticks in coincidence with the first clock-parameter a. After death of a it starts to tick in coincidence with the second clock-parameter b (Fig. 4). The death of clock a means that a can never tick again. The set of instants I_a includes the death instant $a \downarrow$ such that $(\forall i \in I_a)(i \preccurlyeq a \downarrow)$.

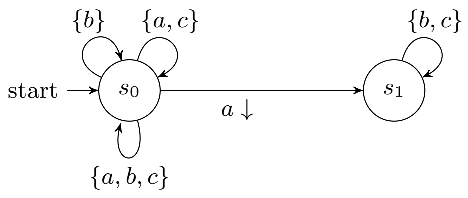


Fig. 4. The automaton for concatenation expression

Defer (Delay). $a\ (\ ns\ )\rightsquigarrow b$. The parameter of the expression are a (the base clock), b (the delay clock) and ns that represents a sequence of elements from $\mathbb{N}_{>0}$. The sequence of the natural numbers can have an infinite part. Let $ns[i]$ be the i^{th} element of the sequence. We assume that if $i \geqslant 0$ then ns has an infinite part, if $0 \leqslant i < p$ there are p elements in the sequence. Every tick of the base clock starts up the counter for respective element of the sequence. For every tick of the delay clock the relative counter is decreased. When the counter reaches 1 the respective instant of clock b occurs. The textual form of the expression is $c = a\ deferred\ b\ for\ ns$.

Sup. $a \vee b$. The expression is index-dependent. The expression $a \vee b$ defines a clock that is slower than both a and b and whose k^{th} tick is coincident with the later of the k^{th} ticks of a and b. The formal definition is presented as:

$$a \boxed{\preccurlyeq} a \vee b$$
$$b \boxed{\preccurlyeq} a \vee b$$
$$\forall c \in C,\ a \boxed{\preccurlyeq} c \implies a \vee b \boxed{\preccurlyeq} c$$
$$\forall c \in C,\ b \boxed{\preccurlyeq} c \implies a \vee b \boxed{\preccurlyeq} c$$

This is a typical example of unbounded transition system (with an infinite number of states).

Inf. $a \wedge b$. This is index-dependent and unbounded. It is the dual of the previous one. The result of the expression is the slowest of faster clocks. It means that the defined clock is faster than both a and b , the k^{th} tick of this clock occurs in coincidence with the earlier of the k^{th} ticks of both a and b. The definition is as follows:

$$a \wedge b \boxed{\preccurlyeq} a$$
$$a \wedge b \boxed{\preccurlyeq} b$$
$$\forall c \in C,\ c \boxed{\preccurlyeq} a \implies c \boxed{\preccurlyeq} a \wedge b$$
$$\forall c \in C,\ c \boxed{\preccurlyeq} b \implies c \boxed{\preccurlyeq} a \wedge b$$

This operator is also unbounded.

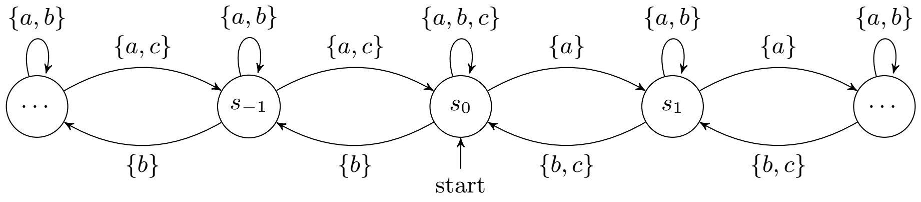


Fig. 5. The automaton for sup expression : $c = a \vee b$

4.3 Unbounded CCSL Operators

Lazy evaluation or call-by-need [9] is an evaluation strategy that delays the evaluation of an expression until its value is actually needed. This approach allows the construction of potentially infinite data structures and avoids repeated evaluations. To construct the algorithm of the parallel execution of several automata, it is necessary to have the possibility to work with infinite data structures (transition systems with an infinite number of states). Lazy evaluation provides the apparatus for this task. The CCSL has four basic unbounded operators which can be represented as infinite automata: the causality relation, the precedence relation, the inf expression (the fastest of slower clocks) and the sup expression (the slowest of faster clocks).

Example 1. Let us consider as an example the automaton for precedence relation (Fig. 6).

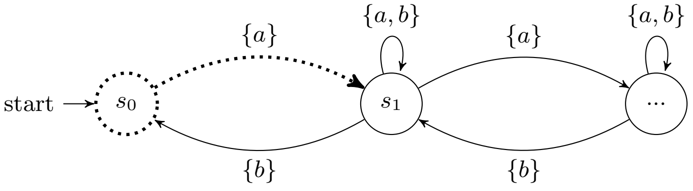


Fig. 6. The automaton for precedence relation

If we only consider the possible outgoing transitions of each state, we can distinguish the initial state (s_0 on Fig. 6) from the other ones. In the initial state, only a can tick and must tick alone. In the other states, if a ticks alone, we must progress to the right (increasing an unbounded counter), if b ticks alone, we must go back to the left (decreasing the counter). If a and b tick simultaneously, we remain in the same state. We assume that each state has a default transition to itself, when nothing happens.

5 The Synchronized Product: An Algorithm

The input for the algorithm is the finite set of initial transition systems $A = \{A_i\}$ and the set of clocks involved in the synchronized product building. The output is the composition of $\{A_i\}$.

The easiest way to understand the principle of the algorithm work is to consider an example. Suppose that we want to build composition of three CCSL operators $a \boxed{\prec} b$, $b \boxed{\prec} c$ and $c = a \$ 1$ (the last operator is a special kind of *defer* expression, according to it the resulting clock starts to tick simultaneously with the base clock after the predefined number of base clock instants). The correspondent automata are shown in Figure 7. State machines A_1 and A_2 for *precedence* relations are unbounded. In our case the set of input automata for the algorithm is $A = \{A_1, A_2, A_3\}$.

The set of participative clocks is denoted as C. It is assumed that C is finite and ordered. Each clock from this set has its own unique number, which can be determined with $\psi : C \to \mathbb{N}$ function. In the proposed example $C = \{a, b, c\}$, function ψ can be defined as follows: $\psi(a) = 1$, $\psi(b) = 2$, $\psi(c) = 3$. $StateDomain = \{stall, tick, dead\}$ is the set of clock possible states, the following abbreviation is used: $stall \leftrightarrow S$, $tick \leftrightarrow T$, $dead \leftrightarrow D$.

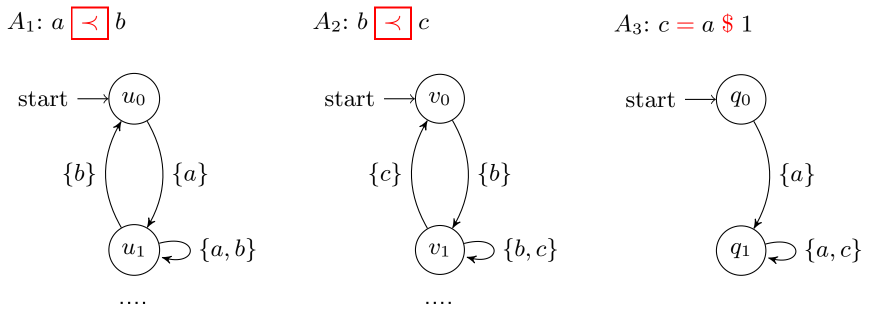


Fig. 7. The set of input automata for composition of operators $a \boxed{\prec} b$, $b \boxed{\prec} c$, $c = a \$ 1$ (A_1 and A_2 are unbounded)

For the algorithm description, the following formal definition of input automata is used. It is based on labeled transition systems introduced by Arnold [10].
$A_i = \langle C_i, S_i, T_i, \beta_i, \lambda_i, \tau_i, s_i^{init} \rangle$, where
$C_i \subset C$: the ordered set of clocks involved in the automaton execution.
S_i: the set of states, $s_i^{init} \in S_i$: the initial state of the automaton.
T_i: the set of transitions.
$\beta_i : T_i \to S_i$: the mapping, for each transition returns a destination state.
$\lambda_i : T_i \to StateDomain^{|C_i|}$ is the action function.
The function λ_i associates with each transition the set of clock states for clocks of the correspondent CCSL operators.
$\tau_i : S_i \to T_i^P$ returns the set of outgoing transitions for a state.

Example 2. For the input automaton A_1: $C_1 = \{a, b\}$, $S_1 = \{u_0, u_1, ...\}$.
For now, we have been using the reduced automata notation where each denoted transition was labeled with clocks ticking during the transition execution. For composition building the explicit automata notation is used. In Figure 8, the outgoing transitions of the initial state u_0 for the automaton A_1 are shown. According to the reduced notation u_0 has only one outgoing transition which corresponds to clock a ticks. But during the execution of this transition clock b can be either *dead* or *stall*. So, in fact instead of one transition there are two outgoing transitions t^0_{TS} and t^0_{TD} in A_1. Besides that, if none of the clocks tick, A_1 remains in the initial state. It is possible if both a and b are either *stall* or *dead* or in any combination of these states. Consequently, there is in total four different outgoing self transitions that do not change the state of the automaton.

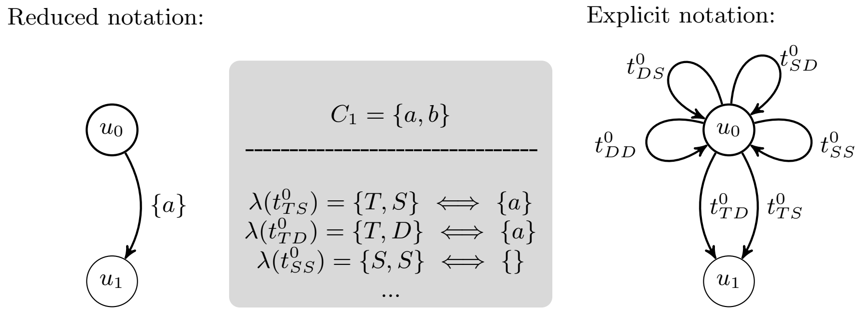


Fig. 8. The initial state outgoing transitions for automaton A_1

Thus, $\tau_1(u_0) = \{t^0_{SS}, t^0_{SD}, t^0_{DS}, t^0_{DD}, t^0_{TS}, t^0_{TD}\}$ where $\lambda_1(t^0_{SS}) = \{S, S\}$ ... $\lambda_1(t^0_{TD}) = \{T, D\}$.

Taking into account everything said above $T_1 = \{t^0_{SS}, t^0_{SD}, t^0_{DS}, t^0_{DD}, t^0_{TS}, t^0_{TD}...\}$ and $\beta_1(t^0_{SS}) = u_0$, $\beta_1(t^0_{TS}) = u_1$.

We denote the synchronized product of $\{A_i\}$ through the tuple
$R = \langle C, S, T, \beta, \lambda, \tau, s^{init} \rangle$ where
C: the set of clocks of the composite automaton is equal to the set of global clocks.
S: the set of composite states $s_1 \times ... \times s_{|A|}$, $s^{init} = s^{init}_1 \times ... \times s^{init}_{|A|}$ is the composite initial state.
T: the set of transitions.
$\forall t \in T, \beta(t) = \beta_1(t) \times ... \times \beta_{|A|}(t)$
$\forall t \in T, \lambda(t) = \lambda_1(t) \times ... \times \lambda_{|A|}(t)$.
$\tau : S \to T^P$ returns the set of outgoing transitions for a composite state, the given function is built during the algorithm execution.

For composition building an auxiliary set of unconsidered composite states is used. It is denoted as N. In order to select an element from N a choice function $pickState : N \to N$ is introduced, such that $pickState(n) \in N$ for each $n \in N$.

5.1 The Composition Algorithm

The composition algorithm is introduced as follows:

Algorithm 1. The composition algorithm

```
Require: Initialize a set of clocks $C$ and a set of input automata $A$
 1: $N := \{s^{init}\}$
 2: while $N \neq \emptyset$ do
 3:     $cur := pickState(N)$
 4:     $N := N \setminus cur$
 5:     $computeReachableCompositeStates(cur)$
 6: end while
```

At the beginning of the algorithm execution, N contains the initial composite state s^{init}. At each iteration one state from N is picked and removed from the set of unconsidered states. For the selected composite state the set of reachable states for initial automata is calculated and based on the obtained result new composite states and respective transitions are built and added to R (Algorithm 2).

Reachable states are found for each of the input automata for all possible clock state combinations. The local array *StateDomain opt* [] of the function $computeReachableCompositeStates(cur)$ (Algorithm 2) is dedicated for storing clock states. Each element of the given array $opt[i]$ corresponds to the current state of $\psi^{-1}(i)$ clock (Fig. 9). If at least for one input transition system the reachable state correspondent to current value of *opt* array is not found the next state combination is considered.

Algorithm 2. $computeReachableCompositeStates(cur)$ function

```
 1: $StateDomain\ opt$ [] % an array of $|C|$ length
 2: for $\forall opt \in StateDomain^{|C|}$ do
 3:     if $determineReachability(opt, cur) = true$ then
 4:         $T := T \cup t$ such that $\lambda(t) = opt$
 5:         for $\forall i = 1..|A|$ do
 6:             $s[i] := \beta_i(\lambda_i^{-1}(opt[\psi(c_1)] \times ... \times opt[\psi(c_{|C_i|})]))$
 7:         end for
 8:         $S := S \cup s$
 9:         if $s \notin N$ & $s \notin S$ then
10:             $N := N \cup s$
11:         end if
12:     end if
13: end for
```

To find the reachable state for the input automaton A_i, all outgoing transitions from its currently considered state $cur[i]$ are examined. If the value of appropriate action function on one of these transitions matches one of the states stored in

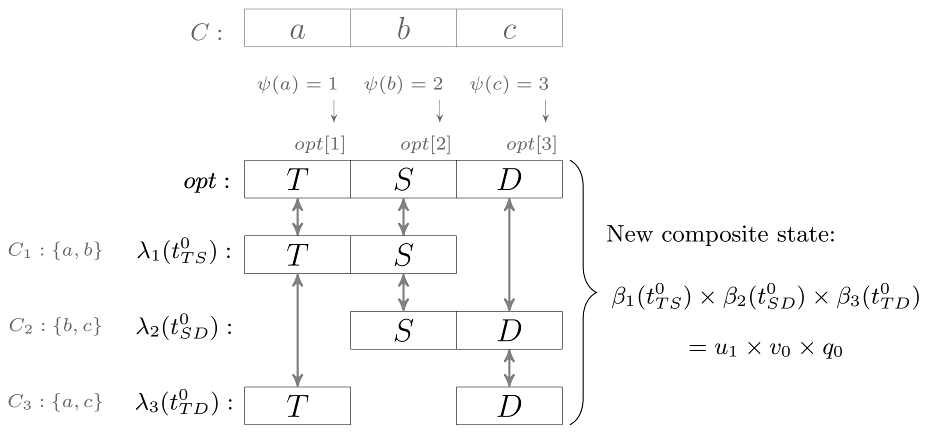


Fig. 9. The principle of reachable states determination

opt array, the reachable state is found and is identified with β_i function on this transition. The new composite state is the cartesian product of all the reachable states for all input automata (Fig. 9).

If the new composite state does not belong neither to N nor to the set of composite states S, N is expanded with this state. The algorithm terminates when the set of unconsidered states is empty.

The following function is used for determination if the reachable states for input transition systems exist. It has two input parameters: *opt* is a current considered combination of clock states, *cur* is a currently considered composite state. For the state $cur[i]$ of the input automaton A_i the outgoing transitions are taken and the value of action function of this transition is compared with appropriate elements of *opt* array. If it is equal the correspondent element, $marks[]$ array is set to *true*.

Algorithm 3. $determineReachability(opt, cur)$ function

```
1: $Boolean\ marks\ [] := false$ % an array of $|A|$ length
2: for $\forall i \in 1..|A|$ do
3:    for $\forall t_j \in \tau_i(cur[i])$ do
4:       if $\lambda(t_j)[k] = opt[\psi(c_k)]\ \forall k = 1..|C_i|$ then
5:          $marks[i] := true$
6:       end if
7:    end for
8: end for
9: if $marks[i] = true\ \forall i = 1..|A|$ then
10:    return $true$
11: end if
12: return $false$
```

For the proposed example, the synchronized product R of automata A_1, A_2 and A_3 is shown in Figure 10 (the reduced notation is used for compact representation).

$$\begin{tikzcd}
\text{start} \arrow[r] & u_0 \times v_0 \times q_0 \arrow[r, bend left, "\{a\}"] & u_1 \times v_0 \times q_1 \arrow[r, bend left, "\{b\}"] & u_0 \times v_1 \times q_1 \arrow[l, bend left, "\{a, c\}"]
\end{tikzcd}$$

Fig. 10. The resulting automaton R for composition of operators $a \boxed{\prec} b$, $b \boxed{\prec} c$, $c = a \$ 1$

5.2 Time Complexity of the Algorithm

To calculate the complexity of the composition algorithm, it is necessary to determine the time complexity of each of the used functions.

To determine existence of the reachable states for the input automata which would match the currently considered combination of clock states, function *determineReachability*() is used. It has three nested loops. The outer loop (lines 2-8) considers all input state machines $\{A_i\}$ and is repeated $|A|$ times. The loop in lines 3-7 (Algorithm 3) runs over all outgoing transitions from the current state of the input transition system A_i. Because the worst case is meant, the number of outgoing transitions can be appreciated as $3^{|C_i|}$. That is, every state can potentially have an exponential number of outgoing transitions T_i^{worst}, such that for all $t \in T_i^{worst}$ $\lambda_i(t)$ are different combinations of clock states for clocks from C_i. The nested loop (line 4) is used for the equality determination of *opt* array elements and action function values. It requires $|C_i|$ repetitions. To check if all initial automata have reachable states the values of *marks*[] array must be examined. The code in line 9 is executed $|A|$ times. Thus, the complexity of function *determineReachability*() is equal to

$$|A| \times 3^{|C_i|} \times |C_i| + |A| = |A|(3^{|C_i|} \times |C_i| + 1) \quad (1)$$

For computing reachable composite states for the resulting automaton R all possible clock state combinations must be processed. Consequently, the number of iterations of the loop in lines 2-13 (Algorithm 2) is equal to $3^{|C|}$. For obtaining the new composite state from reachable ones (lines 5-7) $|A| \times |C_i|$ number of operations must be performed. Taking into account the complexity of *determineReachability*() function, one can express the total complexity of function *computeReachableCompositeStates*() as

$$3^{|C|}(|A|(3^{|C_i|} \times |C_i| + 1) + |A| \times |C_i|) = 3^{|C|} \times |A| \times |C_i|(3^{|C_i|} + \frac{1}{|C_i|} + 1) \quad (2)$$

The composition algorithm works while they are some unconsidered states in N (Algorithm 1, lines 2-6). Therefore, the formula expressing the time complexity of the composition building is as follows:

$$|N| \times 3^{|C|} \times |A| \times |C_i|(3^{|C_i|} + \frac{1}{|C_i|} + 1) \tag{3}$$

From the obtained result (3) we can conclude that the developed composition algorithm has exponential complexity in the number of clocks involved in the composition. Additionally, it depends on the number of states in the resulting automaton. If the composition is infinite, the algorithm does not terminate. When the synchronized product is finite, the complexity is finite in the number of participative clocks in the worst case.

5.3 The Implementation of the Algorithm

For a software implementation of the algorithm, it is necessary to represent each kind of the built automata in terms of classes and to define the suitable presentation of the states. There are two types of input transition systems: finite and infinite. For the latter ones, lazy evaluation is used to compute the next state on demand.

All classes are divided into six main, architecturally significant packages: *entities*, *composition*, *abstractions*, *states*, *automata* and *executors*. The *entities package* contains the primary classes-entities, which are necessary for the implementation of the composition algorithm such as the class for the representation of the clock notion. The class for the resulting automaton and all related classes can be found in the *composition package*. The *abstractions package* stores two main interfaces that specify responsibilities for all automata and states. The *states package* includes all classes intended to work with states of all kinds of automata. The implementation for all previously formally defined transition systems for the basic CCSL operators is contained in the *automata package*. The *executors package* includes class-executors that are responsible for organizing the correct interactions of all classes so as to build the synchronized product of the input automata.

5.4 Algorithm Evaluation Results

Using the implementation mentioned above series of tests were performed. It was shown that practical time complexity corresponded to the theoretical one.

All sets of CCSL specifications were divided into three conventional groups: specifications with independent operators, the set of dependent CCSL operators and mixed ones.

Independent operators do not have common clocks, e.g. $a \boxed{\subset} b$, $c \boxed{\subset} d$. On the contrary, dependent operators are related through involved clocks. In the case of two precedence relations $a \boxed{\prec} b$ and $b \boxed{\prec} c$ clock b is common. The mixed CCSL specifications include both independent and dependent pairs of operators.

For the execution time testing the sets of operators were chosen so that the synchronize product was known to be finite. During the experiments extreme cases were examined, *i.e.,* CCSL specifications with all independent operators, or, specifications where all the operators depend on at least one other operator from the same specification. The given choice was based on the necessity to evaluate execution of the algorithm in critical cases.

One of the considered CCSL specifications with independent operators was the set of independent *subclocking* relations ($a \boxed{\subset} b$, $c \boxed{\subset} d$, $e \boxed{\subset} f$ ...). Because the state space for each of the appropriate input automaton contained only one state the finite composition existed. Moreover, the given example covered all specifications consisting from the set of synchronous CCSL operators such as *intersection* , *union* and *coincidence* as all of them had similar state spaces. As a sample of dependent operators the CCSL specification "precedence chain" was considered. $a \boxed{\prec} b$, $b \boxed{\prec} c$ and $c = a \ \$ \ 1$ is an example of the chain. The last operator $c = a \ \$ \ 1$ guarantees that the composition is finite despite the fact that automata for $a \boxed{\prec} b$, $b \boxed{\prec} c$ are unbounded.

Tests for 6, 8, 10,12 and 14 involved clocks were performed with respectively 3, 4, 5, 6 and 7 input automata for the independent *subclocking* operators set and 6, 8, 10, 12, 14 initial state machines for the precedence chain specification. The obtained evaluation results have shown that the execution time for both introduced examples was as worst as it was predicted, *i.e.,* its grows exponentially with the increasing of the number of participative clocks.

6 Complexity Improvement

The proposed "brute force" algorithm supposes exhaustive consideration of all possible state combinations of all participative clocks at each iteration.

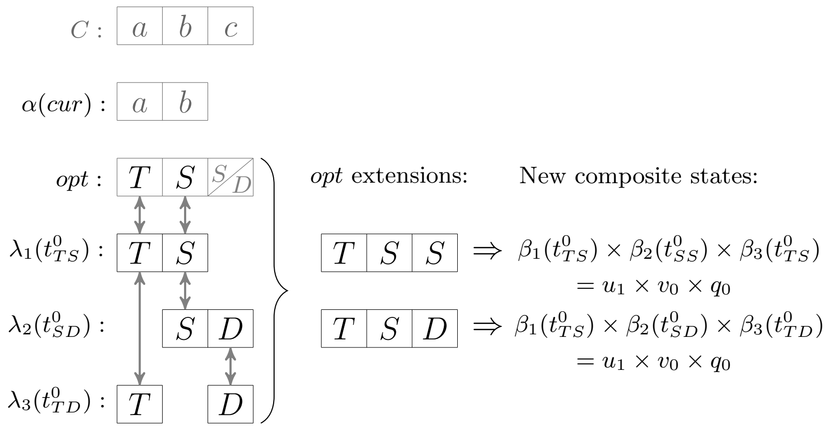
Let us go back to the considered example with composition of automata A_1, A_2 and A_3 for operators that correspond to the precedence chain. The reduced notation of the resulting automaton for their composition is shown in Figure 10. Taking into account the explicit notation of the resulting automaton R, we can conclude that in order to find 13 outgoing transitions for its initial composite state $u_0 \times v_0 \times q_0$ $3^3 = 27$ options must be considered (4 of 13 transitions conform to clock a tick when clocks b and c can be either dead or stall, other nine are transitions "by default"). For the given simplest example the number of uselessly examined combinations ($27 - 13 = 14$) does not seem so critical, but for specifications with larger number of clocks the difference can be huge. To overcome the complexity problem and to adapt the algorithm for reality and practical specifications, a modification of the brute force algorithm was proposed. It is based on the active clock consideration approach.

6.1 Active Clock Modification

To understand the given modification, the notion of active clock must be introduced. Active clock is a clock that can potentially tick during the execution of one of the outgoing transitions from the state. For example, in Figure 7 for the initial state u_0 of automaton A_1 the set of active clocks contains only clock a. The principle of the active clock consideration approach consists in the idea that at each iteration of the algorithm only combinations of active clock states are examined.

The definition of input automata was extended with the function $\alpha_i : S \to 2^{|C_i|}$. It returns the set of active clocks for each specified state. For resulting automaton it is defined as $\alpha(s) = \alpha_1(s[1]) \times ... \times \alpha_{|A|}(s[|A|])$ for all $s \in S$.

The modified version of the algorithm examines only state combinations for active clocks $\widetilde{C} = \alpha(s)$. Thus, the number of repetitions of the loop in lines 3-17 (Algorithm 2) is reduced to $3^{|\widetilde{C}|}$.



C :
a
b
c
$\alpha(cur)$:
opt :
T
S
S/D
$\lambda_1(t^0_{TS})$:
$\lambda_2(t^0_{SD})$:
$\lambda_3(t^0_{TD})$:
opt extensions:
New composite states:
T S S $\Rightarrow \beta_1(t^0_{TS}) \times \beta_2(t^0_{SS}) \times \beta_3(t^0_{TS}) = u_1 \times v_0 \times q_0$
T S D $\Rightarrow \beta_1(t^0_{TS}) \times \beta_2(t^0_{SD}) \times \beta_3(t^0_{TD}) = u_1 \times v_0 \times q_0$

Fig. 11. The principle of reachable states determination for the active clock approach ($cur = u_0 \times v_0 \times q_0$)

To determine if reachable states exist for the given input automata (Algorithm 3) only values of λ_i function for active clocks obtained with α_i mapping are compared with appropriate elements of *opt* array. If respective values are equal, then for each extended clock state combination obtained form *opt* array the new composite states are calculated. Extended clock state combinations are build by the use of sequential substitution of states *stall* and *dead* for each inactive clock (Fig. 11).

Therefore, the theoretical complexity of composition process is still exponential but in the number of active clocks. In the worst case, if the set of independent operators is considered, it degenerates to the brute force algorithm complexity. This can be explained by the fact that at each algorithm iteration all clocks can potentially tick.

Figure 13 compares the composition execution time for the brute force algorithm and the active clock approach for the set of independent *subcloking* operators. It is obvious that for independent CCSL specification the active clock modification does not improve time complexity. On the contrary, the time of

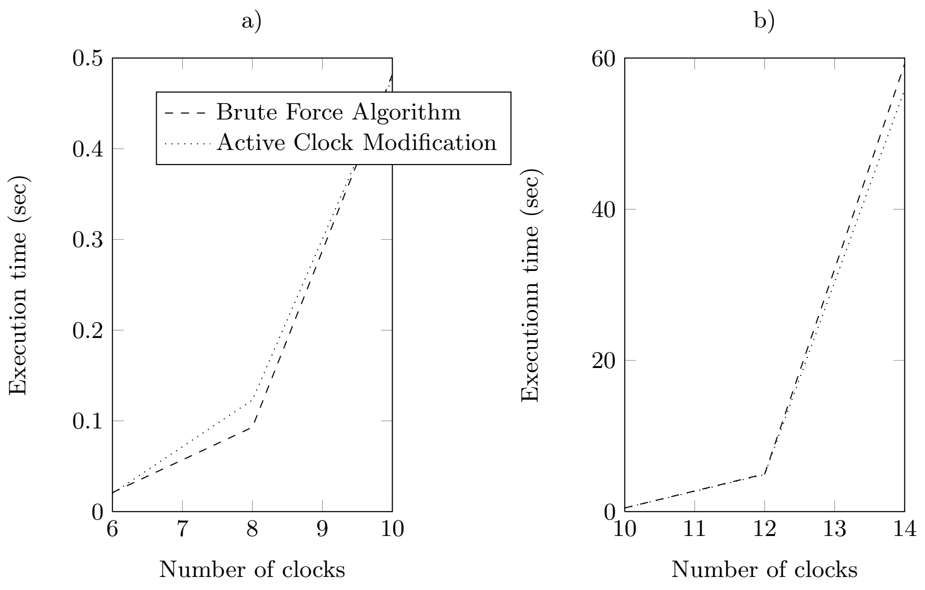


Fig. 12. The brute force algorithm versus the active clock modification. The precedence chain CCSL specification ($a \boxed{\prec} b$, $b \boxed{\prec} c$, $c = a \$ 1$).

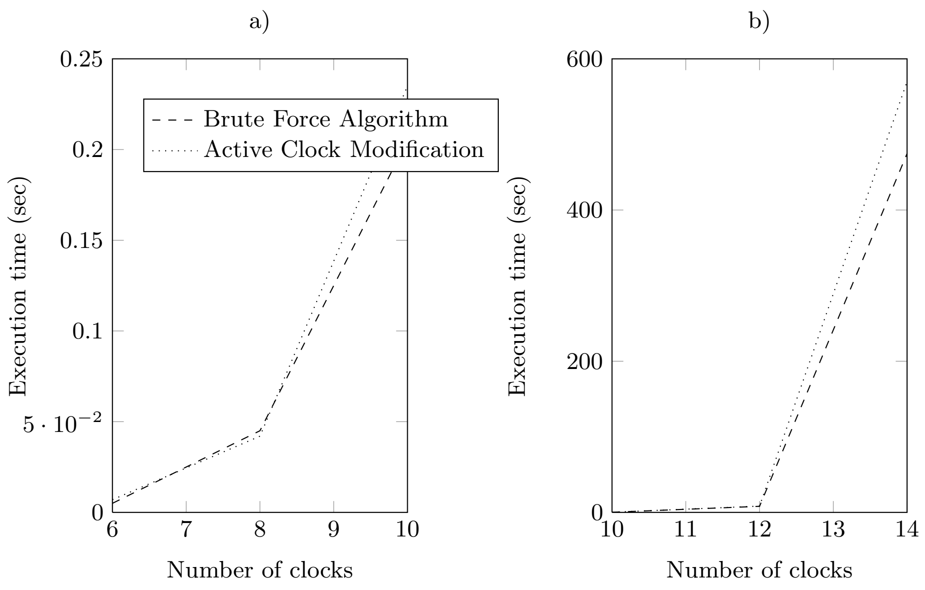


Fig. 13. The brute force algorithm versus the active clock modification. The CCSL specification of independent subclocking operators ($a \boxed{\subset} b$, $c \boxed{\subset} d$).

composition building for the precedence chain as an example of CCSL specification with dependent operators is less than for the case of modification using (Fig. 12).

Despite the fact, that the execution time improvement is not very impressive, the active clock consideration method is fundamental for further time complexity reducing.

6.2 Composition Strategies

It was ascertained, that using the active clock approach in aggregation with different composition building strategies can significantly improve the execution time of the synchronized product. For the range of practical CCSL specifications two strategies were identified: the splitting of the composition process and the clock hiding method.

Composition Splitting. The basic principle of the composition spitting strategy is the following. The composition process can be broken down into several

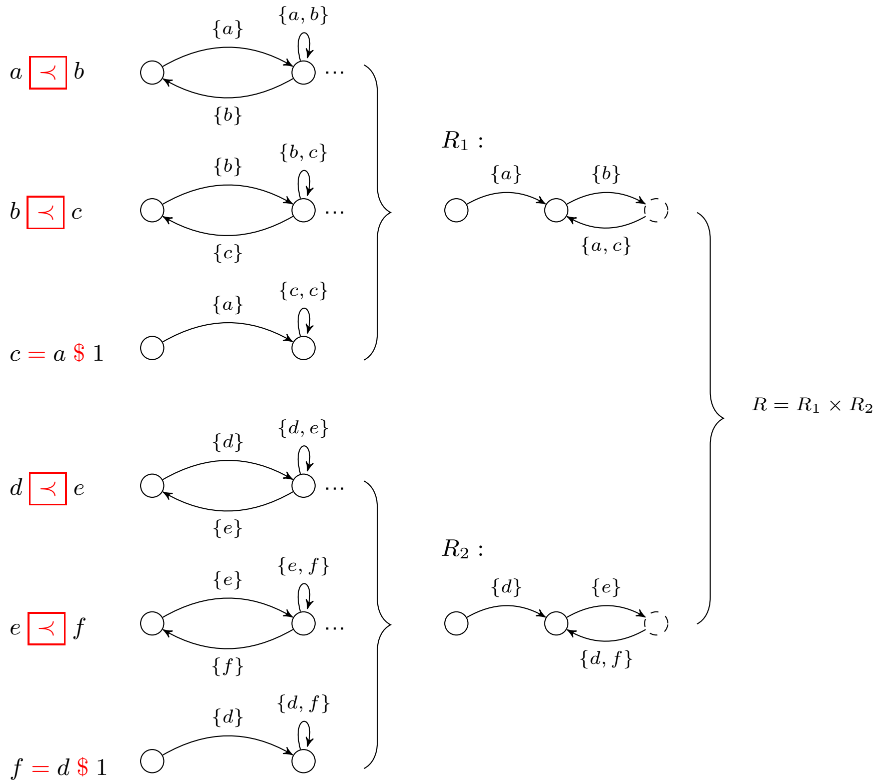


Fig. 14. The composition splitting strategy

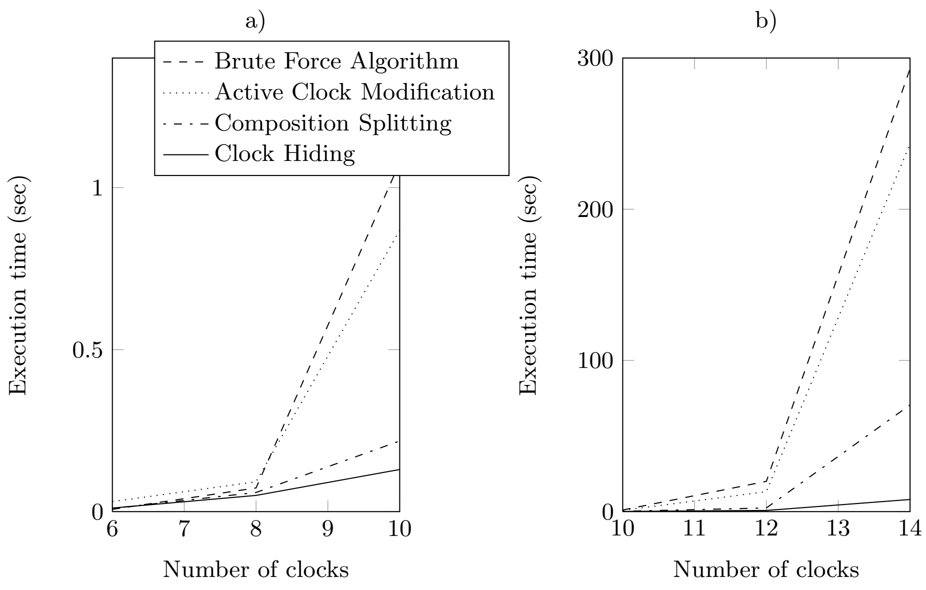


Fig. 15. Comparison of the execution time for the brute force algorithm, the pure active clock modification, the splitting strategy and the hiding clock approach. Two independent precedence chains ($a \boxed{\prec} b$, $b \boxed{\prec} c$, $c = a \$ 1$, $d \boxed{\prec} e$, $e \boxed{\prec} f$, $f = d \$ 1$).

steps. Suppose that it is necessary to build composition of the CCSL specification shown in Figure 14 with 6 operators and 6 different clocks : a, b, c, d, e and f. As it can be noticed, the proposed set of operators contains two precedence chains. It is possible to split up the given set of operators into subsets and build resulting automata for each of them. For now the splitting of the specifications has been performed manually. On the next step the synchronized product of the obtained state machines R_1 and R_2 can be built. Therefore, if the active clock approach is used, in the worst case, if current states are those marked with dotted lines in the figure, only 4 clocks a, c, d, f out of 6 are considered as active. It means that the cost for building the final composition R is decreased.

Clock Hiding Strategy. To understand the clock hiding strategy, it is necessary to introduce the notion of local clock. A *local clock* is a clock, belonging to some CCSL specification and invisible for other part of the specification. For better understanding this notion the alternate relation $a \boxed{\sim} b$ can be considered. The given operator means that ticks of clocks a and b take turn. This relation is defined with three simpler CCSL operators $a \boxed{\prec} b$, $b \boxed{\prec} c$ and $c = a \$ 1$ introduced before. For the definition of alternate relation three clocks a, b and c are used. But the final relation depends on only 2 clocks a and b. Thus, clock c is an auxiliary or local clock for derivation of new relations and it is invisible to the external world. The structure of CCSL is hierarchical, all operators can be used for formalization of new ones, that is why local clocks are essential part of the clock constraint language.

The work of the clock hiding approach is based on the idea that it is allowed not to consider local clocks during the composition building of intermediate state machines. The same set of steps as for composition splitting is repeated but with one improvement: all local clocks are deleted from intermediate automata $R1$ and $R2$ (Fig. 14). For the first subset the local clock is c and for the second precedence chain is f. Thus, after removing local clocks, in the worst case only 2 clocks a and d are considered as active instead of three.

Figure 15 compares the algorithm execution time for the composition building of two precedence chains. The significant improvement of the algorithm time complexity in comparison with composition splitting can be observed, to say nothing about the initial version of the algorithm.

7 Conclusion

We have proposed a data structure based on lazy evaluation to encode the semantics of CCSL operators as finite or infinite transition systems. Then the composition of CCSL operators is computed as the synchronized product of those transition systems. The underlying definitions have been implemented in Java. The sate-based representation for basic expressions and relations of the CCSL kernel have been proposed. The lazy evaluation is considered and applied for building an intentional representation of infinite automata for the corresponding so-called unbounded operators.

The complexity analysis of the algorithm has been conducted. It has allowed estimating the time resources needed by the composition algorithm which solves a given computational problem. It has been shown that the computational complexity of the developed algorithm is exponential in the number of clocks and is linear with the size of the resulting automaton. Such an automaton can be infinite in which case the (semi)algorithm does not terminate. A set of experiments for different CCSL specifications has also been conducted. Based on the obtained results useful cases tractable in CCSL have been identified. Three approaches for complexity improvement have been proposed. They include the *active clock* algorithm modification and two composition building strategies built on top of the active clock approach. A significant improvement of the execution time has been obtained.

References

1. OMG: UML Profile for MARTE, v1.0. Object Management Group, formal/2009-11-02 (November 2009)
2. André, C.: Syntax and Semantics of the Clock Constraint Specification Language (CCSL). Research Report RR-6925, INRIA (2009)
3. Mallet, F.: CCSL: specifying clock constraints with UML/MARTE. Innovations in Systems and Software Engineering 4(3), 309–314 (2008)

4. Benveniste, A., Caspi, P., Edwards, S.A., Halbwachs, N., Le Guernic, P., de Simone, R.: The synchronous languages 12 years later. Proceedings of the IEEE 91(1), 64–83 (2003)
5. Deantoni, J., Mallet, F.: Timesquare: Treat your models with logical time. In: Furia, C.A., Nanz, S. (eds.) TOOLS 2012. LNCS, vol. 7304, pp. 34–41. Springer, Heidelberg (2012)
6. Gascon, R., Mallet, F., DeAntoni, J.: Logical time and temporal logics: Comparing UML MARTE/CCSL and PSL. In: TIME, pp. 141–148 (2011)
7. Yin, L., Mallet, F., Liu, J.: Verification of MARTE/CCSL time requirements in Promela/SPIN. In: ICECCS, pp. 65–74 (2011)
8. Mallet, F.: Automatic generation of observers from MARTE/CCSL. In: IEEE Int. Symp. on Rapid System Prototyping, pp. 86–92. IEEE (October 2012)
9. Johnsson, T.: Efficient compilation of lazy evaluation. In: SIGPLAN Symposium on Compiler Construction, pp. 58–69. ACM (1984)
10. Arnold, A.: Synchronized products of transition systems and their analysis. In: ICATPN, pp. 26–27 (1998)
11. Romenska, Y., Mallet, F.: Lazy parallel synchronous composition of infinite transition systems. In: ICTERI. CEUR Workshop Proceedings, vol. 1000, pp. 130–145. CEUR-WS.org (2013)
12. Alur, R., Dill, D.L.: A theory of timed automata. Theoretical Computer Science 126, 183–235 (1994)
13. Suryadevara, J., Seceleanu, C., Mallet, F., Pettersson, P.: Verifying MARTE/CCSL mode behaviors using UPPAAL. In: Hierons, R.M., Merayo, M.G., Bravetti, M. (eds.) SEFM 2013. LNCS, vol. 8137, pp. 1–15. Springer, Heidelberg (2013)
14. Sorea, M.: Lazy approximation for dense real-time systems. In: Lakhnech, Y., Yovine, S. (eds.) FORMATS/FTRTFT 2004. LNCS, vol. 3253, pp. 363–378. Springer, Heidelberg (2004)
15. Herbreteau, F., Srivathsan, B., Walukiewicz, I.: Lazy abstractions for timed automata. In: Sharygina, N., Veith, H. (eds.) CAV 2013. LNCS, vol. 8044, pp. 990–1005. Springer, Heidelberg (2013)
16. Lee, E.A., Messerschmitt, D.G.: Static scheduling of synchronous data flow programs for digital signal processing. IEEE Trans. Computers 36(1), 24–35 (1987)
17. Lee, E.A., Parks, T.M.: Dataflow process networks. In: De Micheli, G., Ernst, R., Wolf, W. (eds.) Readings in hardware/software co-design, pp. 59–85. Kluwer Academic Publishers, Norwell (2002)
18. Mallet, F., Millo, J.-V.: Boundness issues in CCSL specifications. In: Groves, L., Sun, J. (eds.) ICFEM 2013. LNCS, vol. 8144, pp. 20–35. Springer, Heidelberg (2013)
19. Arnold, A.: Finite transition systems - semantics of communicating systems. Prentice Hall international series in computer science. Prentice Hall (1994)
20. Arnold, A.: Nivat's processes and their synchronization. Theor. Comput. Sci. 281(1-2), 31–36 (2002)
21. DeAntoni, J., André, C., Gascon, R.: CCSL denotational semantics. Research Report RR-8000, INRIA (2010)

On Existence of Total Input-Output Pairs of Abstract Time Systems

Ievgen Ivanov[1,2]

[1] Taras Shevchenko National University of Kyiv, Ukraine
[2] Paul Sabatier University, Toulouse, France
ivanov.eugen@gmail.com

Abstract. We consider a class of abstract mathematical models called blocks which generalize some input-output system models which are frequently used in system theory, cybernetics, control theory, signal processing. A block can be described by a multifunction which maps a collection of input signals (input signal bunch) to a non-empty set of collections of output signals (set of output signal bunches). The input and output signal bunches are defined on a subset of a continuous time domain.

We investigate and provide methods for proving the existence of a pair of corresponding input and output signal bunches of a given block, both components of which are defined on the entire time domain (a total input-output pair), and the existence of a total output signal bunch corresponding to a given total input signal bunch.

Keywords: input-output system, abstract time system, dynamical system, signal transformer, semantics.

1 Introduction

An abstract view of a computing system as a transformation of data, a function, or an input-output relation is rather common in computer science. In fact, this view is rooted in foundations of computing and is notable in the works of A. Turing and A. Church.

Nevertheless, a large amount of computing systems used today act not as pure data transformers, but as agents interacting with physical processes. Such systems are now frequently called cyber-physical systems [1, 2]. Examples include automotive systems, robotics, process control, medical devices, etc. [3].

As was stressed in [4], an important aspect that cyber-physical systems must take into account is the passage of (physical) time. The actions of such systems must be properly timed. Besides, the computational aspect of a system must be understood and modeled in a close relation with physical processes. However, this is not taken into account when a system is viewed as an input-output relation on data.

A simple way to resolve this is to view a system as an input-output relation on time-varying quantities (signals). A view of this kind is extensively used in signal processing and control theory [5, 6]. However, the kinds of mathematical models of systems usually considered in these fields (e.g. systems of linear or non-linear difference or differential equations, transfer function representation of linear systems, etc.) do not provide a high-level abstraction of processes that take place in cyber-physical systems [1].

A high-level treatment of systems as input-output relations (or as relations in general) can be found in mathematical systems theory. During the second half of the XX century a large number of works that dealt with general mathematical theory of systems were published by L. Zadeh [7, 8], R. Kalman [9], M. Arbib [10], G. Klir [11], W. Wymore [12], M. Mesarovic [13], B.P. Zeigler [14], V.M. Matrosov [15], and others [16–18].

Many of these works were inspired and influenced by the General Systems Theory by L. Bertalanffy, Cybernetics introduced by N. Wiener, information theory introduced by C. Shannon, circuit theory in electrical engineering, automata theory, control theory. A historical account on the mutual influence between these fields is given in [19, 11]. In particular, the approach developed by M. Mesarovic [13] is based on formalization of a system as a relation on objects. Other approaches such as those developed by M. Arbib [10], W. Wymore [12], B.P. Zeigler [14] resulted from unification of the theory of systems described by differential equations and automata theory.

With regard to the input-output view, many of the mentioned works introduce some kind of abstract view of a system as an input-output relation on time-varying quantities (e.g. a general time system [13, Section 2.5], external behavior of a dynamical system [9, Section 1.1], oriented abstract object [7, Chapter 1, Paragraph 4], I/O observational frame [14, Section 5.3]) and consider such a relation as a mathematical representation of an observable behavior of a real-world system. The most basic example is the definition of a Mesarovic time system [13] as a binary relation $S \subseteq I \times O$, where I and O are sets of input and output functions on a time domain T ($I \subseteq A^T$, $O \subseteq B^T$).

However, one aspect that is not sufficiently investigated in works on mathematical systems theory with regard to time systems is partiality of input and/or output signals as functions of time. This aspect becomes most important, when the input-output relation describing a real-world system results not from a direct observation of its behavior, but as an abstraction of a lower level mathematical model of this system. The reason is that a lower level model (e.g. a system of differential equations, a hybrid system, etc. [20]) may describe the behavior of a real-world system only on a bounded time interval, after which the model's behavior becomes undefined. This can indicate a real phenomenon (e.g. termination or destruction of the real system), or a failure of the model [21].

For example, a phenomenon of a finite time blow-up is well known in the theory of differential equations and applied mathematics [21, 22]. It is characterized by the unbounded growth of the value of one or several system variables during a bounded time interval. A simple example is a (non-zero) solution $x(t) = 1/(c-t)$, $c = \mathit{const}$ of the equation $x'(t) = x(t)^2$, for which $|x(t)| \to +\infty$, when $t \to c$. A survey of the respective results and applications can be found in [21–23].

Another situation when a mathematical model defines a system's behavior on a bounded time interval is a Zeno behavior [24, 25] which arises in hybrid (discrete-continuous) systems [20]. In this case, a hybrid system performs an infinite sequence of discrete steps during a bounded total time, but each step takes a non-zero time. A simple example in which this behavior arises is a hybrid automaton [20] which models a bouncing ball [24].

It should be noted that the problems of detection of finite time blow-up or Zeno behaviors, their physical interpretation, and if necessary, adjustment of a model to avoid such behaviors are non-trivial, so one cannot assume that any available and useful model of real-world system would be free of them.

This dictates that when using an input-output abstraction of a real system based on an available mathematical model of this system, one must take into account partial input and outputs.

In the previous work [26] we introduced a class of input-output abstractions of real-world systems which we called a class of *blocks*. A block can be thought of as a generalization of a Mesarovic time system which takes into account partiality of inputs and outputs as functions of time.

Basically, a block is a multifunction which maps a collection of input signals (*input signal bunch*) to a (non-empty) set of collections of output signals (a set of *output signal bunches*), and a signal is a partial function on a continuous time domain. The operation of a block can be described by a set of input-output pairs (I/O pairs) (i, o), where i is an input signal bunch and o is a corresponding output signal bunch. The main aspects captured by this notion are nondeterminism (multiple possible output signal bunches corresponding to one input signal bunch), continuous time, partiality of inputs and output signals.

In the work [26] we studied the notions of causality (nonanticipation), refinement, and composition for blocks.

In this work we continue to investigate properties of blocks and consider the following questions.

We introduced the notion of a block to take into account the possibility of partial inputs and outputs, or, in other words, to allow I/O pairs (i, o), where $dom(i)$ and $dom(o)$ may not cover the whole time domain (note that $dom(i) \neq dom(o)$ is also possible). In the work [26] we gave the following interpretation to this partiality: the case $dom(o) \subset dom(i)$ for an I/O pair (i, o) means that a block receives an input signal bunch i, but does not process it completely. It outputs o and terminates abnormally. On the other hand, the case $dom(o) = dom(i)$ means that a block processes the input signal bunch i completely, outputs o, and terminates. In particular, if T is a time domain, (i, o) is an I/O pair, and $dom(i) = T$, then if $dom(o) = T$, the block processes the input completely and normally and outputs o, otherwise, it outputs o and terminates abnormally at some time moment. In the former case, ($dom(i) = dom(o) = T$) we say that (i, o) is a total I/O pair.

One can say that in models such as Mesarovic time system all I/O pairs are total. But in blocks total I/O pairs form only a subset of the set of all I/O pairs.

In this chapter we consider the following question about total I/O pairs:

(A) How can one prove that a given block B has a total I/O pair (if B indeed has a total I/O pair) ?

Using the same techniques which will used to answer this question, in this chapter we will also give an answer to the following question:

(B) How can one prove that for a given input signal bunch i, where $dom(i) = T$, there exists an I/O pair (i, o) with $dom(o) = T$?

That is, a block admits a total output for a given total input.

In this chapter we will consider (A) and (B) for strongly nonanticipative blocks [26] only. As we argued in [26], strongly nonanticipative blocks are sufficient for modeling physically realizable (causal) real-world systems.

The practical significance of the questions (A) and (B) follows from the interpretation of the case $dom(o) \subset dom(i)$ as an abnormal termination of a block on the input i. More specifically, the methods used for solving (B) can interpreted as methods of proving that it is possible for a block to process a given input normally (without errors) and can be rather straightforwardly linked with such domains as viability theory, control synthesis, real-time software verification, etc.

The chapter is organized in the following way. In Section 2 we recall the definition of a block and other related definitions and facts from [26]. In Section 3 we show that each strongly nonanticipative block has a representation in the form of an abstract dynamical system of a special kind called Nondeterministic Complete Markovian System (NCMS) [27]. In Section 4 we use the facts about existence of global-in-time trajectories of NCMS which were shown in [27] and the representation given in Section 3 in order to prove criteria which answer the questions (A) and (B).

2 Preliminaries

2.1 Notation

We will use the following notation: $\mathbb{N} = \{1, 2, 3, ...\}$, $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$, $\mathbb{R}_+$ is the set of nonnegative real numbers, $f : A \to B$ is a total function from A to B, $f : A \tilde{\to} B$ is a partial function from A to B, 2^A is the power set of a set A, $f|_X$ is the restriction of a function f to a set X. If A, B are sets, then B^A denotes the set of all total functions from A to B and ${}^A B$ denotes the set of all partial function from A to B. For a function $f : A \tilde{\to} B$ the symbol $f(x) \downarrow$ ($f(x) \uparrow$) means that $f(x)$ is defined (respectively undefined) on the argument x.

We denote the domain and range of a function as $dom(f) = \{x \,|\, f(x) \downarrow\}$ and $range(f) = \{y \mid \exists x \ f(x) \downarrow \wedge \ y = f(x)\}$ respectively. We will use the the same notation for the domain and range of a binary relation: if $R \subseteq A \times B$, then $dom(R) = \{x \,|\, \exists\, y \ (x, y) \in R\}$ and $range(R) = \{y \,|\, \exists\, x \ (x, y) \in R\}$.

We will use the notation $f(x) \cong g(x)$ for the strong equality (where f and g are partial functions): $f(x) \downarrow$ iff $g(x) \downarrow$ and $f(x) \downarrow$ implies $f(x) = g(x)$.

The symbol $\circ$ denotes a functional composition: $(f \circ g)(x) \cong g(f(x))$.

The notation $X \mapsto y$, where X is a given set and y is a given value, means a constant function defined on X which takes the value y.

By T we denote the (positive real) time scale $[0, +\infty)$. We assume that T is equipped with a topology induced by the standard topology on $\mathbb{R}$.

Additionally, we define the following class of sets:

$$\mathcal{T}_0 = \{\emptyset, T\} \cup \{[0, x) \mid x \in T \backslash \{0\}\} \cup \{[0, x] \mid x \in T\}$$

i.e. the set of (possibly empty, bounded or unbounded) intervals with left end 0.

The symbols $\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\Leftrightarrow$ denote the logical operations of negation, disjunction, conjunction, implication, and equivalence respectively.

2.2 Multi-valued Functions

A multi-valued function [28] assigns one or more resulting values to each argument value. An application of a multi-valued function to an argument is interpreted as a nondeterministic choice of a result.

Definition 1 ([28]). *A (total) multi-valued function from a set A to a set B (denoted as $f : A \xrightarrow{tm} B$) is a function $f : A \to 2^B \backslash \{\emptyset\}$.*

Thus the inclusion $y \in f(x)$ means that y is a possible value of f on x.

2.3 Named Sets

We will use a simple notion of a named set to formalize an assignment of values to variable names in program and system semantics.

Definition 2. *([28]) A named set is a partial function $f : V \tilde{\to} W$ from a non-empty set of names V to a set of values W.*

A named set can be considered as special case of a more general notion of nominative data [28] which reflects hierarchical data organizations.

We will use a special notation for the set of named sets: ${}^V W$ denotes the set of all named sets $f : V \tilde{\to} W$ (this notation just emphasises that V is interpreted as a set of names). We consider named sets equal, if their graphs are equal.

An expression of the form $[n_1 \mapsto a_1, n_2 \mapsto a_2, ...]$ (where $n_1, n_2, ...$ are distinct names) denotes a named set d such that the graph of d is $\{(n_1, a_1), (n_2, a_2), ...\}$. A nowhere-defined named set is called an *empty named set* and is denoted as $[]$.

For any named sets d_1, d_2 we write $d_1 \subseteq d_2$ (*named set inclusion*), if the graph of a function d_1 is a subset of the graph of d_2.

2.4 Signals, Signal Bunches, and Blocks

Informally, a block is an abstract model of a system which receives input signals and produces output signals (Fig. 1). The input signals can be thought of as certain time-varying characteristics (attributes) of the external environment of the system which are relevant for (the operation of) this system. Each instance of an input signal has a certain time domain on which it is defined (present). An input signal bunch can be thought of as a collection of instances of input signals of the system. Each input signal bunch i has an associated domain of existence ($dom(i)$) which is a superset of the union of the domains of signals contained in i. The domain of an input signal bunch can be thought of as a time span of the existence of the external environment of the system. The output signals can be considered as effects (results) of the system's operation. An output signal bunch, or simply an output of the block, can be thought of as a collection of instances of output signals of the system. The output signals have domains of definition (presence) and each output signal bunch o has an associated domain of existence ($dom(o)$) which is a superset of the union of the domains of signals contained in o. The domain of an output signal bunch can be thought of as a time span during which the system operates. It is assumed that for an output signal bunch o which corresponds to a given input signal bunch i the inclusion $dom(o) \subseteq dom(i)$ holds (i.e. the system does not operate when the environment does not exist). However, in the general case, the presence of a given input signal at a given time does not imply the presence of a certain output signal at the same or any other time moment.

A block can operate nondeterministically, i.e. for one input signal bunch it may choose an output signal bunch from a set of possible variants. However, for any input signal bunch there exists at least one corresponding output signal bunch (although the values of all signals in it may be absent at all times, which means that the block does not produce any output values).

Normally, a block processes the whole input signal bunch, and does or does not produce output values. However, in certain cases a block may not process the whole input signal bunch and may terminate at some time moment before its end. This situation is interpreted as an abnormal termination of a block.

Let W be a fixed non-empty set of values.

Definition 3 (**[26]**)

(1) A signal is a partial function from T to W ($f : T \tilde{\to} W$).
(2) A V-signal bunch (where V is a set of names) is a function $sb : T \tilde{\to} {}^{V} W$ such that $dom(sb) \in \mathcal{T}_0$. The set of all V-signal bunches is denoted as $Sb(V, W)$.
(3) A signal bunch is a V-signal bunch for some V.
(4) A signal bunch sb is trivial, if $dom(sb) = \emptyset$ and is total, if $dom(sb) = T$. A trivial signal bunch is denoted as $\perp$.
(5) For a given signal bunch sb, a signal corresponding to a name x is a partial function $t \mapsto sb(t)(x)$. This signal is denoted as $sb[x]$.
(6) A signal bunch sb_1 is a prefix of a signal bunch sb_2 (denoted as $sb_1 \preceq sb_2$), if $sb_1 = sb_2|_A$ for some $A \in \mathcal{T}_0$.

The relation $\preceq$ on V-signal bunches is a partial order (for an arbitrary V).

It can be generalized to pairs as follows: for any signal bunches sb_1, sb_2, sb'_1, sb'_2 denote $(sb_1, sb_2) \preceq^2 (sb'_1, sb'_2)$ iff there exists $A \in \mathcal{T}_0$ such that $sb_1 = sb'_1|_A$ and $sb_2 = sb'_2|_A$. The relation $\preceq^2$ is a partial order on pairs of signal bunches.

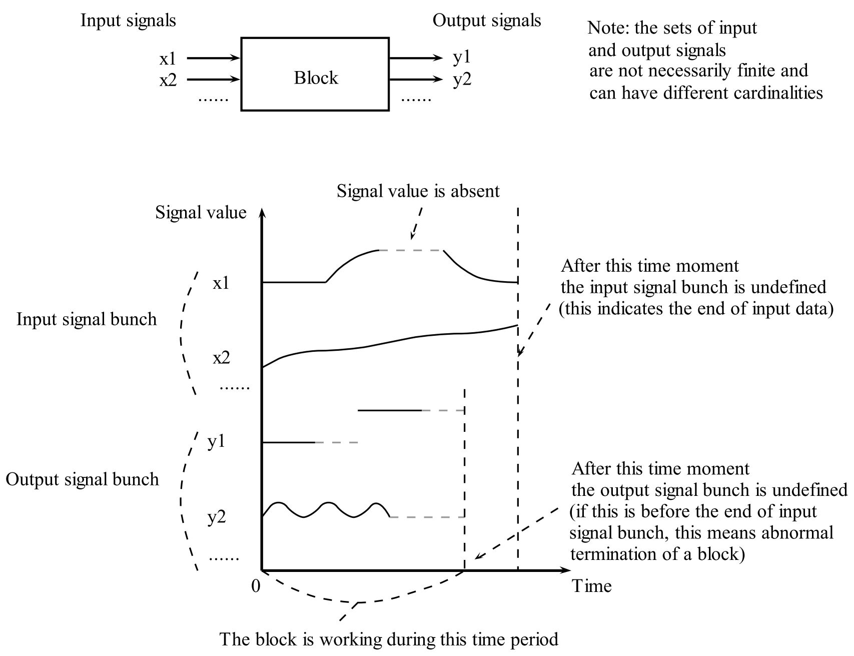

Input signals
Output signals
x1
x2
......
Block
y1
y2
Note: the sets of input and output signals are not necessarily finite and can have different cardinalities
Signal value is absent
Signal value
Input signal bunch
Output signal bunch
After this time moment the input signal bunch is undefined (this indicates the end of input data)
After this time moment the output signal bunch is undefined (if this is before the end of input signal bunch, this means abnormal termination of a block)
0
Time
The block is working during this time period

Fig. 1. An illustration of a block with the input signals x_1, x_2, ... and the output signals y_1, y_2, ... The input and output signals are lumped into an input and output signal bunch respectively. Solid curves represent (present) signal values. Dashed horizonal segments denote absence of a signal value. Dashed vertical lines indicate the right boundaries of the domains of signal bunches.

A block has a syntactic aspect (e.g. a description in a specification language) and a semantic aspect – a partial multi-valued function on signal bunches.

Definition 4. *(1) A block is an object B (syntactic aspect) together with an associated set of input names $In(B)$, a set of output names $Out(B)$, and a total multi-valued function $Op(B) : Sb(In(B), W) \xrightarrow{tm} Sb(Out(B), W)$ (operation, semantic aspect) such that $o \in Op(B)(i)$ implies $dom(o) \subseteq dom(i)$.*

(2) Two blocks B_1, B_2 are semantically identical, if $In(B_1) = In(B_2)$, $Out(B_1) = Out(B_2)$, and $Op(B_1) = Op(B_2)$.

(3) An I/O pair of a block B is a pair of signal bunches (i, o) such that $o \in Op(B)(i)$. The set of all I/O pairs of B is denoted as $IO(B)$ and is called the input-output (I/O) relation of B.

An inclusion $o \in Op(B)(i)$ means that o is a possible output of a block B on the input i. For each input i there is some output o. The domain of o is a subset of the domain of i. If o becomes undefined at time t, but i is still defined at t, we interpret this as an (unrecoverable) error during the operation of the block B.

2.5 Causal and Strongly Nonanticipative Blocks

Definition 5. *A block B is deterministic, if $Op(B)(i)$ is a singleton set for each $In(B)$-signal bunch i.*

Definition 6. *A deterministic block B is causal iff for all signal bunches i_1, i_2 and $A \in \mathcal{T}_0$, $o_1 \in Op(B)(i_1)$, $o_2 \in Op(B)(i_2)$, the equality $i_1|_A = i_2|_A$ implies $o_1|_A = o_2|_A$.*

This means that the value of the output signal bunch at time t can depend only on the values of the input signal at times $\leq t$.

Definition 7. *A block B is a sub-block of a block B' (denoted as $B \trianglelefteq B'$), if $In(B) = In(B')$, $Out(B) = Out(B')$, and $IO(B) \subseteq IO(B')$.*

Definition 8. *A block B is strongly nonanticipative, if for each $(i, o) \in IO(B)$ there exists a deterministic causal sub-block $B' \trianglelefteq B$ such that $(i, o) \in IO(B')$.*

Informally, the operation of a strongly nonanticipative block B can be interpreted as a two-step process:

1. Before receiving the input signals, the block B (nondeterministically) chooses a deterministic causal sub-block $B' \trianglelefteq B$ (response strategy).
2. The block B' receives input signals of B and produces the corresponding output signals (response) which become the output signals of B.

2.6 Nondeterministic Complete Markovian Systems (NCMS)

The notion of a NCMS was introduced in [27] as a special kind of abstract dynamical systems for the purpose of studying the relation between existence of global and local trajectories of dynamical systems. This notion is close to the notion of a *solution system* introduced by O. Hájek in [29], but there are some more and less important differences which will be described below.

Let $T = \mathbb{R}_+$ be the positive real time scale. Denote by $\mathfrak{T}$ the set of all connected subsets of T (i.e. bounded and unbounded intervals) with cardinality greater than one.

Let Q be a set (a state space) and Tr be some set of functions of the form $s : A \to Q$, where $A \in \mathfrak{T}$. Let us call its elements (partial) trajectories.

Definition 9 ([27]). *A set of trajectories Tr is closed under proper restrictions (CPR), if $s|_A \in Tr$ for each $s \in Tr$ and $A \in \mathfrak{T}$ such that $A \subseteq dom(s)$.*

Definition 10 ([27])

(1) A trajectory $s_1 \in Tr$ is a subtrajectory of $s_2 \in Tr$ (denoted as $s_1 \sqsubseteq s_2$), if $dom(s_1) \subseteq dom(s_2)$ and $s_1 = s_2|_{dom(s_1)}$.
(2) A trajectory $s_1 \in Tr$ is a proper subtrajectory of $s_2 \in Tr$ (denoted as $s_1 \sqsubset s_2$), if $s_1 \sqsubseteq s_2$ and $s_1 \neq s_2$.
(3) Trajectories $s_1, s_2 \in Tr$ are incomparable, if neither $s_1 \sqsubseteq s_2$, nor $s_2 \sqsubseteq s_1$.

The set $(Tr, \sqsubseteq)$ is a (possibly empty) partially ordered set (poset).

Definition 11 ([27]). *A CPR set of trajectories Tr is*

(1) Markovian (see Fig. 2 below), if for each $s_1, s_2 \in Tr$ and $t \in T$ such that $t = \sup dom(s_1) = \inf dom(s_2)$, $s_1(t) \downarrow$, $s_2(t) \downarrow$, and $s_1(t) = s_2(t)$, the following function s belongs to Tr:
$$s(t) = \begin{cases} s_1(t), & t \in dom(s_1) \\ s_2(t), & t \in dom(s_2) \end{cases}$$
(2) complete, if each non-empty chain in $(Tr, \sqsubseteq)$ has a supremum.

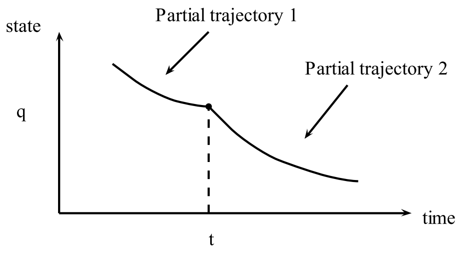


Fig. 2. Markovian property for nondeterministic systems with partial trajectories. If one partial trajectory ends and another begins in the state q at time t (both are defined at t), then their concatenation is a partial trajectory.

Definition 12 ([27]). *A nondeterministic complete Markovian system (NCMS) is a triple (T, Q, Tr), where Q is a set (state space) and Tr (trajectories) is a set of functions $s : T \tilde{\to} Q$ such that $dom(s) \in \mathfrak{T}$, which is CPR, complete, and Markovian.*

The main similarities and differences between a NCMS and a solution system [29] are:

- The time domain T and the set of states Q of NCMS correspond to the time domain R and the phase-space P of a solution system. For simplicity we assume that T is fixed to be $\mathbb{R}_+$, while in [29] R can can be any subset of $\mathbb{R}$.
- Trajectories of NCMS correspond to the members of a solution system (solutions). However, their domains cannot be singleton sets, while solutions can have singleton time domains.
- CPR property of NCMS basically corresponds to the Partialization property of solution systems. The difference is that Partialization allows restrictions on singleton sets, while CPR does not include this.
- Markovian property of NCMS basically corresponds to the Concatenation property of solution systems. By themselves these properties are not equivalent: Markovian property is weaker in the sense that it does not allow one to make a union of two trajectories, if the intersection of their domains is

not singleton. But using both CPR and Markovian properties, one can make a union of two trajectories even if their domains have non-singleton intersection. The term "Markovian" is meant to indicate that if a system is in a given state, the set of its possible futures does not depend in its past.
- In the stand-alone (process-independent) definition of a solution system [29, Definition 2.1], there is no assumption which corresponds / is analogous to the Completeness property of NCMS.

2.7 LR Representation of NCMS

In this subsection we will describe a convenient general representation of NCMS in terms of certain predicates.

Definition 13 ([27]). *Let $s_1, s_2 : T \tilde{\to} Q$. Then s_1 and s_2 coincide:*

(1) on a set $A \subseteq T$, if $s_1|_A = s_2|_A$ (this is denoted as $s_1 \doteq_A s_2$);
(2) in a left neighborhood of $t \in T$, if either $t = 0$ and $s_1(0) = s_2(0)$, or $t > 0$ and there exists $t' \in [0, t)$, such that $s_1 \doteq_{(t',t]} s_2$ (this is denoted as $s_1 \doteq_{t-} s_2$);
(3) in a right neighborhood of $t \in T$, if there exists $t' > t$, such that $s_1 \doteq_{[t,t')} s_2$ (this is denoted as $s_1 \doteq_{t+} s_2$).

Let Q be a set. Denote by $ST(Q)$ the set of pairs (s, t) where $s : A \to Q$ for some $A \in \mathfrak{T}$ and $t \in A$.

Definition 14 ([27]). *A predicate $p : ST(Q) \to Bool$ is called*

(1) left-local, if $p(s_1, t) \Leftrightarrow p(s_2, t)$ whenever $\{(s_1, t), (s_2, t)\} \subseteq ST(Q)$ and $s_1 \doteq_{t-} s_2$, and, moreover, $p(s, t)$ whenever t is the least element of $dom(s)$;
(2) right-local, if $p(s_1, t) \Leftrightarrow p(s_2, t)$ whenever $\{(s_1, t), (s_2, t)\} \subseteq ST(Q)$, $s_1 \doteq_{t+} s_2$, and, moreover, $p(s, t)$ whenever t is the greatest element of $dom(s)$.

Let us denote by $LR(Q)$ the set of all pairs (l, r), where $l : ST(Q) \to Bool$ is a left-local predicate and $r : ST(Q) \to Bool$ is a right-local predicate.

Definition 15. *A pair $(l, r) \in LR(Q)$ is called a LR representation of a NCMS $\Sigma = (T, Q, Tr)$, if*

$$Tr = \{s : A \to Q \mid A \in \mathfrak{T} \wedge (\forall t \in A \; l(s, t) \wedge r(s, t))\}.$$

Theorem 1. *(About LR representation)*

(1) Each pair $(l, r) \in LR(Q)$ is a LR representation of a NCMS with the set of states Q.
(2) Each NCMS has a LR representation.

The proof follows immediately from [27, Theorem 1] and is omitted here.

3 Representation of Strongly Nonanticipative Blocks

In this section we will introduce a representation of a strongly nonanticipative block in the form of an abstract dynamical system of a special kind (initial I/O NCMS). Although we mainly concentrate on the mere existence of such a representation, the proofs of lemmas given below actually describe a method for constructing a particular concrete representation of a block.

Let W denote a fixed non-empty set of values.

Definition 16. *An input-output (I/O) NCMS is an NCMS* (T, Q, Tr) *such that* Q *has a form* ${}^{I}W \times X \times {}^{O}W$ *for some sets* I *(set of input names),* $X \neq \emptyset$ *(set of internal states), and* O *(set of output names). The* ${}^{I}W$ *is called an input data set and* ${}^{O}W$ *is called an output data set.*

Informally, an I/O NCMS describes possible evolutions of triples (d_{in}, x, d_{out}) of input data ($d_{in} \in {}^{I}W$), internal state ($x \in X$), and output data ($d_{out} \in {}^{O}W$).

Lemma 1. *Each I/O NCMS* (T, Q, Tr) *has a unique set of input names, internal states, and output names.*

The proof follows immediately from the definitions.

For a I/O NCMS Σ we will denote as $In(\Sigma)$ its unique set of input names, as $Out(\Sigma)$ its set of output names, and as $IState(\Sigma)$ its internal state space.

For any I/O NCMS $\Sigma = (T, Q, Tr)$ and a state $q \in Q$ we will denote as $in(q)$, $istate(q)$, $out(q)$ the projections of q on the first, second, and third coordinate respectively. Correspondingly, for any $s \in Tr$, $in \circ s$, $istate \circ s$, $out \circ s$, denote a composition of the respective projection map with a trajectory.

For each $i \in Sb(In(\Sigma), W)$ let us denote

- $S(\Sigma, i) = \{s \in Tr \mid dom(s) \in \mathcal{T}_0 \ \wedge in \circ s \preceq i\}$;
- $S_{max}(\Sigma, i)$ is the set of all $\sqsubseteq$-maximal (i.e. non-continuable) trajectories from $S(\Sigma, i)$;
- $S_{init}(\Sigma, i) = \{s(0) \mid s \in S(\Sigma, i)\}$;
- $S_{init}(\Sigma) = \{s(0) \mid s \in Tr \wedge dom(s) \in \mathcal{T}_0\}$.

For each $Q' \subseteq Q$ let us denote:

$$Sel_{1,2}(Q', d, x) = \{q \in Q' \mid \exists d' \ q = (d, x, d')\},$$

i.e., selection of states from Q' by the value of the first and second component.

For each $Q' \subseteq Q$ and $i \in Sb(In(\Sigma), W)$ let us denote:

$$o_{all}(\Sigma, Q', i) = \begin{cases} \{\bot\}, & Q' = \emptyset \text{ or } i = \bot; \\ \{\{0\} \mapsto out(q) \mid q \in Q'\}, & Q' \neq \emptyset \text{ and } \\ & dom(i) = \{0\}; \\ \{out \circ s \mid s \in S_{max}(\Sigma, i) \wedge s(0) \in Q'\} \cup & Q' \neq \emptyset \text{ and } \\ \quad \cup \{\{0\} \mapsto out(q) \mid q \in Q' \backslash S_{init}(\Sigma, i)\}, & \{0\} \subset dom(i), \end{cases}$$

where $\{0\} \mapsto out(q)$ denotes a signal bunch defined on $\{0\}$ which takes a value $out(q)$. Also, for each $Q_0 \subseteq Q$ let us denote:

$$O_{all}(\Sigma, Q_0, i) = \begin{cases} \{\perp\}, & dom(i) = \emptyset; \\ \bigcup_{x \in IState(\Sigma)} o_{all}(\Sigma, Sel_{1,2}(Q_0, i(0), x), i), & dom(i) \neq \emptyset. \end{cases}$$

Definition 17. *An initial I/O NCMS is a pair (Σ, Q_0), where $\Sigma = (T, Q, Tr)$ is a I/O NCMS and Q_0 is a set (admissible initial states) such that $S_{init}(\Sigma) \subseteq Q_0 \subseteq Q$.*

Definition 18. *A NCMS representation of a block B is an initial I/O NCMS (Σ, Q_0) such that*

(1) $In(B) = In(\Sigma)$ and $Out(B) = Out(\Sigma)$;
(2) $Op(B)(i) = O_{all}(\Sigma, Q_0, i)$ for all $i \in Sb(In(B), W)$.

Informally, the operation of a block B represented by an initial I/O NCMS (Σ, Q_0) on an input signal bunch i can be described as follows:

(1) If $i(0)$ is undefined, then B stops (the output signal bunch is $\perp$).
(2) Otherwise, B chooses an arbitrary internal state $x \in IState(\Sigma)$.
(3) If there is no admissible initial state $q \in Q_0$ with $in(q) = i(0)$ and $istate(q) = x$ (i.e. $Sel_{1,2}(Q_0, i(0), x) = \emptyset$), then B stops.
(4) Otherwise, B chooses an arbitrary $q \in Q_0$ such that $in(q) = i(0)$ and $istate(q) = x$ (i.e. $q \in Sel_{1,2}(Q_0, i(0), x)$).
(5) If $dom(i) = \{0\}$ or there is no trajectory s which starts in q and is defined on some interval (of positive length) from $\mathcal{T}_0$, then B outputs $out(q)$ at time 0 and stops.
(6) Otherwise, B chooses an arbitrary maximal trajectory s defined on an interval from $\mathcal{T}_0$ such that $s(0) = q$ and $in \circ s \preceq i$ and outputs the signal bunch $out \circ s$.

The main result of this section is the following:

Theorem 2. *(About representation of a strongly nonanticipative block) Each strongly nonanticipative block has a NCMS representation.*

In the rest of the section we will prove several helper lemmas, and finally, give a proof of this theorem.

Lemma 2. *Let (T, Q, Tr) be a NCMS, Q' be a set, $f : Q \to Q'$ be an injective function, and $Tr' = \{f \circ s \mid s \in Tr\}$. Then (T, Q', Tr') is a NCMS.*

The proof follows immediately from the definition of NCMS and is omitted here.

Lemma 3. *Let (T, Q^j, Tr^j), $j \in J$ be an indexed family of NCMS such that $Q^j \cap Q^{j'} = \emptyset$, if $j \neq j'$. Let $Q = \bigcup_{j \in J} Q^j$ and $Tr = \bigcup_{j \in J} Tr^j$. Then (T, Q, Tr) is a NCMS.*

The proof follows immediately from the definition of NCMS and is omitted here.

Lemma 4. *Let Σ be a I/O NCMS, $i \in Sb(In(\Sigma), W)$, and $s \in S(\Sigma, i)$. Then there exists $s' \in S_{max}(\Sigma, i)$ such that $s \sqsubseteq s'$.*

Proof. Consider a set $G = \{s'' \in S(\Sigma, i) \mid s \sqsubseteq s''\}$. From the completeness property of NCMS it follows that each non-empty $\sqsubseteq$-chain of elements of G has a least upper bound in the poset $(Tr, \sqsubseteq)$ which belongs to G. Moreover, $G \neq \emptyset$. Then Zorn's lemma implies that G has some $\sqsubseteq$-maximal element s'. Then $s' \in S_{max}(\Sigma, i)$ and $s \sqsubseteq s'$. □

Lemma 5. *If $\Sigma = (T, Q, Tr)$ is a I/O NCMS, $Q' \subseteq Q$, $i \in Sb(In(\Sigma), W)$, then*

(1) $o_{all}(\Sigma, Q', i) \subseteq Sb(Out(\Sigma), W)$;
(2) $dom(o) \subseteq dom(i)$ for each o in $o_{all}(\Sigma, Q', i)$;
(3) $o_{all}(\Sigma, Q', i) \neq \emptyset$.

The proof follows from the definitions and Lemma 4 and is omitted here.

Lemma 6. *Each initial I/O NCMS is a NCMS representation of a unique (up to semantic identity) block.*

Proof. Uniqueness up to semantic identity is obvious from Definition 18. Let us prove that if (Σ, Q_0) is an initial I/O NCMS, where $\Sigma = (T, Q, Tr)$, then it is a NCMS representation of some block.

Let $i \in Sb(In(\Sigma), W)$. Let us show that $O_{all}(\Sigma, Q_0, i)$ is a non-empty subset of $Sb(Out(\Sigma), W)$ and $dom(o) \subseteq dom(i)$ for all $o \in O_{all}(\Sigma, Q_0, i)$. This is obvious, if $dom(i) = \emptyset$. Consider the case when $dom(i) \neq \emptyset$. Then $O_{all}(\Sigma, Q_0, i) = \bigcup_{x \in IState(\Sigma)} o_{all}(\Sigma, Sel_{1,2}(Q_0, i(0), x), i)$. For any $x \in IState(\Sigma)$ we have $Sel_{1,2}(Q_0, i(0), x) \subseteq Q_0 \subseteq Q$. Besides, $IState(\Sigma) \neq \emptyset$. Then Lemma 5 implies that $O_{all}(\Sigma, Q_0, i) \in 2^{Sb(Out(\Sigma), W)} \backslash \{\emptyset\}$ and $dom(o) \subseteq dom(i)$ for all $o \in O_{all}(\Sigma, Q_0, i)$. Thus (Σ, Q_0) is a NCMS representation of a block. □

Lemma 7. *Let B be a deterministic causal block. Then B has a NCMS representation.*

Proof (Sketch). Let $X = \{i \in Sb(In(B), W) \mid \exists\ t \in T\ dom(i) = [0, t]\}$ and $Q = {}^{In(B)}W \times X \times {}^{Out(B)}W$. Then $X \neq \emptyset$. Let *in*, *istate*, *out* denote projection maps from Q on the first, second, and third coordinate respectively. Let Tr be the set of all functions of the form $s : A \to Q$, where $A \in \mathfrak{T}$, such that

(a) for each $t \in dom(s) \backslash \{0\}$ we have $dom(istate(s(t))) = [0, t]$ and $istate(s(t))(t) = in(s(t))$ and if $s(0) \downarrow$, then $istate(s(t))(0) = in(s(0))$;
(b) for each $t \in dom(s)$ we have $t \in dom(o)$ and $out(s(t)) = o(t)$, where o is a unique member of $Op(B)(istate(s(t)))$;
(c) if $t_1, t_2 \in dom(s) \backslash \{0\}$ and $t_1 \leq t_2$, then $istate(s(t_1)) \sqsubseteq istate(s(t_2))$.

It is straightforward to check that $\Sigma = (T, Q, Tr)$ is a NCMS.

Let $i \in Sb(In(B), W)$ and $o \in Op(B)(i)$. Let us show that for each $s \in S(\Sigma, i)$, $out \circ s = o|_{dom(s)}$, and if $s \in S_{max}(\Sigma, i)$, then $out \circ s = o$.

Let $s \in S(\Sigma, i)$. Then $dom(s) \in \mathcal{T}_0$ and $in \circ s \preceq i$ by definition of $S(\Sigma, i)$ and $istate(s(t))(t) = in(s(t)) = i(t)$ for all $t \in dom(s)\backslash\{0\}$ by (a). If $t', t \in dom(s)$ and $0 < t' \leq t$, then $i(t') = istate(s(t'))(t') = istate(s(t))(t')$ by (c). Moreover, we have $s(0) \downarrow$, whence $istate(s(t))(0) = in(s(0)) = i(0)$ for each $t \in dom(s)\backslash\{0\}$ by (a). Then for each $t \in dom(s)\backslash\{0\}$, $istate(s(t)) = i|_{[0,t]}$, because $dom(istate(s(t))) = [0, t]$. Then $Op(B)(istate(s(t))) = Op(B)(i|_{[0,t]}) = \{o|_{[0,t]}\}$, because B is deterministic and causal. Then $out(s(t)) = (o|_{[0,t]})(t)$ and $t \in dom(o)$ for each $t \in dom(s)$ by (b). This implies that $dom(s) \subseteq dom(o)$ and for all $t \in dom(s)$, $out(s(t)) = o(t)$. Thus $out \circ s = o|_{dom(s)}$. We have $\{0\} \subset dom(s) \subseteq dom(o)$, so $dom(o) \in \mathfrak{T}$. Because $in(s(0)) = i(0)$ and $out(s(0)) = o(0)$, it is easy to see that a function $s' : dom(o) \to Q$ such that $s'(0) = s(0)$ and $s'(t) = (i(t), i|_{[0,t]}, o(t))$ for all $t \in dom(o)\backslash\{0\}$ satisfies (a), (b), and (c). Moreover, $s' \in Tr$, $dom(s') \in \mathcal{T}_0$, and $in \circ s' = i|_{dom(o)} \preceq i$. Then $s' \in S(\Sigma, i)$. Besides, $s'|_{dom(s)} = s$. Then if $s \in S_{max}(\Sigma, i)$, then $s' = s$ and $out \circ s = o$.

Let us denote $Q_0 = \{(d_{in}, x, d_{out}) \in Q \mid \exists(i, o) \in IO(B)\ \{0\} \in dom(o) \wedge d_{in} = i(0) \wedge d_{out} = o(0)\}$. It is straightforward to show that $S_{init}(\Sigma) \subseteq Q_0$. Thus (Σ, Q_0) is an initial I/O NCMS. Obviously, $In(\Sigma) = In(B)$, $Out(\Sigma) = Out(B)$.

It is easy to check that Q_0 satisfies the following property:

(d) if $(i, o) \in IO(B)$, $q \in Q_0$, $i \neq \perp$, $in(q) = i(0)$, then $o \neq \perp$ and $out(q) = o(0)$.

Now let us show that (Σ, Q_0) is a NCMS representation of B. It is sufficient to show that $Op(B)(i) = O_{all}(\Sigma, Q_0, i)$ for all $i \in Sb(In(B), W)\backslash\{\perp\}$.

Let $i \in Sb(In(B), W)\backslash\{\perp\}$ and $o \in Op(B)(i)$. Consider the following cases:

(1) $Sel_{1,2}(Q_0, i(0), x) = \emptyset$ for some $x \in IState(\Sigma)$. Then there is no $(i', o') \in IO(B)$ such that $i'(0) = i(0)$ and $o'(0) \downarrow$. Then $o = \perp$ and $Sel_{1,2}(Q_0, i(0), x) = \emptyset$ for all $x \in IState(\Sigma)$. Then $O_{all}(\Sigma, Q_0, i) = \{\perp\} = Op(B)(i)$.

(2) $Sel_{1,2}(Q_0, i(0), x) \neq \emptyset$ for all $x \in IState(\Sigma)$ and $dom(i) = \{0\}$. Then $o(0) \downarrow$ and $out(q) = o(0)$ for each $q \in Sel_{1,2}(Q_0, i(0), x) \subseteq Q_0$ by the property (d). Then $o_{all}(\Sigma, Sel_{1,2}(Q_0, i(0), x), i) = \{\{0\} \mapsto o(0)\} = \{o\}$ for all $x \in IState(\Sigma)$, whence $O_{all}(\Sigma, Q_0, i) = Op(B)(i)$.

(3) $Sel_{1,2}(Q_0, i(0), x) \neq \emptyset$ for all $x \in IState(\Sigma)$, $\{0\} \subset dom(i)$, and $dom(o) \subseteq \{0\}$. If $in(q) = i(0)$ for some $q \in S_{init}(\Sigma, i)$, then $q = s(0)$ for some $s \in S(\Sigma, i)$, whence $out \circ s = o|_{dom(s)}$ as we have shown above, but this is impossible, because $\{0\} \subset dom(s)$ and $dom(o) \subseteq \{0\}$. Thus $in(q) \neq i(0)$ for each $q \in S_{init}(\Sigma, i)$. Then for each $x \in IState(\Sigma)$, $s(0) \notin Sel_{1,2}(Q_0, i(0), x)$ for all $s \in S_{max}(\Sigma, i)$ and $Sel_{1,2}(Q_0, i(0), x) \cap S_{init}(\Sigma, i) = \emptyset$.
Then $o_{all}(\Sigma, Sel_{1,2}(Q_0, i(0), x), i) = \{\{0\} \mapsto out(q) \mid q \in Sel_{1,2}(Q_0, i(0), x)\}$ for each $x \in IState(\Sigma)$, whence $O_{all}(\Sigma, Q_0, i) = \{\{0\} \mapsto out(q) \mid q \in Q_0 \wedge in(q) = i(0)\} \neq \emptyset$. Because $in(q) = i(0)$ for some $q \in Q_0$, by the property (d) we have $0 \in dom(o)$ and $O_{all}(\Sigma, Q_0, i) = \{\{0\} \mapsto o(0)\} = \{o\} = Op(B)(i)$.

(4) $Sel_{1,2}(Q_0, i(0), x) \neq \emptyset$ for all $x \in IState(\Sigma)$ and $\{0\} \subset dom(o)$. We have $dom(o) \in \mathfrak{T}$. Let $x \in IState(\Sigma)$ and $q \in Sel_{1,2}(Q_0, i(0), x)$. Then $in(q) = i(0)$ and have $out(q) = o(0)$ by the property (d). It is easy to see that a function $s' : dom(o) \to Q$ such that $s'(0) = q$ and $s'(t) = (i(t), i|_{[0,t]}, o(t))$ for all $t \in dom(o)\backslash\{0\}$ satisfies (a), (b), and (c). Moreover, $s' \in Tr$, $dom(s') \in$

$\mathcal{T}_0$, and $in \circ s' = i|_{dom(o)} \preceq i$. Then $s' \in S(\Sigma, i)$. Then $s'(0) = q \in S_{init}(\Sigma, i)$. Because $q \in Sel_{1,2}(Q_0, i(0), x)$ is arbitrary, we have $Sel_{1,2}(Q_0, i(0), x) \subseteq S_{init}(\Sigma, i)$. Then $o_{all}(\Sigma, Sel_{1,2}(Q_0, i(0), x), i) = \{out \circ s \mid s \in S_{max}(\Sigma, i) \wedge s(0) \in Sel_{1,2}(Q_0, i(0), x)\} = \{o\}$ for each $x \in IState(\Sigma)$, because $out \circ s = o$ for any $s \in S_{max}(\Sigma, i)$ as we have show above and $Sel_{1,2}(Q_0, i(0), x) \neq \emptyset$. Then $O_{all}(\Sigma, Q_0, i) = \{o\} = Op(B)(i)$, because $IState(\Sigma) \neq \emptyset$.

In all cases (1)-(4) we have $O_{all}(\Sigma, Q_0, i) = Op(B)(i)$. Thus (Σ, Q_0) is a NCMS representation of the block B. □

Let $\Sigma_1 = (T, Q_1, Tr_1)$ and $\Sigma_2 = (T, Q_2, Tr_2)$ be I/O NCMS such that $In(\Sigma_1) = In(\Sigma_2)$ and $Out(\Sigma_1) = Out(\Sigma_2)$.

Let us introduce the following notions.

Definition 19. *(1) a state embedding from Σ_1 to Σ_2 is a function $f : Q_1 \to Q_2$ such that $\{f \circ s \mid s \in Tr_1\} = \{s \in Tr_2 \mid \exists t \in dom(s)\ \exists q \in Q_1\ s(t) = f(q)\}$ and there exists an injective function $g : IState(\Sigma_1) \to IState(\Sigma_2)$ such that for all $q \in Q_1$, $f(q) = (in(q), g(istate(q)), out(q))$.*

(2) A state embedding from an initial I/O NCMS (Σ_1, Q_0^1) to an initial I/O NCMS (Σ_2, Q_0^2) is a state embedding f from Σ_1 to Σ_2 such that for each $q \in Q_1$, $q \in Q_0^1$ iff $f(q) \in Q_0^2$.

Note that it follows immediately from this definition that a state embedding from Σ_1 to Σ_2 is an injective function.

Lemma 8. *Let $\Sigma_1 = (T, Q_1, Tr_1)$ and $\Sigma_2 = (T, Q_2, Tr_2)$ be I/O NCMS, $In(\Sigma_1) = In(\Sigma_2)$ and $Out(\Sigma_1) = Out(\Sigma_2)$, and f be a state embedding from Σ_1 to Σ_2. Let $i \in Sb(In(\Sigma_1), W)$. Then $S_{max}(\Sigma_2, i) \supseteq \{f \circ s \mid s \in S_{max}(\Sigma_1, i)\}$ and $\{q \in S_{init}(\Sigma_2, i) \mid \exists q' \in Q_1\ q = f(q')\} = \{f(q'') \mid q'' \in S_{init}(\Sigma_1, i)\}$.*

The proof follows immediately from the definitions and is omitted here.

Lemma 9. *For $j = 1, 2$ let (Σ_j, Q_0^j) be a NCMS representation of a block B_j. Assume that $In(\Sigma_1) = In(\Sigma_2)$ and $Out(\Sigma_1) = Out(\Sigma_2)$ and there exists a state embedding f from (Σ_1, Q_0^1) to (Σ_2, Q_0^2). Then $B_1 \trianglelefteq B_2$.*

Proof (Sketch). Assume that $\Sigma_1 = (T, Q_1, Tr_1)$ and $\Sigma_2 = (T, Q_2, Tr_2)$. We have $In(\Sigma_1) = In(\Sigma_2)$ and $Out(\Sigma_1) = Out(\Sigma_2)$. Because f is a state embedding, there exists an injective function $g : IState(\Sigma_1) \to IState(\Sigma_2)$ such that $f(q) = (in(q), g(istate(q)), out(q))$ for all $q \in Q$.

Let $i \in Sb(In(B), W)$. Then for $j = 1, 2$, $Op(B_j)(i) = O_{all}(\Sigma_j, Q_0^j, i)$.

Let us show that $O_{all}(\Sigma_2, Q_0^2, i) \supseteq O_{all}(\Sigma_1, Q_0^1, i)$. This is obvious, if $i = \perp$, so assume that $i \neq \perp$. Let us fix some $x_1 \in IState(\Sigma_1)$. Denote $Q_1' = Sel_{1,2}(Q_0^1, i(0), x_1)$ and $Q_2' = Sel_{1,2}(Q_0^2, i(0), g(x_1))$. Because g is injective and $Q_0^2 \supseteq \{f(q) \mid q \in Q_0^1\}$, it is straightforward to show that $Q_2' \supseteq \{f(q) \mid q \in Q_1'\}$ and $Q_2' \neq \emptyset$ iff $Q_1' \neq \emptyset$.

Let us show that $o_{all}(\Sigma_2, Q_2', i) \supseteq o_{all}(\Sigma_1, Q_1', i)$. This is obvious, if $Q_1' = \emptyset$ or $Q_2' = \emptyset$, so assume that $Q_1' \neq \emptyset$ and $Q_2' \neq \emptyset$.

Consider the case when $dom(i) = \{0\}$. Because $Q'_2 \supseteq \{f(q) \mid q \in Q'_1\}$, it is easy to check that $o_{all}(\Sigma_2, Q'_2, i) = o_{all}(\Sigma_1, Q'_1, i)$.

Consider the case when $\{0\} \subset dom(i)$. By Lemma 8 we have $S_{max}(\Sigma_2, i) \supseteq \{f \circ s \mid s \in S_{max}(\Sigma_1, i)\}$ and $\{q \in S_{init}(\Sigma_2, i) \mid \exists q' \in Q_1 \; q = f(q')\} = \{f(q'') \mid q'' \in S_{init}(\Sigma_1, i)\}$. Because f is injective and $Q'_2 \supseteq \{f(q) \mid q \in Q'_1\}$,

$$\{out \circ s \mid s \in S_{max}(\Sigma_2, i) \wedge s(0) \in Q'_2\} \supseteq \{out \circ (f \circ s) \mid s \in S_{max}(\Sigma_1, i) \wedge \\ \wedge f(s(0)) \in Q'_2\} \supseteq \{out \circ s \mid s \in S_{max}(\Sigma_1, i) \wedge s(0) \in Q'_1\}.$$

Because $Q'_1 \subseteq Q_1$ and f is injective, it is straightforward to check that $Q'_2 \backslash S_{init}(\Sigma_2, i) \supseteq \{f(q) \mid q \in Q'_1 \backslash S_{init}(\Sigma_1, i)\}$. Then from the definition of o_{all} it follows that $o_{all}(\Sigma_2, Q'_2, i) \supseteq o_{all}(\Sigma_1, Q'_1, i)$.

Because $x_1 \in IState(\Sigma_1)$ is arbitrary, it easily follows that $Op(B_2)(i) \supseteq Op(B_1)(i)$. We conclude that $B_1 \trianglelefteq B_2$. □

Definition 20. *A disjoint union of an indexed family of initial I/O NCMS* $((\Sigma_j, Q_0^j))_{j \in J}$, *where* $J \neq \emptyset$ *and* $\Sigma_j = (T, Q_j, Tr_j)$ *for each* $j \in J$, *is a pair* (Σ, Q_0), *where* $\Sigma = (T, Q, Tr)$ *and*

(1) $Q = {}^{IN}W \times (\bigcup_{j \in J} \{j\} \times IState(\Sigma_j)) \times {}^{OUT}W$, *where* $IN = \bigcup_{j \in J} In(\Sigma_j)$, *and* $OUT = \bigcup_{j \in J} Out(\Sigma_j)$;
(2) $Tr = \{f_j \circ s \mid j \in J \wedge s \in Tr_j\}$;
(3) $Q_0 = \{f_j(q) \mid j \in J \wedge q \in Q_0^j\}$;

where for each $j \in J$, $f_j : Q_j \to Q$ *is a function such that*

$$f_j(q) = (in(q), (j, istate(q)), out(q)), \;\; q \in Q_j.$$

Lemma 10. *Let* (Σ, Q_0) *be a disjoint union of an indexed family of initial I/O NCMS* $((\Sigma_j, Q_0^j))_{j \in J}$, *where* $J \neq \emptyset$. *Then* (Σ, Q_0) *is an initial I/O NCMS.*

The proof is straightforward and is omitted here.

Definition 21. *(1) A complete set of sub-blocks of a block* B *is a set* $\mathcal{B}$ *of sub-blocks of* B *such that* $IO(B) = \bigcup_{B' \in \mathcal{B}} IO(B')$.
(2) A complete indexed family of sub-blocks of a block B *is an indexed family* $(B_j)_{j \in J}$ *such that* $\{B_j \mid j \in J\}$ *is a complete set of sub-blocks of* B.

Definition 22. *A state-restriction of a NCMS* $\Sigma = (T, Q, Tr)$ *on a set* Q', *denoted as* $\Sigma|_{Q'}$, *is a triple* $(T, Q \cap Q', \{s \in Tr \mid \forall t \in dom(s) \; s(t) \in Q'\})$.

Lemma 11. $\Sigma|_{Q'}$ *is a NCMS for each NCMS* $\Sigma = (T, Q, Tr)$ *and set* Q',

The proof follows immediately from the definition of NCMS and is omitted here.

Lemma 12. *Let* $(B_j)_{j \in J}$ *be a complete indexed family of sub-blocks of a block* B, *where* $J \neq \emptyset$. *Assume that for each* $j \in J$, B_j *has a NCMS representation* (Σ_j, Q_0^j). *Let* (Σ, Q_0) *be a disjoint union of* $((\Sigma_j, Q_0^j))_{j \in J}$. *Then* (Σ, Q_0) *is a NCMS representation of* B.

Proof (Sketch). Assume that $\Sigma_j = (T, Q_j, Tr_j)$ for each $j \in J$ and $\Sigma = (T, Q, Tr)$.

By Lemma 10, (Σ, Q_0) is an initial I/O NCMS, whence by Lemma 6, there exists a block B' (unique up to semantic identity) such that (Σ, Q_0) is a NCMS representation of B'.

For each $j \in J$ we have $In(\Sigma_j) = In(B_j) = In(B)$ and $Out(\Sigma_j) = Out(B_j) = Out(B)$, because $B_j \trianglelefteq B$. Because $J \neq \emptyset$, $In(B') = In(\Sigma) = \bigcup_{j \in J} In(\Sigma_j) = In(B)$ and $Out(B') = Out(\Sigma) = \bigcup_{j \in J} Out(\Sigma_j) = Out(B)$.

For each $j \in J$, let $g_j : IState(\Sigma_j) \to IState(\Sigma)$ and $f_j : Q_j \to Q$ be functions such that $g_j(x) = (j, x)$ for all $x \in IState(\Sigma_j)$ and $f_j(q) = (in(q), g_j(istate(q)), out(q))$ for all $q \in Q_j$.

Using Lemma 9 it is not difficult to show that $B \trianglelefteq B'$.

Let us show that $B' \trianglelefteq B$. Let $(i, o) \in IO(B')$. Then $o \in O_{all}(\Sigma, Q_0, i)$. If $i = \perp$, then $o = \perp$ and $(i, o) \in IO(B)$ (because B is a block). Consider the case when $i \neq \perp$. Then there exists $x^* \in IState(\Sigma) = \bigcup_{j \in J}\{j\} \times IState(\Sigma_j)$ such that $o \in o_{all}(\Sigma, Sel_{1,2}(Q_0, i(0), x^*), i)$. Then there exists $j \in J$ and $x_j^* \in IState(\Sigma_j)$ such that $x^* = (j, x_j^*)$.

Let $Q_j' = {}^{In(\Sigma)}W \times (\{j\} \times IState(\Sigma_j)) \times {}^{Out(\Sigma)}W$, and $\Sigma_j' = \Sigma|_{Q_j'}$. By Lemma 11, Σ_j' is a NCMS. We will denote by Tr_j' the set of trajectories of Σ_j'. Moreover, Q_j' is the set of states of Σ_j' and $In(\Sigma_j') = In(\Sigma)$, $Out(\Sigma_j') = Out(\Sigma)$. Besides, Σ_j' is an I/O NCMS and $S_{init}(\Sigma_j') = S_{init}(\Sigma|_{Q_j'}) \subseteq S_{init}(\Sigma) \cap Q_j' \subseteq Q_0 \cap Q_j' \subseteq Q_j'$, because (Σ, Q_0) is an initial I/O NCMS. Denote $Q_{0,j}' = Q_0 \cap Q_j'$. Then $(\Sigma_j', Q_{0,j}')$ is an initial I/O NCMS. Moreover, $x^* \in IState(\Sigma_j')$.

It is straightforward to show that $o \in O_{all}(\Sigma_j', Q_{0,j}', i)$. By Lemma 6, there exists a block B_j' such that $(\Sigma_j', Q_{0,j}')$ is a NCMS representation of B_j'. Let $g : IState(\Sigma_j') \to IState(\Sigma_j)$ and $f : Q_j' \to Q_j$ be functions such that $g((j, x)) = x$ for $x \in IState(\Sigma_j')$ and $f(q) = (in(q), g(istate(q)), out(q))$ for all $q \in Q_j'$. Obviously, $In(\Sigma_j') = In(\Sigma_j)$, $Out(\Sigma_j') = Out(\Sigma_j)$, and g is injective. Moreover, f is an inverse of f_j, whence $\{f \circ s \mid s \in Tr_j'\} = Tr_j$. Because for any $s \in Tr_j$, $dom(s) \neq \emptyset$ and for each $t \in dom(s)$, $s(t) = f(f_j(s(t)))$, where $f_j(s(t)) \in Q_j'$, we have $\{f \circ s \mid s \in Tr_j'\} = \{s \in Tr_j \mid \exists t \in dom(s)\ \exists q \in Q_j'\ s(t) = f(q)\}$. Then f is a state embedding from Σ_j' to Σ_j. Moreover, for each $q \in Q_j'$, $q \in Q_{0,j}' = Q_0 \cap Q_j'$ iff $q = f_j(q')$ for some $q' \in Q_0^j$ iff $f(q) \in Q_0^j$. Then f is a state embedding from $(\Sigma_j', Q_{0,j}')$ to (Σ_j, Q_0^j). Then $B_j' \trianglelefteq B_j$ by Lemma 9. Because $o \in O_{all}(\Sigma_j', Q_{0,j}', i) = Op(B_j')(i)$, we have $o \in Op(B_j)(i)$, whence $(i, o) \in IO(B)$. Thus $B' \trianglelefteq B$. We conclude that $B \trianglelefteq B'$ and $B' \trianglelefteq B$, so B and B' are semantically identical and (Σ, Q_0) is a NCMS representation of B. $\square$

Now we can prove Theorem 2.

Proof (of Theorem 2).

Let B be a strongly nonanticipative block. Let us show that B has a NCMS representation. Let $\mathcal{R}$ be the set of all relations $R \subseteq IO(B)$ such that R is an I/O relation of a deterministic causal block. For each $R \in \mathcal{R}$ let us define a block B_R such that $IO(B_R) = R$, $In(B_R) = In(B)$, $Out(B_R) = Out(B)$. Then B_R is a deterministic causal block for each $R \in \mathcal{R}$ and $IO(B) = \bigcup_{R \in \mathcal{R}} IO(B_R)$, because B is strongly nonanticipative. Then $(B_R)_{R \in \mathcal{R}}$ is a complete indexed family of sub-blocks of B and $\mathcal{R} \neq \emptyset$. By Lemma 7, for each $R \in \mathcal{R}$ there exists an initial I/O NCMS (Σ_R, Q_0^R) which is a NCMS representation of B_R. Let (Σ, Q_0) be a disjoint union of $((\Sigma_R, Q_0^R))_{R \in \mathcal{R}}$. Then by Lemma 12, (Σ, Q_0) is a NCMS representation of B. □

4 Existence of Total I/O Pairs of Strongly Nonanticipative Blocks

4.1 Using NCMS Representation

The following theorems show that the questions (A) and (B) formulated in Section 1 can be reduced to the problem of existence of total trajectories of NCMS.

Theorem 3. *Let B be a strongly nonanticipative block and (Σ, Q_0) be its NCMS representation, where $\Sigma = (T, Q, Tr)$. Then B has a total I/O pair iff there exists $s \in Tr$ such that $dom(s) = T$.*

Proof. Let us prove the "If" part. Assume that $s \in Tr$ and $dom(s) = T$. Let $q_0 = s(0)$, $x = istate(q_0)$, $i = in \circ s$, $o = out \circ s$, and $Q' = Sel_{1,2}(Q_0, i(0), x)$. Then $q_0 \in S_{init}(\Sigma) \subseteq Q_0$, whence $q_0 \in Q'$, so $Q' \neq \emptyset$. Besides, $s \in S_{max}(\Sigma, i)$, because $dom(s) = T$ and $in \circ s = i \preceq i$. Then because $s(0) \in Q'$, we have $o = out \circ s \in o_{all}(\Sigma, Q', i)$ by the definition of o_{all}. Then $o \in O_{all}(\Sigma, Q_0, i) = Op(B)(i)$, because $i \neq \perp$ and (Σ, Q_0) is a NCMS representation of B. Then $(i, o) \in IO(B)$ and $dom(i) = dom(o) = T$. Thus B has a total I/O pair.

Now let us prove the "Only if" part. Assume that B has a total I/O pair $(i, o) \in IO(B)$. Because (Σ, Q_0) is a NCMS representation of B and $i \neq \perp$, we have $o \in O_{all}(\Sigma, Q_0, i)$. Then there is $x \in IState(\Sigma)$ such that $o \in o_{all}(\Sigma, Q', i)$, where $Q' = Sel_{1,2}(Q_0, i(0), x)$. Then $o = out \circ s$ for some $s \in S_{max}(\Sigma, i)$ such that $s(0) \in Q'$, because $dom(o) = T$. Then $s \in Tr$, $dom(s) = T$. □

Theorem 4. *Let B be a strongly nonanticipative block and (Σ, Q_0) be its NCMS representation, where $\Sigma = (T, Q, Tr)$.*

Let $i \in Sb(In(B), W)$ and $dom(i) = T$. Let (l, r) be a LR representation of Σ and $l' : ST(Q) \to Bool$ and $r' : ST(Q) \to Bool$ be predicates such that

$$l'(s,t) \Leftrightarrow l(s,t) \wedge (\min dom(s) \downarrow= t \vee in(s(t)) = i(t)).$$

$$r'(s,t) \Leftrightarrow r(s,t) \wedge (\max dom(s) \downarrow= t \vee in(s(t)) = i(t)).$$

Then

(1) $(l', r') \in LR(Q)$;
(2) If (l', r') is a LR representation of a NCMS $\Sigma' = (T, Q, Tr')$, then $\{o \in Op(B)(i) \mid dom(o) = T\} \neq \emptyset$ iff there exists $s \in Tr'$ such that $dom(s) = T$.

Proof (Sketch).

(1) It is straightforward to check that l' is left-local and r' is right-local.
(2) Assume that (l', r') is a LR representation of a NCMS $\Sigma' = (T, Q, Tr')$. Then $Tr' = \{s : A \to Q \mid A \in \mathfrak{T} \wedge (\forall t \in A\ l'(s,t) \wedge r'(s,t))\}$.

It is straightforward to show that $\{s \in Tr' \mid dom(s) \in \mathcal{T}_0\} = S(\Sigma, i)$.

Let us show that $\{o \in Op(B)(i) \mid dom(o) = T\} \neq \emptyset$ iff there exists $s \in Tr'$ such that $dom(s) = T$.

"If" Assume that $s \in Tr'$ and $dom(s) = T$. Then $s \in S(\Sigma, i)$. Let $q_0 = s(0)$, $x = istate(q_0)$, $o = out \circ s$, and $Q' = Sel_{1,2}(Q_0, i(0), x)$. Then $q_0 \in S_{init}(\Sigma') \subseteq S_{init}(\Sigma) \subseteq Q_0$, whence $q_0 \in Q'$, so $Q' \neq \emptyset$. Besides, $s \in S_{max}(\Sigma, i)$, because $dom(s) = T$. Then because $s(0) \in Q'$, we have $o = out \circ s \in o_{all}(\Sigma, Q', i)$ by the definition of o_{all}. Then $o \in O_{all}(\Sigma, Q_0, i) = Op(B)(i)$, because $i \neq \perp$ and (Σ, Q_0) is a NCMS representation of B. Besides, $dom(o) = T$. Thus $\{o \in Op(B)(i) \mid dom(o) = T\} \neq \emptyset$.

"Only if" Assume that $o \in Op(B)(i)$ and $dom(o) = T$. Because (Σ, Q_0) is a NCMS representation of B and $i \neq \perp$, we have $o \in O_{all}(\Sigma, Q_0, i)$. Then there is $x \in IState(\Sigma)$ such that $o \in o_{all}(\Sigma, Q', i)$, where $Q' = Sel_{1,2}(Q_0, i(0), x)$. Then $o = out \circ s$ for some $s \in S_{max}(\Sigma, i)$ such that $s(0) \in Q'$, because $dom(o) = T$. Then $s \in S(\Sigma, i)$, whence $s \in Tr'$ and $dom(s) = T$.

$\square$

Now we will focus on the problem of existence of total trajectories of a NCMS.

4.2 Existence of Globally Defined Trajectories of NCMS

An obvious method for proving existence of a total trajectory of a NCMS with a given LR representation (l, r) is just guessing a function $s : T \to Q$ such that $\forall t \in T\ l(s,t) \wedge r(s,t)$.

As an alternative to guessing an entire trajectory one can try to find/guess for each t a partial trajectory s_t defined in a neighborhood of t which satisfies $l(s_t, t) \wedge r(s_t, t)$ in such a way that all s_t, $t \in T$ can be glued together into a total function. An important aspect here is that the admissible choices of s_t, $s_{t'}$ for distant time moments $t, t' \in T$ (i.e. such that s_t, $s_{t'}$ appear as subtrajectories of some total trajectory) can be dependent.

However, this method can be generalized: instead of guessing an exact total trajectory or its exact locally defined subtrajectories, one can guess some "region" (subset of trajectories) which presumably contains a total trajectory and has some convenient representation. It is desirable that for this region the proof of existence of a total trajectory can be accomplished by finding/guessing locally defined trajectories in a neighborhood of each time moment independently, or at least so that when choosing a local trajectory in a neighborhood of a time moment t one does not need to care about a choice of a local trajectory in a neighborhood of a distant time moment.

We formalize the described generalized method of proving existence of total trajectories of a NCMS as follows.

Let $\Sigma = (T, Q, Tr)$ be a fixed NCMS.

Definition 23. *Σ satisfies*

(1) local forward extensibility (LFE) property, if for each $s \in Tr$ of the form $s : [a, b] \to Q$ $(a < b)$ there exists a trajectory $s' : [a, b'] \to Q$ such that $s' \in Tr$, $s \sqsubseteq s'$ and $b' > b$.

(2) global forward extensibility (GFE) property, if for each trajectory s of the form $s : [a, b] \to Q$ there exists a trajectory $s' : [a, +\infty) \to Q$ such that $s \sqsubseteq s'$.

Theorem 5. *Let (l, r) be a LR representation of Σ. Then Σ has a total trajectory iff there exists a pair $(l', r') \in LR(Q)$ such that*

(1) $l'(s, t) \Rightarrow l(s, t)$ and $r'(s, t) \Rightarrow r(s, t)$ for all $(s, t) \in ST(Q)$;
(2) $\forall t \in [0, \epsilon]$ $l'(s, t) \wedge r'(s, t)$ for some $\epsilon > 0$ and a function $s : [0, \epsilon] \to Q$;
(3) if (l', r') is a LR representation of a NCMS Σ', then Σ' satisfies GFE.

Proof (Sketch). Let us prove the "If" part. Assume that (1)-(3) hold. By (2) there exists $\epsilon > 0$ and $s : [0, \epsilon] \to Q$ such that $l'(s, t) \wedge r'(s, t)$ for all $t \in [0, \epsilon]$. Let $\Sigma' = (T, Q, Tr')$ be a NCMS such that (l', r') is a LR representation of Σ' (which exists, because $(l', r') \in LR(Q)$). Then by (3), Σ' satisfies GFE. Besides, $s \in Tr'$. Then there exists $s' : [0, +\infty) \to Q$ such that $s' \in Tr'$ and $s \sqsubseteq s'$. Then $l'(s, t) \wedge r'(s, t)$ for all $t \in T$, whence $s' \in Tr$, because of (1), so Σ has a total trajectory.

Now let us prove the "Only if" part. Assume that Σ has a total trajectory $s^* \in Tr$. Let $l' : ST(Q) \to Bool$ and $r' : ST(Q) \to Bool$ be predicates such that

$$l'(s, t) \Leftrightarrow l(s, t) \wedge (\min dom(s) \downarrow = t \vee s(t) = s^*(t)).$$

$$r'(s, t) \Leftrightarrow r(s, t) \wedge (\max dom(s) \downarrow = t \vee s(t) = s^*(t)).$$

It is easy to check that $(l', r') \in LR(Q)$ and (l', r') satisfies (1)-(3). □

Theorem 5 means that existence of a total trajectory of a NCMS Σ with LR representation (l, r) can be proved using the following approach:

(1) Choose/guess a pair $(l', r') \in LR(Q)$, where $l'(s, t) \Rightarrow l(s, t)$ and $r'(s, t) \Rightarrow r(s, t)$ for all $(s, t) \in ST(Q)$.
By Definition 15 and Theorem 1, this pair is a LR representation of the NCMS $\Sigma' = (T, Q, Tr')$, where

$$Tr' = \{s : A \to Q \mid A \in \mathfrak{T} \wedge (\forall t \in A \; l'(s, t) \wedge r'(s, t))\} \subseteq Tr.$$

The set $Tr' \subseteq Tr$ plays the role of a region which presumably contains a total trajectory.

(2) If it is possible to find a function s on a small segment $[0, \epsilon]$ which satisfies $l'(s, t) \wedge r'(s, t)$ for $t \in [0, \epsilon]$ (i.e. s is a trajectory of Σ') and prove that Σ' satisfies GFE, then Σ has a total trajectory.

To complete this method of proving existence of a total trajectory, in the next section we will show that the GFE property of a NCMS can be proven by proving existence of certain locally defined trajectories independently in a neighborhood of each time moment.

4.3 Reduction of the GFE Property to the LFE Property

As above, let $\Sigma = (T, Q, Tr)$ be a fixed NCMS.

Definition 24 ([27]). *A right dead-end path (in Σ) is a trajectory $s : [a, b) \to Q$, where $a, b \in T$, $a < b$, such that there is no $s' : [a, b] \to Q$, $s \in Tr$ such that $s \sqsubset s'$ (i.e. s cannot be extended to a trajectory on $[a, b]$).*

Definition 25 ([27]). *An escape from a right dead-end path $s : [a, b) \to Q$ (in Σ) is a trajectory $s' : [c, d) \to Q$ (where $d \in T \cup \{+\infty\}$) or $s' : [c, d] \to Q$ (where $d \in T$) such that $c \in (a, b)$, $d > b$, and $s(c) = s'(c)$. An escape s' is called infinite, if $d = +\infty$.*

Definition 26 ([27]). *A right dead-end path $s : [a, b) \to Q$ in Σ is called strongly escapable, if there exists an infinite escape from s.*

Lemma 13. *If $s : [a, b) \to Q$ is a right dead-end path and $c \in (a, b)$, then $s|_{[c,b)}$ is a right dead-end path.*

The proof follows immediately from the CPR and Markovian properties of Σ.

Lemma 14. *Σ satisfies GFE iff Σ satisfies LFE and each right dead-end path is strongly escapable.*

The proof is analogous to the proof of Lemma 3 in [27] and is omitted here.

Definition 27

(1) A right extensibility measure is a function $f^+ : \mathbb{R} \times \mathbb{R} \tilde{\to} \mathbb{R}$ such that $A = \{(x, y) \in T \times T \mid x \leq y\} \subseteq dom(f^+)$, $f(x, y) \geq 0$ for all $(x, y) \in A$, $f^+|_A$ is strictly decreasing in the first argument and strictly increasing in the second argument, and for each $x \geq 0$, $f^+(x, x) = x$ and $\lim_{y\to+\infty} f^+(x, y) = +\infty$.

(2) A right extensibility measure f^+ is called normal, if f^+ is continuous on $\{(x, y) \in T \times T \mid x \leq y\}$ and there exists a function α of class K_∞ (i.e. the function $\alpha : [0, +\infty) \to [0, +\infty)$ is continuous, strictly increasing, and $\lim_{x\to+\infty} \alpha(x) = +\infty$, $\alpha(0) = 0$) such that $\alpha(y) < y$ for all $y > 0$ and the function $y \mapsto f^+(\alpha(y), y)$ is of class K_∞.

Let us fix a right extensibility measure f^+.

Definition 28. *A right dead-end path $s : [a, b) \to Q$ is called f^+-escapable (Fig. 3), if there exists an escape $s' : [c, d] \to Q$ from s such that $d \geq f^+(c, b)$.*

An example of a right extensibility measure is $f^+(x, y) = 2y - x$ $(x \leq y)$. In this case, for a right dead-end path to be f^+-escapable it is necessary that there exists an escape $s' : [c, d] \to Q$ with $d - b \geq b - c$.

Theorem 6. *Assume that f^+ is a normal right extensibility measure and Σ satisfies LFE. Then each right dead-end path is strongly escapable iff each right dead-end path is f^+-escapable.*

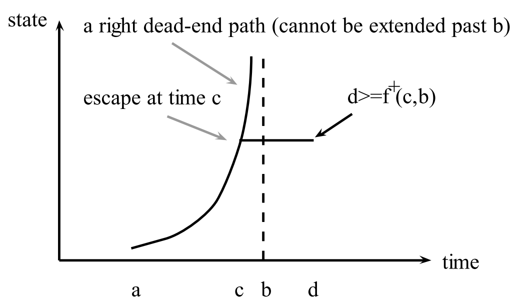


Fig. 3. An f^+-escapable right dead-end path $s : [a, b) \to Q$ (shown as a curve) and a corresponding escape $s' : [c, d] \to Q$ (shown as a horizonal segment) with $d \geq f^+(c, b)$

Proof (Sketch). The statement of this theorem is similar to the statement of [27, Theorem 2] with the difference that here it is assumed that Σ satisfies LFE instead of a stronger condition called weak local extensibility (WLE) [27] (which is used in the proof of [27, Lemma 15]) and the right extensibility measure is assumed to be normal. However, it is straightforward to check that the proof given in [27] is valid for the statement formulated here. □

Theorem 7 (A criterion for the GFE property). *Let (l, r) be an LR representation of a NCMS Σ and f^+ be a normal right extensibility measure. Then Σ satisfies GFE iff for each $t > 0$ there exists $\epsilon \in (0, t]$ such that for each $t_0 \in [t - \epsilon, t)$ and $s : [t_0, t] \to Q$:*

(1) $(\forall \tau \in [t_0, t]\ l(s, \tau) \wedge r(s, \tau)) \Rightarrow \exists t_1 > t$
$\exists s' : [t, t_1] \to Q\ s'(t) = s(t) \wedge (\tau \in dom(s')\ l(s', \tau) \wedge r(s', \tau))$;
(2) $(\forall \tau \in [t_0, t)\ l(s, \tau) \wedge r(s, \tau)) \wedge \neg l(s, t) \Rightarrow \exists t_1 \in (t_0, t)$
$\exists s' : [t_1, f^+(t_1, t)] \to Q\ s'(t_1) = s(t_1) \wedge (\tau \in dom(s')\ l(s', \tau) \wedge r(s', \tau))$.

Proof. Let us prove the "If" part. Assume that for each $t > 0$ there exists $\epsilon \in (0, t]$ such that (1) and (2) hold for each $t_0 \in [t - \epsilon, t)$ and $s : [t_0, t] \to Q$.

Firstly, let us show that Σ satisfies LFE. Let $\bar{s} : [a, b] \to Q$ be a trajectory of Σ. Then $b > a \geq 0$. Then for $t = b$ there exists $\epsilon \in (0, t]$ such that (1) holds for each $t_0 \in [t - \epsilon, t)$ and $s : [t_0, t] \to Q$. Let $t_0 = \max\{a, t - \epsilon\}$ and $s = \bar{s}|_{[t_0,t]}$. Then $s \in Tr$ by the CPR property and $l(s, \tau) \wedge r(s, \tau)$ for all $\tau \in [t_0, t]$, and by (1) there exists $t_1 > t = b$ and $s' : [t, t_1] \to Q$ such that $s'(t) = s(t) = \bar{s}(t)$ and $l(s', \tau) \wedge r(s', \tau)$ for all $\tau \in dom(s')$. Then $s' \in Tr$. Let us define $s'' : [a, t_1] \to Q$ as follows: $s''(\tau) = \bar{s}(\tau)$, if $\tau \in [a, b]$ and $s''(\tau) = s'(\tau)$, if $\tau \in [b, t_1]$. Then $s'' \in Tr$ by the Markovian property. Also, $\bar{s} \sqsubseteq s''$ and $t_1 > b$. So Σ satisfies LFE.

Secondly, let us show that each right dead-end path in Σ is f^+-escapable. Let $\bar{s} : [a, b) \to Q$ be a right dead-end path in Σ . Then $b > 0$. Then for $t = b$ there exists $\epsilon \in (0, t]$ such that (2) holds for each $t_0 \in [t - \epsilon, t)$ and $s : [t_0, t] \to Q$. Let $t_0 = \max\{a, t - \epsilon\}$ and s be some continuation of $\bar{s}|_{[t_0,t)}$ on $[t_0, t]$. Then $s|_{[t_0,t)} \in Tr$ by the CPR property and $l(s, \tau) \wedge r(s, \tau)$ for all $\tau \in [t_0, t)$. Besides,

$\neg l(s,t)$, because $\bar{s}$ is a dead-end path and $r(s,t)$ holds. Then by (2) there exists $t_1 \in (t_0,t)$ and $s' : [t_1, f^+(t_1,t)] \to Q$ such that $s'(t_1) = s(t_1)$ and $l(s',\tau) \wedge r(s',\tau)$ for all $\tau \in dom(s')$. Then $s' \in Tr$. Moreover, $t_1 \in (a,b)$, $s'(t_1) = s(t_1) = \bar{s}(t_1)$, and $\max dom(s') \geq f^+(t_1,b)$. Thus s' is an escape from $\bar{s}$ and $\bar{s}$ is f^+-escapable.

Now by Theorem 6, each right dead-end path in Σ is strongly escapable. Then by Lemma 14, Σ satisfies GFE.

Now let us prove the "Only if" part. Assume that Σ satisfies GFE. Let $t > 0$. Let us choose an arbitrary $\epsilon \in (0,t]$. Assume that $t_0 \in [t-\epsilon, t)$ and $s : [t_0,t] \to Q$.

Let us show (1). Assume that $l(s,\tau) \wedge r(s,\tau)$ for all $\tau \in [t_0,t]$. Then $s \in Tr$ and by GFE there exists $s_1 : [t_0,+\infty] \to Q$ such that $s_1 \in Tr$ and $s \sqsubseteq s_1$. Let $t_1 = t+1$ and $s' = s|_{[t,t_1]}$. Then $s' \in Tr$ by the CPR property and $s'(t) = s(t)$ and $l(s',\tau) \wedge r(s',\tau)$ for all $\tau \in dom(s')$.

Let us show (2). Assume that $l(s,\tau) \wedge r(s,\tau)$ for all $\tau \in [t_0,t)$. Then $s|_{[t_0,t)} \in Tr$. Firstly, consider the case when $s|_{[t_0,t)}$ is a right dead-end path in Σ. Then by Lemma 14 it is strongly escapable, so there exists $t_1 \in (t_0,t)$ and $s_1 : [t_1,+\infty) \to Q$ such that $s_1(t_1) = s(t)$ and $s_1 \in Tr$. Let $s' = s_1|_{[t_1,f^+(t_1,t)]}$. Then $s' \in Tr$ by the CPR property and $s'(t_1) = s(t_1)$ and $l(s',\tau) \wedge r(s',\tau)$ for all $\tau \in dom(s')$.

Now assume that $s|_{[t_0,t)}$ is not a right dead-end path. Then there exists $s_0 : [t_0,t] \to Q$ such that $s_0 \in Tr$ and $s|_{[t_0,t)} \sqsubseteq s_0$. Then by GFE there exists $s_1 : [t_0,+\infty] \to Q$ such that $s_1 \in Tr$ and $s_0 \sqsubseteq s_1$. Let us choose any $t_1 \in (t_0,t)$ and define $s' = s_1|_{[t_1,f^+(t_1,t)]}$. Then $s' \in Tr$ by the CPR property, and $s'(t_1) = s_1(t_1) = s_0(t_1) = s(t_1)$ and $l(s',\tau) \wedge r(s',\tau)$ for all $\tau \in dom(s')$. $\square$

This theorem means that to prove the GFE property, it is sufficient to prove the existence of certain locally defined trajectories in a neighborhood of each $t \in T$.

5 Conclusion

We have considered the questions of the existence of total I/O pairs of a given strongly nonanticipative block and the existence of a total output signal bunch for a given total input signal bunch. We have reduced them to the problem of the existence of total trajectories of NCMS using a NCMS representation. For the latter problem we have proposed a criterion which can be reduced to the problem of checking the existence of certain locally defined trajectories.

References

1. Baheti, R., Gill, H.: Cyber-physical systems. The Impact of Control Technology, 161–166 (2011)
2. Lee, E.A., Seshia, S.A.: Introduction to embedded systems: A cyber-physical systems approach (2013), `Lulu.com`
3. Shi, J., Wan, J., Yan, H., Suo, H.: A survey of cyber-physical systems. In: 2011 International Conference on Wireless Communications and Signal Processing (WCSP), pp. 1–6. IEEE (2011)
4. Lee, E.A.: Computing needs time. Communications of the ACM 52, 70–79 (2009)
5. Oppenheim, A.V., Willsky, A.S., Nawab, S.H.: Signals and systems. Prentice-Hall (1983)

6. Levine, W.S.: The control handbook. CRC Press (1996)
7. Zadeh, L.A., Desoer, C.A.: Linear System Theory: The State Space Approach. McGraw-Hill (1963)
8. Zadeh, L.A.: The concepts of system, aggregate, and state in system theory (1969)
9. Kalman, R.E., Falb, P.L., Arbib, M.A.: Topics in Mathematical System Theory (Pure & Applied Mathematics S.). McGraw-Hill Education (1969)
10. Padulo, L., Arbib, M.: System theory: a unified state-space approach to continuous and discrete systems. W.B. Saunders Company (1974)
11. Klir, G.J.: Facets of Systems Science (IFSR International Series on Systems Science and Engineering). Springer (2001)
12. Wymore, A.W.: A mathematical theory of systems engineering: the elements. Wiley (1967)
13. Mesarovic, M.D., Takahara, Y.: Abstract Systems Theory. Lecture Notes in Control and Information Sciences. Springer (1989)
14. Zeigler, B.P., Praehofer, H., Kim, T.G.: Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems. Academic Press (2000)
15. Matrosov, V.M., Anapolskiy, L., Vasilyev, S.: The method of comparison in mathematical systems theory. Nauka, Novosibirsk (1980) (in Russian)
16. Willems, J.C.: Paradigms and puzzles in the theory of dynamical systems. IEEE Transactions on Automatic Control 36, 259–294 (1991)
17. Polderman, J.W., Willems, J.C.: Introduction to mathematical systems theory: a behavioral approach. Springer, Berlin (1997)
18. Lin, Y.: General systems theory: A mathematical approach. Springer (1999)
19. Seising, R.: Cybernetics, system(s) theory, information theory and fuzzy sets and systems in the 1950s and 1960s. Information Sciences 180, 4459–4476 (2010)
20. Goebel, R., Sanfelice, R.G., Teel, A.: Hybrid dynamical systems. IEEE Control Systems 29, 28–93 (2009)
21. Ball, J.: Finite time blow-up in nonlinear problems. Nonlinear Evolution Equations, pp. 189–205 (1978)
22. Levine, H.A.: The role of critical exponents in blowup theorems. Siam Review 32, 262–288 (1990)
23. Goriely, A.: Integrability and nonintegrability of dynamical systems, vol. 19. World Scientific Publishing Company (2001)
24. Zhang, J., Johansson, K.H., Lygeros, J., Sastry, S.: Zeno hybrid systems. International Journal of Robust and Nonlinear Control 11, 435–451 (2001)
25. Heymann, M., Lin, F., Meyer, G., Resmerita, S.: Analysis of Zeno behaviors in a class of hybrid systems. IEEE Trans. on Automatic Control 50, 376–383 (2005)
26. Ivanov, Ie.: An abstract block formalism for engineering systems. In: Ermolayev, V., Mayr, H.C., Nikitchenko, M., Spivakovsky, A., Zholtkevych, G., Zavileysky, M., Kravtsov, H., Kobets, V., Peschanenko, V.S. (eds.) ICTERI. CEUR Workshop Proceedings, vol. 1000, pp. 448–463. CEUR-WS.org (2013)
27. Ivanov, Ie.: A criterion for existence of global-in-time trajectories of nondeterministic Markovian systems. In: Ermolayev, V., Mayr, H.C., Nikitchenko, M., Spivakovsky, A., Zholtkevych, G. (eds.) ICTERI 2012. CCIS, vol. 347, pp. 111–130. Springer, Heidelberg (2013)
28. Nikitchenko, N.S.: A composition nominative approach to program semantics. Technical report, IT-TR 1998-020, Technical University of Denmark (1998)
29. Hájek, O.: Theory of processes, i. Czechoslovak Mathematical Journal 17, 159–199 (1967)

Specializations in Symbolic Verification

Vladimir Peschanenko[1], Anton Guba[2], and Constantin Shushpanov[3]

[1] Kherson State University, 27, 40 rokiv Zhovtnya str., Kherson, 73000 Ukraine
vladimirius@gmail.com
[2] Glushkov Institute of Cybernetics of NAS of Ukraine, 40, Glushkova ave., Kyiv, 03680
antonguba@ukr.net
[3] LLC «Information Software Systems», 15, Bozhenko str., Kyiv, 03680 Ukraine
costa@iss.org.ua

Abstract. We present the technique that allows splitting first-order logic formulae into parts which helps to use the special algorithms of satisfiability checking and predicate transformer, which are the specializations. We describe the mathematical description of the algorithm of the constructing specializations. We prove the correctness of satisfiability and predicate transformer functions. We consider forward and backward applicability of basic protocols during symbolic modeling and verification. We introduce the examples for each specialization. We provide the experiments with typical real examples.

Keywords: Symbolic modeling, satisfiability, predicate transformer.

1 Introduction

The technique of symbolic verification of requirement specifications of software systems has shown good results in automatic detection of reachability of deadlocks and violation of user-defined properties [1]. In previous works [2-4] symbolic models of systems being transition systems with symbolic states represented by formulae of first order logic were considered. A relation of transitions between the formulae is determined and marked by basic protocols, which are considered as actions, performed in the system. A basic protocol is a formula of dynamic logic $\forall x(\alpha(x,a) \rightarrow < P(x,a) > \beta(x,a))$ and it describes local properties of the system in terms of pre- and postconditions α and β. Both are formulae of first order multisorted logic interpreted on a data domain, P is a process, represented by means of MSC diagram and describes the reaction of a system triggered by the precondition, x is a set of typed data variables, and a is a set of environment attributes. The general theory of basic protocols is presented in [5].

A transition is considered as an operator in the space of postcondition formulae. As the operator transforms one formula to another, in [6] a term "predicate transformer" is used. Thus, to compute transitions between the states of such models basic protocols are interpreted as predicate transformers: for given symbolic state of the system and given basic protocol the direct predicate transformer generates the next symbolic state as its strongest postcondition, and the backward predicate transformer generates the previous symbolic state as its weakest precondition. These concepts have been implemented in VRS (Verification of Requirement Specifications) system [7] and IMS (Insertion Modeling System) system [8].

An number of papers with novel and very efficient techniques for computing satisfiability using SAT/SMT has been published in the last years, and some very efficient SMT tools are now available (e.g., BarceLogic [9], CVCLite/CVC/CVC4 [10,11,12], DLSAT [13], haRVey [14], MathSAT [15], SDSAT [16], TSAT++ [17], UCLID [18], Yices [19], Verifun [20], Zapato [21], Z3 [22]). A number of benchmarks, mostly derived from verification problems, is available at the SMT-LIB [23]. Workshops devoted to SMT and official competitions on SMT tools are run yearly.

All these tools could be configured with the help of many parameters, which means the usage of some techniques, tactics, heuristics, in order to gain in performance. In the paper [24] the algorithm configuration problem is stated as follows: given an algorithm, a set of parameters for the algorithm, and a set of input data, found parameter values under which the algorithm achieves the best possible performance on the input data. It gives a possibility of automated tuning of algorithm for obtaining performance on formulae of some theory.

Usually during modeling of real projects we deal with complex environment states and simple formulae of basic protocols (pre- and postconditions). It means that we should check the satisfiability of the conjunction of the environment state and the precondition formula and transform this whole big formula with the help of predicate transformer [6]. Obviously, the manipulation with whole formulae is not required for most of cases.

For example, let $i, j : \mathrm{int}, f : \mathrm{int} \to \mathrm{int}$ be attributes and $f(i) > 0 \wedge f(0) < 5 \wedge$ $\wedge j > 0$ be an environment state, and $1 \to j := j + 1$ be a basic protocol. Let us apply this basic protocol to the environment state. First, the satisfiability of conjunction of basic protocol precondition and environment state should be checked: $f(i) > 0 \wedge f(0) < 5 \wedge j > 0$. This checking should use the notion of functional symbols: $(i = 0) \to (f(i) = f(0))$. After that we should apply basic protocol postcondition to conjunction of environment state and precondition (see section Application of Basic Protocol):

$$\exists (v : \mathrm{int})(f(i) > 0 \wedge f(0) < 5 \wedge v > 0 \wedge (j = v + 1)) \Rightarrow$$
$$\Rightarrow f(i) > 0 \wedge f(0) < 5 \wedge \exists (v : \mathrm{int})(v > 0 \wedge (j = v + 1)) \Rightarrow$$
$$\Rightarrow f(i) > 0 \wedge f(0) < 5 \wedge j > 1$$

It is known that basic protocol changes attribute *j* only (see section about predicate transformers). It means that we could apply basic protocol to small part of environment state that depends on *j*, but not to whole environment state formula. In this example it could be $j > 0$ only. If there are no predicates in projects, which could compare values of attribute *j* with values of other attributes, then we could use some special theories for manipulating with such formulae. In this example numeral intervals could be used for representation of values of attribute *j*. We call such special theories "*Specialization of sat, pt functions*" according to our general algorithm.

So, the main goal of this paper is to present a mathematical description of algorithm of constructing specializations and a few particular approaches to specialization which speed up modeling of industrial models. This paper is a continuation of the [25], where only concrete values as a kind of specialization were described.

In Section 2 we describe the process of forward application of a basic protocol with the help of the satisfiability and the forward predicate transformer. In Section 3 we present an applicability of basic protocols using satisfiability and backward predicate transformer. The specializations by memory usage and functional symbols are proposed in Section 4. The results of experiments are discussed in Section 5. The implementation of specialization support is described in Section 6. In Section 7 we summarize advantages of usage of the specializations and what could be done in the nearest future.

2 Forward Application of Basic Protocol

Let *S(a)* be an environment state, $\forall x(\alpha(x,a) \rightarrow < P(x,a) > \beta(x,a))$ be a basic protocol, where x – parameters of basic protocol, a – attributes of model, $D(x,a) = S(a) \wedge \alpha(x,a)$ – conjunction of environment state and precondition of basic protocol.

At the first step of application of basic protocol satisfiability of conjunction of environment state and precondition of basic protocol is checked: $sat(D(x,a))$. If the formula is unsatisfiable, then basic protocol is not applicable to environment state *S(a)*. If not, then process $P(x,a)$ is run and after forward predicate transformer is applied: $pt(D(x,a), \beta(x,a))$. The process of $P(x,a)$ is not considered in the paper, because the specialization tries to speed up the functions *sat* and *pt*.

2.1 Satisfiability

The function of checking formula satisfiability *sat* is based on the Shostak method, adapted to combination of numerical, symbolic and enumerated data types. If all of the attribute expressions (simple attributes and functional symbols with parameters) that are free in the formula *S* are simple, then for satisfiability checking it is sufficient to prove validity of the closed formula $\exists(a,x)D(x,a)$, where a is a set of all simple attributes which occur in *S*, *x* is a set of parameters of basic protocol. For attribute expressions with parameters (including access functions to the elements of arrays), the Ackermann reduction of the uninterpreted functional symbols is used, where attribute expression is an attribute or functional symbol with parameters.

The Shostak method consists of the following. An expression of the form $f(x)$ is called as *Functional Expression*, if *f* is an attribute and *x* is a list of its parameters. At first, superpositions of functional expressions are eliminated by successive substitution of every internal occurrence of $f(x)$ by a new variable y, bounded by existential quantifier and added to the formula $y = f(x)$. For example, formula $P(f(g(x)))$ is replaced by formula $\exists y(y = g(x) \wedge P(f(y)))$. After all such replacements there will not be complex functional expressions in the formula. Further, for every attribute expression *f* of functional type all its occurrences $f(x_1),...,f(x_n)$ with the different parameters $x_1,...,x_n$ are considered. Occurrence $f(x_i)$ is replaced by variable y_i, bounded by existential quantifier and substitutive equations $(x_i = x_j) \rightarrow (y_i = y_j)$ are added. Now in the formula there are only simple attributes, and a method considered in [26] is used.

2.2 Forward Predicate Transformer

In general case, the post-condition looks like $\beta(x,a) = R(x,a) \wedge C(x,a)$, where $R = (r_1 := t_1 \wedge r_2 := t_2 \wedge ...)$ is a conjunction of assignments and *C(x,a)* is a formula part of post-condition.

We will consider three sets of functional expressions (we consider attributes as a functional expression with 0 arity): *r*, *s* and *z*. Set $r = \{r_1, r_2,...\}$ consists of the left parts of assignment, and also of other functional expressions that recursively depend on the left parts. In other words, *r* consists of the left parts of assignments and, if some functional expression *f* is included into this set, then all functional expressions in which *f* occurs are also included in *r*. Set $s = \{s_1, s_2,...\}$ consists of functional expressions which have external occurrences (not in arguments of such functional expressions) in formula part *C* of post-condition, but do not coincide with expressions from the set *r*. Finally, set $z = \{z_1, z_2,...\}$ consists of functional expressions which have external occurrences in formula *D* in right parts of assignments and in internal occurrences (in arguments if functional expressions) of the functional expressions of formula part *C* of post-condition and left parts of assignments, but these assignments are not included in two other sets (including parameter of basic protocol). Now, considering formulae, from which a post-condition and formula *D* are constructed as functions of external occurrences of elements of these sets, we get a presentation of post-condition in the following form:

$$B(r,s,z) = (r_1(r,s,z) := t_1(r,s,z) \wedge r_2(r,s,z) := t_2(r,s,z) \wedge ...) \wedge C(r,s,z),$$

Predicate transformer is determined by the following formula:

$$\text{pt}(D(r,s,z), \beta(r,s,z)) = q_1 \vee q_2 \vee ..., \text{where}$$
$$q_i = \exists(u,v)(D(u,v,z) \wedge R(u,v,z) \wedge E_i(u,v,z) \wedge C(r,s,z)),$$
$$R(u,v,z) = (r_1(u,v,z) = t_1(u,v,z)) \wedge (r_2(u,v,z) = t_2(u,v,z)) \wedge ...),$$

Formula $R(u,v,z)$ is a quantifier-free part of the assignment formula. Sets of the variables *u,v* represents new variables for each attribute expression from *r,s* sets correspondently. The *pt* substitutes attributes from *r,s* set to variables from *u,v* sets correspondently in corresponded part of formula.

Each of disjunctive members q_i corresponds to one of possible means of identification of functional expressions occurring in formulae $\beta(x,a)$, and $E_i(u,v,z)$ is a set of equalities and inequalities corresponding to such identification.

To describe the construction of $E_i(u,v,z)$ we will consider the set *M* of all pairs of functional expressions in the form $(f(k),f(l)), k=(k_1,k_2,...), l=(l_1,l_2,...)$, where $f(k)$ is chosen from set *z*, and $f(l)$ – from sets *r* and *s*. These functional expressions shall be equal if their arguments were equal before application of basic protocol.

Let us choose arbitrary subset $N \subseteq M$ (including an empty set for every pair $(f(k),f(l)) \in N$ we will consider conjunction of equalities $k=l,(k_1=l_1 \wedge k_2=l_2 \wedge ...)$. We will unite all such conjunctions in one and will add to it conjunctive negations of all equalities, which correspond to pairs which are not included into the set *N*. We will denote the obtained formula as $G_i(r,s,z)$. If this formula is satisfiable, then the choice is successful. Now obviously, $f(k)$ is not independent and shall change the value because $G_i(r,s,z)$ is true. Thus, $f(k)$ shall change the value in the same way as $f(l)$. $E_i(r,s,z)=G_i(r,s,z) \wedge H_i(z,u,v)$ where $H_i(z,u,v)$ is a conjunction of equalities $f(k)=w$ if a variable *w* corresponds to $f(l)$. Thus, if $f(k)$ coincides with several functional expressions, it is not important what variable is chosen (transitivity of equality) [6].

3 Backward Application of Basic Protocol

Let $S(a)$ be an environment state after the application of the basic protocol $\forall x(\alpha(x,a) \rightarrow < P(x,a) > \beta(x,a))$, where x is parameters of basic protocol, a – attributes of model, $\beta(x,a)=R(x,a) \wedge C(x,a)$, where $R=(r_1:=t_1 \wedge r_2:=t_2 \wedge ...)$ is a conjunction of assignments and C is a formula part of post-condition, $D(x,a)=S(a) \wedge C(x,a)$ is a conjunction of environment state and formula part of postcondition of basic protocol.

3.1 Satisfiability

At first step of application of basic protocol in backward mode satisfiability of conjunction of environment state and formula part of postcondition of basic protocol is checked: $sat(D(x,a))$. If the formula is unsatisfiable, then the basic protocol is not applicable to environment state $S(a)$.If not, then process $P(x,a)$ is run and after a backward predicate transformer is applied: $pt^{-1}(D(x,a),\beta(x,a))$.

3.2 Backward Predicate Transformer

A backward predicate transformer considers three sets of functional expressions *r*, *s* and *z* (as forward too). A postcondition of the basic protocols is represented by the following formula:

$$B(r,s,z) = (r_1(r,s,z) := t_1(r,s,z) \wedge r_2(r,s,z) := t_2(r,s,z) \wedge ...) \wedge C(r,s,z)$$

A backward predicate transformer is determined by the following formula:

$pt^{-1}(D(r,s,z),\beta(r,s,x)) = q_1^{-1} \vee q_2^{-1} \vee ...$, where

$q_i^{-1} = \exists(u,v)(D(u,v,z) \wedge R'(u,r,s,z) \wedge E_i(u,v,z)) \wedge \alpha(r,s,z)$,

$R'(u,r,s,z) = (u_1(r,s,z) = t_1(r,s,z)) \wedge (u_2 = t_2(r,s,z)) \wedge ...), u = \{u_1, u_2, ...\}$,

Each of disjunctive members q_i corresponds to one of possible identification of functional expressions, occurring in formulae $\beta(x,a)$ and environment state *S(a)*, where $E_i(u,v,z)$ are conjunction of equalities and inequalities corresponding to such identification. Formula $E_i(u,v,z)$ is built in the same way as in forward predicate transformer [27].

4 Specialization

We propose to use two types of specializations:

1. Specialization by memory usage
2. Specialization by functional symbol

4.1 Specialization by Memory Usage

Let a_1, a_2 be sets of attributes from initial environment state and $a_1 \cap a_2 = \varnothing \wedge a_1 \cup a_2 = a$, $S(a) = S_1(a_1) \wedge S_2(a_2)$ is environment state, $B(x,a) = \forall x(\alpha_1(x_1,a_1) \wedge \alpha_2(x_2,a_2) \rightarrow< P(x,a) > \beta_1(x_1,a_1) \wedge \beta_2(x_2,a_2))$ is basic protocol, where $x_1 \cap x_2 = \varnothing \wedge x_1 \cup x_2 = x$.

If $B'(x,a) = \forall x(\vee_i \alpha_i(x,a) \rightarrow< P(x,a) > \beta(x,a))$ then $sat(S(a) \wedge (\vee_i \alpha_i(x,a))) = \\ = \vee_i sat(S(a) \wedge \alpha_i(x,a))$ and $pt(S(a) \wedge (\vee_i \alpha_i(x,a)), \beta(x,a)) = \vee_i pt(S(a) \wedge \\ \wedge \alpha_i(x,a), \beta(x,a))$. So, in the next text we consider basic protocol as $B(x,a)$ only.

Theorem 1. If $a_1 \cap a_2 = \varnothing \wedge a_1 \cup a_2 = a$, $S(a) = S_1(a_1) \wedge S_2(a_2)$, and $B(x,a) = \forall x(\alpha_1(x_1,a_1) \wedge \alpha_2(x_2,a_2) \rightarrow< P(x,a) > \beta_1(x_1,a_1) \wedge \beta_2(x_2,a_2))$ then

$$sat(S_1(a_1) \wedge \alpha_1(x_1,a_1) \wedge S_2(a_2) \wedge \alpha_2(x_2,a_2)) =$$
$$= sat(S_1(a_1) \wedge \alpha_1(x_1,a_1)) \wedge sat(S_2(a_2) \wedge \alpha_2(x_2,a_2))$$

Proof
Function *sat* builds closed formula. So,

$$sat(S_1(a_1) \wedge \alpha_1(x_1, a_1) \wedge S_2(a_2) \wedge \alpha_2(x_2, a_2)) =$$
$$= \exists(v_1, v_2, x_1, x_2)(S_1(a_1) \wedge \alpha_1(x_1, a_1) \wedge S_2(a_2) \wedge \alpha_2(x_2, a_2))$$

where v_1, v_2 are variables generated for attribute expression which depend on attributes a_1, a_2. It is known that $a_1 \cap a_2 = \varnothing \wedge a_1 \cup a_2 = a \wedge x_1 \cap x_2 = \varnothing \wedge$ $\wedge x_1 \cup x_2 = x$. It means that scope of quantifiers could be narrowed:

$$\exists(v_1, v_2, x_1, x_2)(S_1(a_1) \wedge \alpha_1(x_1, a_1) \wedge S_2(a_2) \wedge \alpha_2(x_2, a_2)) =$$
$$= \exists(v_1, x_1)(S_1(a_1) \wedge \alpha_1(x_1, a_1)) \wedge \exists(v_2, x_2)(S_2(a_2) \wedge \alpha_2(x_2, a_2)) = \text{i}$$
$$= sat(S_1(a_1) \wedge \alpha_1(x_1, a_1)) \wedge sat(S_2(a_2) \wedge \alpha_2(x_2, a_2))$$

Theorem is proved.
This theorem means the following:

1. If $S(a) = S_1(a_1) \wedge S_2(a_2)$ and $\alpha(a, x) = \alpha_1(x_1, a_1)$ and $S(a)$ is satisfiable, then it is enough to check satisfiability of conjunction of $S_1(a_1) \wedge \alpha_1(x_1, a_1)$ for satisfiability checking of $S(a) \wedge \alpha(x, a)$. Checking of satisfiability of $S_2(a_2)$ is not required.
2. Checking of each part $sat(S_i(a_i) \wedge \alpha_i(x_i, a_i))$ could be done concurrently.

This case could be easily generalized to $a_1, ..., a_n$ case, because if it is possible to build subsets $a_i^1, a_i^2 \in a_i \wedge a_i^1 \cap a_i^2 = \varnothing \wedge a_i^1 \cup a_i^2 = a_i$ and to spilt an environment state and basic protocol accordingly to the *theorem 1,* then $sat(\underset{i}{\wedge} S_i(a_i) \wedge \alpha_i(x_i, a_i)) = \underset{i}{\wedge} sat(S_i(a_i) \wedge \alpha_i(x_i, a_i))$. So, after if we say about such pair of two sets $a_i^1, a_i^2 \in a_i \wedge a_i^1 \cap a_i^2 = \varnothing \wedge a_i^1 \cup a_i^2 = a_i$, then we understand that it could be applicable and for *n* sets.

Let us see how forward and backward predicate transformer can be applied.

Theorem 2. For forward application of basic protocol it is true that: if $a_1 \cap a_2 = \varnothing \wedge a_1 \cup a_2 = a$, $S(a) = S_1(a_1) \wedge S_2(a_2)$, and $B(x, a) = \forall x(\alpha_1(x_1, a_1) \wedge \alpha_2(x_2, a_2) \rightarrow < P(x, a) > \beta_1(x_1, a_1) \wedge \beta_2(x_2, a_2))$ then

$$pt(S_1(a_1) \wedge \alpha_1(x_1, a_1) \wedge S_2(a_2) \wedge \alpha_2(x_2, a_2), \beta_1(x_1, a_1) \wedge \beta_2(x_2, a_2)) =$$
$$= pt(S_1(a_1) \wedge \alpha_1(x_1, a_1), \beta_1(x_1, a_1)) \wedge pt(S_2(a_2) \wedge \alpha_2(x_2, a_2), \beta_2(x_2, a_2))$$

Proof.
pt function builds sets r, s, z from postcondition $\beta_1(x_1, a_1) \wedge \beta_2(x_2, a_2)$ and formula $S_1(a_1) \wedge \alpha_1(x_1, a_1) \wedge S_2(a_2) \wedge \alpha_2(x_2, a_2)$, where r is a set of attribute expressions from left parts of assignments of postcondition, s is a set of attribute expressions from formula part of postcondition, z is a set of other attribute expressions

from formula and postcondition. We know that sets of attribute expressions from pairs $S_1(a_1) \wedge \alpha_1(x_1, a_1)$, $\beta_1(x_1, a_1)$ and $S_2(a_2) \wedge \alpha_2(x_2, a_2)$, $\beta_2(x_2, a_2)$ are not intersected. It means that we could split each set r, s, z on subsets $r = r_1 \cup r_2, s = s_1 \cup s_2, z = z_1 \cup z_2$ and $r_1 \cap r_2 = \varnothing$, $s_1 \cap s_2 = \varnothing$, $z_1 \cap z_2 = \varnothing$, because $a_1 \cap a_2 = \varnothing$. Let us write formula which is built by *pt* function.

Let $D(a,x) = D_1(x_1,a_1) \wedge D_2(x_2,a_2), D_1(x_1,a_1) = S_1(a_1) \wedge \alpha_1(x_1,a_1)$, $D_2 = S_2(a_2) \wedge \alpha_2(x_2,a_2)$ and $\beta_1(x_1,a_1) = R_1(r_1,s_1,z_1) \vee C_1(r_1,s_1,z_1)$, $\beta_2(x_2,a_2) = R_2(r_2,s_2,z_2) \vee C_2(r_2,s_2,z_2)$. So, general formula of predicate transformer is the following:

$$\underset{i}{\vee} q_i = \exists(u,v)(D(u,v,z) \wedge R(u,v,z) \wedge (\underset{i}{\vee} E_i(u,v,z)) \wedge C(r,s,z))$$

where $R(u,v,z) = R_1(u_1,v_1,z_1) \wedge R_2(u_2,v_2,z_2)$, $C(r,s,z) = C_1(r_1,s_1,z_1) \wedge$ $\wedge C_2(r_2,s_2,z_2)$. because $\beta(a,x) = \beta_1(x_1,a_1) \wedge \beta_2(x_2,a_2)$.

Let us write in details how to obtain $\underset{i}{\vee} E_i(u,v,z)$. It is known that $r_1 \cap r_2 = \varnothing$, $s_1 \cap s_2 = \varnothing$, $z_1 \cap z_2 = \varnothing$. To build such disjunction we should take into account all pairs of functional attribute expressions from sets *r,s* and *z*. It means that each such pair should be in set of attribute $(r_1 \cup s_1; z_1)$ or $(r_2 \cup s_2; z_2)$. So,

$$\underset{i}{\vee} E_i(u,v,z) = (\underset{i_1}{\vee} E_{i_1}(u_1,v_1,z_1)) \wedge (\underset{i_2}{\vee} E_{i_2}(u_2,v_2,z_2))$$

Let us consider formula of predicate transformer:

$$\begin{aligned}
& pt(S_1(a_1) \wedge \alpha_1(x_1,a_1) \wedge S_2(a_2) \wedge \alpha_1(x_2,a_2), \beta_1(x_1,a_1) \wedge \beta_2(x_2,a_2)) = \\
& \underset{i}{\vee} q_i = \exists(u,v)(D(u,v,z) \wedge R(u,v,z) \wedge (\underset{i}{\vee} E_i(u,v,z)) \wedge C(r,s,z)) = \\
& = \exists(u_1,u_2,u_1,v_2)(D_1(u_1,v_1,z_1) \wedge D_2(u_1,v_1,z_1) \wedge \\
& \wedge R_i(u_1,v_1,z_1) \wedge R_i(u_2,v_2,z_2) \wedge \\
& \wedge (\underset{i_1}{\vee} E_{i_1}(u_1,v_1,z_1)) \wedge (\underset{i_2}{\vee} E_{i_2}(u_2,v_2,z_2)) \wedge \\
& \wedge C_1(r_1,s_1,z_1) \wedge C_2(r_2,s_2,z_2)) = \\
& = \exists(u_1,v_1)(D_1(u_1,v_1,z_1) \wedge R(u_1,v_1,z_1) \wedge (\underset{i_1}{\vee} E_{i_1}(u_1,v_1,z_1)) \wedge \\
& \wedge C_1(r_1,s_1,z_1)) \wedge \exists(u_2,v_2)(D_2(u_2,v_2,z_2) \wedge R(u_2,v_2,z_2) \wedge \\
& \wedge (\underset{i_2}{\vee} E_{i_2}(u_2,v_2,z_2)) \wedge C_2(r_2,s_2,z_2)) \wedge \ldots = \\
& = pt(D_1(x_1,a_1), \beta_1(x_1,a_1)) \wedge pt(D_2(x_2,a_2), \beta_2(x_2,a_2))
\end{aligned}$$

Theorem is proved.

Theorem 3. For backward mode it is true that:

If $a_1 \cap a_2 = \varnothing \wedge a_1 \cup a_2 = a$, $S(a) = S_1(a_1) \wedge S_2(a_2)$, and $B(x,a) = \forall x(\alpha_1(x_1,a_1) \wedge \alpha_2(x_2,a_2) \rightarrow < P(x,a) > \beta_1(x_1,a_1) \wedge \beta_2(x_2,a_2))$ then

$$pt^{-1}(S_1(a_1) \wedge C_1(r_1,s_1,z_1) \wedge S_2(a_2) \wedge C_2(r_2,s_2,z_2),$$
$$\beta_1(x_1,a_1) \wedge \beta_2(x_2,a_2)) = pt^{-1}(S_1(a_1) \wedge C_1(r_1,s_1,z_1), \beta_1(x_1,a_1)) \wedge$$
$$pt(S_2(a_2) \wedge C_2(r_2,s_2,z_2), \beta_2(x_2,a_2))$$

Proof.

$R(u,v,z) = R(u_1,v_1,z_1) \wedge R(u_2,v_2,z_2)$, $C(r,s,z) = C_1(r_1,s_1,z_1) \wedge C_2(r_2,s_2,z_2)$ because $\beta(r_a,x) = \beta_1(a_1,x_1) \wedge \beta_2(a_2,x_2)$. $\underset{i}{\vee} E_i(u,v,z) = (\underset{i_1}{\vee} E_{i_1}(u_1,v_1,z_1)) \wedge$ $\wedge(\underset{i_2}{\vee} E_{i_2}(u_2,v_2,z_2))$ from previous theorem.

$$pt^{-1}(S(a) \wedge C(r,s,z), \beta(r,s,z)) = \underset{i}{\vee} q_i^{-1} =$$
$$= \underset{i}{\vee} \exists(u,v)(S(a) \wedge C(r,s,z) \wedge R'(u,r,s,z) \wedge E_i(u,v,z)) \wedge \alpha(r,s,z) =$$
$$= \exists(u_1,u_2,v_1,v_2)(S_1(r_1,s_1,z_1) \wedge S_2(r_2,s_2,z_2) \wedge$$
$$\wedge C_1(r_1,s_1,z_1) \wedge C_2(r_2,s_2,z_2) \wedge$$
$$\wedge (\underset{i_1}{\vee} E_{i_1}(u_1,v_1,z_1)) \wedge (\underset{i_2}{\vee} E_{i_2}(u_2,v_2,z_2))) \wedge$$
$$\wedge \alpha_1(r_1,s_1,z_1) \wedge \alpha_2(r_2,s_2,z_2) =$$
$$= \exists(u_1,v_1)(S_1(r_1,s_1,z_1) \wedge C_1(r_1,s_1,z_1) \wedge (\underset{i_1}{\vee} E_{i_1}(u_1,v_1,z_1))) \wedge \alpha_1(r_1,s_1,z_1) \wedge$$
$$\wedge \exists(u_2,v_2)(S_2(r_2,s_2,z_2) \wedge C_2(r_2,s_2,z_2) \wedge (\underset{i_2}{\vee} E_{i_2}(u_2,v_2,z_2))) \wedge \alpha_2(r_2,s_2,z_2) =$$
$$= pt^{-1}(S_1(a_1) \wedge C_1(r_1,s_1,z_1), \beta_1(x_1,a_1)) \wedge pt(S_2(a_2) \wedge C_2(r_2,s_2,z_2), \beta_2(x_2,a_2))$$

Theorem is proved.

Theorem 2 and theorem 3 mean that:

1. Functions *pt, pt*$^{-1}$ could be applied separately and concurrently.
2. If postcondition contains $\beta_1(x_1,a_1)$ only, then

$$pt(S_1(a_1) \wedge \alpha_1(x_1,a_1) \wedge S_2(a_2) \wedge \alpha_2(x_2,a_2), \beta_1(x_1,a_1)) =$$
$$= S_2(a_2) \wedge \alpha_2(x_2,a_2) \wedge pt(S_1(a_1) \wedge \alpha_1(x_1,a_1), \beta_1(x_1,a_1))$$
$$pt^{-1}(S_1(a_1) \wedge C_1(r_1,s_1,z_1) \wedge S_2(a_2) \wedge C_2(r_2,s_2,z_2), \beta_1(x_1,a_1)) =$$
$$= S_2(a_2) \wedge C_2(r_2,s_2,z_2) \wedge pt^{-1}(S_1(a_1) \wedge C_1(r_1,s_1,z_1), \beta_1(x_1,a_1))$$

So, functions $sat(D_i(x_i,a_i)), pt(D_i(x_i,a_i), \beta_i(x_i,a_i))$ are called specialization, because we could use some special theories for implementation of it.

5 Examples of Usage of Specializations

5.1 Examples of Specializations by Memory Usage

Example 1. Concrete values. Let $S(a)=(i=2)\wedge S(a/i)$ be an environment state where $i:\text{int}$ and a/i is a set of all attributes in model except *i*, $b=\forall x((i>0)\rightarrow<>(i:=i+1))$. For application of such basic protocol we should check satisfiability of the next formula: $sat((i=2)\wedge(i>0))=1$, and the postcondition should be applied to $(i=2)$: $\exists v((v=2)\wedge(v>0)\wedge(i=v+1))\Rightarrow(i=3)$. For such examples direct *C*++ translation could be used instead of using some special theories, and it will work much faster because it doesn't require any additional checking, just direct translation into *C*++ code and compilation of it.

Example 2. Let $S(a)=(i<2)\wedge S(a/i)$ be environment state where $i:\text{int}$ and a/i is a set of all attributes in model except *i*, $b=\forall x((i>0)\rightarrow<>(i:=i+1))$. For application of such basic protocol we should check satisfiability of the next formula: $sat((i<2)\wedge(i>0))=1$, and the postcondition should be applied to $(i<2)$: $\exists v((v<2)\wedge(v>0)\wedge(i=v+1))\Rightarrow(i>1)\wedge(i<3)$. For such examples numerical intervals could be used. So, $S(a)=(i\in(-\infty;2))\wedge S(a/i)$, $b=\forall x((i\in(0;+\infty))\rightarrow<>(i:=i+1))$. Satisfiability checking looks like just crossing of two numerical intervals: $i\in(-\infty;2)\cap(0;+\infty)\Rightarrow i\in(0;2)\Rightarrow i\in[1;1]$ for integer. Application of *pt* creates the following formula: $\exists v((v\in[1;1])\wedge(i=v+1))\Rightarrow$ $\Rightarrow i-1\in[1;1]\Rightarrow i\in[2;2]$. This approach will work faster than general satisfiability checking and quantifiers eliminations. Such approach could be used for all numeric and enumerated types.

5.2 Examples of Specializations by Functional Symbol

It is not always possible to represent environment state and basic protocols in the following way: $S(a)=S_1(a_1)\wedge S_2(a_2)$, and $B(a,x)=\forall x(\alpha_1(x_1,a_1)\wedge$ $\wedge\alpha_2(x_2,a_2)\rightarrow<P(x,a)>\beta_1(x_1,a_1)\wedge\beta_2(x_2,a_2))$ where $a_1\cap a_2=\varnothing$, $a_1\cup a_2=a$, $x_1\cap x_2=\varnothing\wedge x_1\cup x_2=x$. One of such situation occurs when a value of functional attribute expression and its parameter has different types and belongs to the different subsets a_i. For example, if functional attribute: $i,j:\text{int},f:\text{int}\rightarrow T$ is defined where $T\in(c_1,c_2,c_3)$ is enumerated type with three enumerated constants: c_1,c_2,c_3, then formula $(f(i)=c_1)\wedge i>0$ could be represented with specializations as follows: $(f:v_1=f(i))\wedge(v_1=c_1)\wedge(i>0)$. Let $b=1\rightarrow<>(f(j):=c_2)$ be a basic protocol. Its specialized representation is: $b=1\rightarrow<>(f:v_1=f(j))\wedge(v_1:=c_2)\wedge 1$. It is required to merge such data structures for *pt* function which should consider all pairs of functional attribute expression from sets *r,s* and *z*:

$(f : v_1 = f(i)) \wedge (f : v_1 = f(j)) \Rightarrow (f : v_1 = f(i), v_2 = f(j))$. After that basic protocol should be transformed in the following form: $b = 1 \rightarrow <> (f : v_2 = f(j)) \wedge (v_2 := c_2) \wedge 1$. It is required to take into account two possible combinations: $(i = j) \vee \neg(i = j)$. So, we obtain:

$$\begin{aligned}
& pt((f : v_1 = f(i), v_2 = f(j)) \wedge (v_1 = c_1) \wedge i > 0, v_2 := c_2) = \\
& = (f : v_1 = f(i), v_2 = f(j)) \wedge \\
& \wedge \exists v((i = j) \wedge (v = c_1) \wedge (v_2 = c_2)) \wedge \\
& \wedge \exists v(\neg(i = j) \wedge (v_1 = c_1) \wedge (v_2 = c_2)) \wedge i > 0 \Rightarrow \\
& \Rightarrow (f : v_1 = f(i), v_2 = f(j)) \wedge (v_2 = c_2) \wedge i > 0 \wedge (i = j) \vee \\
& \vee (f : v_1 = f(i), v_2 = f(j)) \wedge (v_1 = c_1) \wedge (v_2 = c_2) \wedge i > 0 \wedge \neg(i = j) \Rightarrow \\
& \Rightarrow (f : v_1 = f(i)) \wedge (v_1 = c_2) \wedge i > 0 \wedge (i = j) \vee \\
& \vee (f : v_1 = f(i), v_2 = f(j)) \wedge (v_1 = c_1) \wedge (v_2 = c_2) \wedge i > 0 \wedge \neg(i = j)
\end{aligned}$$

Let $S(a) = F(f_1, f_2, ..., v_1, v_2, a_1, a_2) \wedge S_1(a_1) \wedge S_2(a_2)$ be an environment state where $f_1 \neq f_2 \neq ...$ are names of functional symbols, v_1, v_2 are variables for each functional attribute expression from sets a_1, a_2 correspondently, and

$$\begin{aligned}
& F(f_1, f_2, ..., v_1, v_2, a_1, a_2) = \\
& = (f_1 : v_1^1 = f_1(t_1^1, t_1^2, ...), v_1^2 = f_1(t_1^1, t_1^2, ...), ..., \\
& f_2 : v_2^1 = f_2(t_2^1, t_2^2, ...), v_2^2 = f_2^2(t_2^1, t_2^2, ...), ..., ...)
\end{aligned}$$

where $v_1^1, v_1^2 \in a_{f_1}, v_2^1, v_2^2 \in a_{f_2}, ...$ are variables of type of functional names $f_1, f_2, ...$ for each attribute expression, a_{fi} is set of attribute, such as $f_i \in a_j$, $t_i^j \in a_i^i \in \{a_i\}$, … - corresponded arguments for each functional with the same name are in one specialization, and Shostak's method could be applied for each right part of equation in *F*.

Let $S(a) = F(f_1, f_2, ..., v_1, v_2, a_1, a_2) \wedge S_1(a_1) \wedge S_2(a_2)$. and

$$\begin{aligned}
& b(a) = \forall x(F_b(f_1, f_2, ..., v_1, v_2, a_1, a_2, x_1, x_2) \wedge \alpha_1(v_1, x_1, a_1) \wedge \\
& \wedge \alpha_2(v_2, x_2, a_2) \rightarrow < P(a, x) > \beta_1(v_1.x_1, a_1) \wedge \beta_2(v_2, x_2, a_2))
\end{aligned}$$

Theorem 4

$$\begin{aligned}
& sat(S(a) \wedge \alpha(x, a)) = sat(\bigwedge_{(i,k,l)} ((f_i(t_i^1, t_i^2, ...) = f_i(t_i'^1, t_i'^2, ...)) \rightarrow (v_i'^k = v_i'^l)) \wedge \\
& \wedge S_1(v_1', a_1) \wedge \alpha_1(v_1', x_1, a_1) \wedge S_2(v_2', a_2) \wedge \alpha_2(v_2', x_2, a_2)) = \\
& = \bigvee_i sat(q_i \wedge S_i(v_i', a_i) \wedge \alpha_i(v_i', a_i, x_i))
\end{aligned}$$

where $f_i(t_i^1, t_i^2, ...) = f_i(t_i'^1, t_i'^2, ...)$ is equality of arguments of functional attribute expressions.

Proof

Let us define $F'(f_1,f_2,...,v'_1,v'_2,a_1,a_2,x_1,x_2)=F(f_1,f_2,...,v_1^1,v_2^1,a_1,a_2)\cup$ $\cup F_b(f_1,f_2,...,v_1^2,v_2^2,a_1,a_2,x_1,x_2)$. We combine all equations with the same name of functional symbol f_i and renaming variables names after such union for equations from basic protocol. After that we obtain sets of variables v'_1,v'_2 and basic protocol $b(a)=\forall x(F_b(f_1,f_2,...,v'_1,v'_2,a_1,a_2,x_1,x_2)\wedge\alpha_1(v'_1,x_1,a_1)\wedge$ $\wedge\alpha_2(v'_2,x_2,a_2)\wedge...\rightarrow<P(a,x)>\alpha_1(v'_1,x_1,a_1)\wedge\alpha_2(v'_2,x_2,a_2))$.

For satisfiable checking we should add corresponded implication for each pair of equation from $F'(f_1,f_2,...,v'_1,v'_2,a_1,a_2,x_1,x_2)$ with the same name of functional symbol f_i.

$$\underset{(i,k,l)}{\wedge}((f_i(t_i^1,t_i^2,...)=f_i^l(t_i'^1,t_i'^2,...))\rightarrow(v_i'^k=v_i'^l))=$$
$$=\underset{i,k,l}{\wedge}(\neg(f_i(t_i^1,t_i^2,...)=f_i(t_i'^1,t_i'^2,...))\vee(v_i'^k=v_i'^l))$$

Each left and right parts of equation and negation of equations are in the same specialization. It means that we could build here a disjunction of conjunction. Each conjunct in such disjunction is q_i which will be in one form of our specialization. So, it means that we could check satisfiability in the following form $\underset{i}{\vee}sat(q_i\wedge S_i(v'_i,a_i)\wedge\alpha'_i(v'_i,a_i,x_i))$.

Theorem is proved.

Let $S(a)$ - environment state, $b(a)=\forall x(\alpha(x,a)\rightarrow<P(a,x)>\beta(x,a)$ - basic protocol.

Theorem 5

$$pt(S(a)\wedge\alpha(x,a),\beta(x,a))=$$
$$=\underset{i}{\vee}(pt'(E_1^i(v'_1,x_1,a_1),S_1(v'_1,a_1)\wedge\alpha_1(v'_1,x_1,a_1),\beta_1(x_1,a_1)))\wedge$$
$$\wedge\, pt'(E_2^i(v'_2,x_2,a_2),S_2(v'_2,a_2)\wedge\alpha_2(v'_2,x_2,a_2),\beta_2(x_2,a_2))))$$

where

$$pt'(E_j^i(v'_j,x_j,a_j),S_j(v'_j,a_j)\wedge\alpha_j(v'_j,x_j,a_j),\beta_j(x_j,a_j))=\underset{k}{\vee}q'_k,$$
$$q'_k=\exists(u,v)(S_i(u,v,z)\wedge\alpha_i(u,v,z)\wedge R(u,v,z)\wedge$$
$$\wedge E_i^j(u,v,z)\wedge E_k(u,v,z)\wedge C(r,s,z)$$

Proof

The sets v'_1,v'_2 are built in the same way as in *theorem 3*. Let us consider a general formula of predicate transformer:

$$pt(D(r,s,z),\beta(r,s,z))=\underset{i}{\vee}\exists(u,v)(D(u,v,z)\wedge R(u,v,z)\wedge E_i(u,v,z)\wedge C(r,s,z).$$

Coefficient $\underset{i}{\vee} E_i(u,v,z)$ looks like disjunction of conjunction of all possible matchings with functional attribute expressions from sets *r,s* and *z*. So, we can present it as conjunction of two disjunctions: $\underset{i}{\vee} E_i(u,v,z) = (\underset{k}{\vee} E_k(u,v,z)) \wedge (\underset{l}{\vee} E_l(u,v,z))$ where $\underset{k}{\vee} E_k(u,v,z)$ is disjunction for matching of functional attribute expression where parameters and its value are from different sets of a_j. $\underset{l}{\vee} E_l(u,v,z)$ is a disjunction of matching of other functional attribute expression. Each conjunct of such disjunction could be considered as a conjunction which depends on different sets of memory a_j. It means that disjunction of conjunction $\underset{k}{\vee} E_k(u,v,z)$ could be prepared early before calling of some *pt* function without corresponded substitution of *x,y*. So, $\underset{k}{\vee} E_k(u,v,z) = \underset{k}{\vee} E_1^k(v_1',x_1,a_1) \wedge E_2^k(v_2',x_2,a_2)$. Disjunction $\underset{l}{\vee} E_l(u,v,z)$ could be presented in the same way. So, the theorem is proved.

Let $S(a)$ - environment state, $b(a) = \forall x(\alpha(x,a) \rightarrow < P(a,x) > \beta(x,a)$ - basic protocol.

Theorem 6

$$\begin{aligned}
& pt^{-1}(S(a) \wedge C(r,s,z), \beta(x,a)) = \\
& = \underset{i}{\vee}(pt'^{-1}(E_1^i(v_1',x_1,a_1), S_1(v_1',a_1) \wedge C_1(v_1',x_1,a_1), \beta_1(x_1,a_1))) \wedge \\
& \wedge\, pt'^{-1}(E_2^i(v_2',x_2,a_2), S_2(v_2',a_2) \wedge C_2(v_2',x_2,a_2), \beta_2(x_2,a_2))))
\end{aligned}$$

where

$$\begin{aligned}
& pt'^{-1}(E_j^i(v_j',x_j,a_j), S_j(v_j',a_j) \wedge C_j(v_j',x_j,a_j), \beta_j(x_j,a_j)) = \underset{k}{\vee} q_k', \\
& q_k' \exists(u,v)(S_j(v_j',a_j) \wedge C_j(v_j',x_j,a_j) \wedge R'(u,r,s,z) \wedge \\
& \wedge\, E_i^j(u,v,z) \wedge E_k(u,v,z)) \wedge \alpha_j(r,s,z)
\end{aligned}$$

This theorem could be proved in the same mode as theorem 4.

6 Implementation of Specializations Support

The implementation of Specializations Support consists of the following modules:

- *Engine* module chooses basic protocol for the application according to the user settings.
- *visited/cycle* module checks if a new state after application of the basic protocol is the same as some other state in the current trace for cycle and in another trace for visited.
- *goal/safety* module checks reachability of the user defined goal or safety conditions.

- *Loader* module loads the user defined data.
- *Clew analyzer* module translates the user defined data into data structures for specializations.
- *Meta Apply BP* module applies basic protocols to the environment state with specializations.
- *Meta SAT* module uses general algorithm of satisfiability checking.
- *Spec SAT 1,…,Spec SAT N* modules use specialized algorithms of satisfiability checking of the formula with specializations.
- *Meta PT* module uses general algorithm of forward/backward predicate transformer.
- *Spec PT 1, …, Spec PT N* modules use specialized algorithms of forward and backward predicate transformers.

The interaction of such modules is represented in Fig. 1.

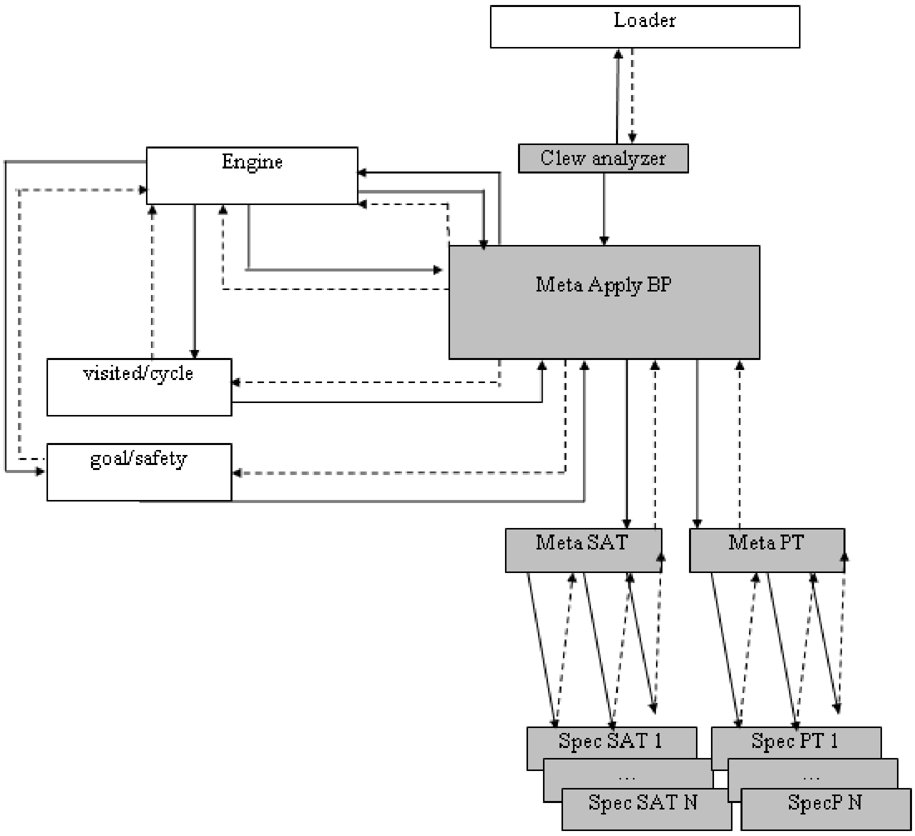


Fig. 1. Interaction of modules in the process of verification of a model

In Fig.1 white rectangles represent modules for classical process of verification, and grey rectangles are for supporting of specializations. So, let us consider in details how it works.

The *Loader* translates user input data into internal representation of general algorithms of satisfiability and predicate transformer. After that *Clew analyzer* tries to split initial environment state and all basic protocols by data for specializations and saves additional information for calling *Meta SAT, Spec SAT 1, …, Spec SAT N, Meta PT, Spec PT 1, …, Spec PT N* modules. After this preparation the model could be run. *Engine* chooses basic protocol, which could be applied with the help of many strategies. Then, it calls *Meta Apply BP* to check applicability of chosen basic protocol. Meta Apply BP tries to apply the satisfiability and predicate transformer algorithms according to the specializations and returns results of such application: *applicable*, *not applicable* or *unknown*. If the result is *unknown,* then *Engine* generates corresponded trace. If result is *not applicable,* then *Engine* tries to check applicability of another basic protocol. If there are no protocols to be applied, then *Engine* generates *deadlock* trace. If basic protocol is *applicable,* then *Engine* tries to check *visited/cycle*. *Visited/cycle* uses satisfiability algorithms from *Meta Apply BP* to check corresponded reachability. *Visited/cycle* returns the following results: *reached*, *not reached* and *unknown*. If result is *reached* or *unknown,* then *Engine* generates the corresponded trace. If result is *not reached,* then *Engine* tries to check *goal/safety*. *Goal/safety* uses satisfiability algorithms from *Meta Apply BP* to check the corresponded reachability. *Goal/safety* returns the following results: *reached*, *not reached* and *unknown*. If result is *reached* or *unknown,* then *Engine* generates the corresponded trace. If result is *not reached,* then *Engine* makes the next iteration of verification. It selects new environment states and tries to find applicability of a basic protocol.

7 Experiments

To evaluate the effectiveness of our approach we present results of our experiments. We run our experiments under Linux Debian version 2.6.26-2-686 on an Intel Xeon E5540 machine with 2.53 GHz and 3 GB RAM. Our framework uses one core for running experiments.

The experiments are rather complicated and complex. The following tasks such as the transformation of the formulae in terms of predicate transformer, saving results of this transformation for the next satisfiability checking, visited state detection and so on are not supported tasks of solvers. At the time of our experiments solvers provide a satisfiability checking of formulae.

From solvers that take part in SMT competition [28] we select Z3 solver according to the following reasons:

- winner of the last SMT-comp in many divisions,
- availability of documentation,
- good support.

Formulae for experiments were selected from SMT-LIB benchmark [29]. We consider them as typical system state formulae. Also we restrict some of them in order to obtain linear formulae after predicate transformer. If we do not provide such restriction a degree of non-linear formulae increases and satisfiability function fails. Then we declare attributes which are changed by means of predicate transformer function. Then we compare the time of modeling of the satisfiability and the predicate transformer, presented in the Section 2, and these algorithms with the specializations.

The first group of experiments refers to specialization by memory usage. Let us present one simple example.

Attributes declaration (environment description) is:

```
(declare-fun P_2 () Int)
(declare-fun P_3 () Int)
(declare-fun P_4 () Int)
(declare-fun P_5 () Int)
(declare-fun P_6 () Int)
(declare-fun P_7 () Int)
(declare-fun P_8 () Int)
(declare-fun P_9 () Int)
(declare-fun mux_I_84972 () Int)
(declare-fun mux_I_84998 () Int)
(declare-fun mux_I_84981 () Int)
(declare-fun dz () Int)
(declare-fun rz () Int)
```

Typical formula of the system state over declared attributes is:

```
(assert (<= 0 P_2))
(assert (<= P_2 43))
(assert (<= (* 939524288 (- 1)) P_3))
(assert (<= P_3 (* 1 (- 1))))
(assert (<= 0 P_4))
(assert (<= P_4 3))
(assert (<= 0 P_5))
(assert (<= P_5 3))
(assert (<= 0 P_6))
(assert (<= P_6 384829079))
(assert (<= 0 P_7))
(assert (<= P_7 15))
(assert (<= 0 P_8))
(assert (<= P_8 58720268))
(assert (<= 0 P_9))
```

```
(assert (<= P_9 15))
(assert (<= 0 mux_I_84972))
(assert (<= mux_I_84972 439804651))
(assert (<= (* 1073741824 (- 1)) mux_I_84998))
(assert (<= mux_I_84998 1073741823))
(assert (<= 0 mux_I_84981))
(assert (<= mux_I_84981 15))
```

The goal is to check satisfiability of the state formula, to modify it according to postcondition of some applicable basic protocol and to provide a hashing of the result for quicker visited/cycle detection mechanism. As it is mentioned in first section usually during modeling of real projects we deal with complex environment states and simple pre- and postconditions of basic protocols.

Let us consider that P_8 and mux_I_84972 are concrete attributes. They have concrete initial values:

P_8: 0, mux_I_84972: 50000,

and are changed by basic protocol as follows:

P_8 = P_8 + 1;
mux_I_84972 = mux_I_84972 – 1;

It means that these two attributes have concrete values all time during trace generation (basic protocols do not change those to symbolic ones). Other attributes are symbolic. We provide a specialization for attributes which are always concrete. We shall check them without calling a solver, if they are valid, we call solver to check satisfiability of the rest of the formula. If sat is returned we simply increment P_8 and decrement mux_I_84972. Results are presented in Table 6. The difference of modeling time for this example and for that one specialized by concrete values is more than in 3 times. Of course, the speedup depends on the project: more concrete attributes we have, more speedup we shall obtain. In [25] it was shown that speedup could be in thousands times.

The second group of experiments refers also to specialization by memory usage, but not to concrete values.

Let us consider an example.

Attributes declaration is the same as above one. The formula of the system state over declared attributes is:

```
(assert (<= P_1 P_2))
(assert (<= 0 P-3))
(assert (let ((?v_0 (+ 8388608 (+ (* P_4 2097152)
P_5)))) (let ((?v_1 (< (+ (+ (+ (+ (+ mux_I_84998 (* (*
?v_0 3) 64)) (- (* 1 (- 1)) P_6)) 1) (* mux_I_84981
P_8)) (* (* P_9 ?v_0) 4)) 0))) (= (+ (* 2 dz) 1) (-
(ite (and (not (= mux_I_84972 0)) ?v_1) 1 0) (ite ?v_1
1 0)))))).
```

Attributes from this formula could be divided into several sets:

1. P_1, P_2,
2. P_3
3. P_4, P_5, P_6, P_8, P_9, mux_I_84998, mux_I_84981, mux_I_84972, and dz.

Memories of attributes in each set are intersected, but memories of attributes from different sets are independent. So we detect what attributes are involved in post-condition and apply predicate transformer only to sets in which these attributes occur.

Moreover, for some groups it is possible to provide a specialized sat and pt functions. For example for the first and second sets satisfiability checking looks like just crossing of two numerical intervals. This approach works faster than general satisfiability checking and quantifiers eliminations.

Then, we inspect a set of industrial projects from out suite. First of all we provide the splitting of formulae into two parts according to attribute types: enumerated part and integer part. For the enumerated part we use *bitsets*, for integer – common Pressburger algorithm. Speedup was about 5-7%. After we specialize an integer part. We consider the attributes which memory is independent and obtain speedup in 10 times.

Let us present how we deal with such kind of formulae on example (formulae will be given in VRS language [5]).

Attributes declaration (environment description) is:

```
types:obj(enumT:(ON,OFF));

attributes:obj(x:int,y:int,z:int,u:int,f:(int)->enumT,
Button:enumT);
```

Initial state is: $f(y) = ON$

Basic protocol is:

$$\neg(Button = OFF) \wedge (f(x) = OFF) \wedge (x = 2y + 10) \wedge (z = 0) \rightarrow<>$$
$$(f(x) := f(x+1)) \wedge (y := x + 10) \wedge u \geq x \wedge (z := z + 1)$$

Hereinafter by *Gen Env(x)*, *Spec Env$_1$(x$_1$)*,… we mean formulae in terms of *Gen Env, Spec Env1*,… data structures.

Initial state with specializations is:

$$Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(y))) \wedge$$
$$Spec\ Env_1(v_1 = ON) \wedge$$
$$Spec\ Env_2(1) \wedge$$
$$Spec\ Env_3(Button = \{ON, OFF\}) \wedge$$
$$Spec\ Env_4(z = (-\infty;+\infty))$$

Precondition with specializations is:

$$Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(x))) \wedge$$
$$Spec\ Env_1(v_1 = OFF) \wedge$$
$$Spec\ Env_2(x = 2y + 10) \wedge Spec\ Env_3(Button = ON) \wedge Spec\ Env_4(1) \wedge$$
$$\wedge\ Gen\ Env(1) \wedge Spec\ Env_1(1) \wedge Spec\ Env_2(u < x) \wedge Spec\ Env_3(1) \wedge$$
$$\wedge\ Spec\ Env_4(z = [0;0])$$

Postcondition with specializations is:

$$Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(x)),$$
$$Spec\ Env_1(v_2) = f(Spec\ Env_2(x+1))) \wedge$$
$$\wedge\ Spec\ Env_1(v_1 := v_2) \wedge$$
$$Spec\ Env_2((y := x + 10) \wedge u \geq x) \wedge$$
$$\wedge\ Spec\ Env_3(1) \wedge Spec\ Env_4(z := z + 1)$$

We try to apply the basic protocol to initial state. Each conjunct of disjunction of precondition is checked separately. Let us consider the conjunction of the environment state and first conjunct of the precondition:

$$Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(y))) \wedge Spec\ Env_1(v_1 = ON) \wedge$$
$$\wedge\ Spec\ Env_2(1) \wedge Spec\ Env_3(Button = \{ON, OFF\}) \wedge Spec\ Env_4(z = (-\infty;+\infty)) \wedge$$
$$\wedge\ Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(x))) \wedge Spec\ Env_1(v_1 = OFF) \wedge$$
$$\wedge\ Spec\ Env_2(x = 2y + 10) \wedge Spec\ Env_3(Button = ON) \wedge Spec\ Env_4(1) \Rightarrow$$
$$\Rightarrow Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(y)), Spec\ Env_1(v_2) =$$
$$= f(Spec\ Env_2(x))) \wedge Spec\ Env_1((v_1 = ON) \wedge (v_2 = OFF)) \wedge$$
$$\wedge\ Spec\ Env_2(x = 2y + 10) \wedge Spec\ Env_3((Button = \{ON, OFF\}) \wedge$$
$$\wedge\ (Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(y)), Spec\ Env_1(v_2) =$$
$$= f(Spec\ Env_2(x))) \wedge Spec\ Env_1((v_1 = ON) \wedge (v_2 = OFF)) \wedge$$
$$\wedge\ Spec\ Env_2(x = 2y + 10) \wedge Spec\ Env_3(Button = ON) \wedge Spec\ Env_4(z = (-\infty;+\infty))$$

Only those specializations which are changed by precondition shall be checked for satisfiability (all specializations connected with *Gen Env* shall be checked too). Let us analyze functional attributes from *Gen Env*:

$$Spec\ Env_2(x = y) \rightarrow Spec\ Env_1(v_2 = v_1) \Rightarrow$$
$$Spec\ Env_2(\neg(x = y)) \vee Spec\ Env_1(v_2 = v_1)$$

$$Sat(E_1 \wedge (Spec\ Env_2(\neg(x = y)) \vee Spec\ Env_1(v_2 = v_1))) \Rightarrow$$
$$\Rightarrow Sat(E_1 \wedge Spec\ Env_2(\neg(x = y)) \vee E_1 \wedge Spec\ Env_1(v_2 = v_1)) \Rightarrow$$
$$\Rightarrow Sat(E_1 \wedge Spec\ Env_2(\neg(x = y))) \vee Sat(E_1 \wedge Spec\ Env_1(v_2 = v_1))$$
$$Sat(E_1 \wedge Spec\ Env_2(\neg(x = y))) \Rightarrow Spec\ Sat_1((v_1 = ON) \wedge (v_2 = OFF)) \wedge$$
$$\wedge\ Spec\ Sat_2((x = 2y + 10) \wedge \neg(x = y)) \wedge Spec\ Sat_3(Button = ON) \Rightarrow$$
$$\Rightarrow 1 \wedge 1 \wedge 1 = 1$$

$Spec\ SAT_4(z = (-\infty;+\infty))$ shall not be checked because the precondition does not change attribute z value.

Let us apply postcondition:

$Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(x)), Spec\ Env_1(v_2) =$
$= f(Spec\ Env_2(x+1))) \wedge Spec\ Env_1(v_1 := v_2) \wedge Spec\ Env_2((y := x+10) \wedge (u \geq x)) \wedge$
$\wedge Spec\ Env_3(1) \wedge Spec\ Env_4(z := z+1)$

Unite these formulae by a conjunction:

$Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(y)), Spec\ Env_1(v_2) =$
$f(Spec\ Env_2(x))) \wedge Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(x)), Spec\ Env_1(v_2) =$
$= f(Spec\ Env_2(x+1))) \Rightarrow$
$\Rightarrow Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(y)), Spec\ Env_1(v_2) =$
$= f(Spec\ Env_2(x)), Spec\ Env_1(v_3) = f(Spec\ Env_2(x+1)))$

Update formula of postcondition:

$Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(y)), Spec\ Env_1(v_2) =$
$f(Spec\ Env_2(x)), Spec\ Env_1(v_3) = f(Spec\ Env_2(x+1))) \wedge$
$\wedge Spec\ Env_1(v_2 := v_3) \wedge Spec\ Env_2((y := x+10) \wedge (u \geq x)) \wedge Spec\ Env_3(1) \wedge$
$\wedge Spec\ Env_4(z := z+1)$

Apply it to *Spec Env₁*. The following cases shall be considered:

- $(v_2 = v_1) \wedge (v_2 = v_3) : (x = y) \wedge (x = x+1) \Rightarrow 0$;
- $(v_2 = v_1) \wedge \neg(v_2 = v_3) : (x = y)$;
- $\neg(v_2 = v_1) \wedge (v_2 = v_3) : \neg(x = y) \wedge (x = y) \Rightarrow 0$;
- $\neg(v_2 = v_1) \wedge \neg(v_2 = v_3) : \neg(x = y)$.

1) Consider the case $(v_2 = v_1) \wedge \neg(v_2 = v_3) : (x = y)$. Apply postcondition to *Spec Env₁.*

$Spec\ Env_1((v_1 = ON) \wedge (v_2 = OFF))$,
$Spec\ Env_1(v_2 := v_3)$
$Spec\ Env_1(\exists(t_1)((v_2 = v_3) \wedge (t_1 = ON) \wedge (t_1 = OFF))) \Rightarrow 0$

2) Consider the case $\neg(v_2 = v_1) \wedge \neg(v_2 = v_3) : \neg(x = y)$:

$Spec\ Env_1((v_1 = ON) \wedge (v_2 = OFF))$,
$Spec\ Env_1(v_2 := v_3)$
$Spec\ Env_1(\exists(t_1)((v_2 = v_3) \wedge (v_1 = ON) \wedge (t_1 = OFF))) \Rightarrow Spec\ Env_1((v_2 = v_3) \wedge (v_1 = OFF))$
$Spec\ Env_2(x = 2y+10)$
$Spec\ Env_2((y := x+10) \wedge (u \geq x))$

$Spec\ Env_2(\exists(t_1,t_2,t_3)((t_1 = 2t_2 + 10) \wedge (y = t_1 + 10) \wedge u \geq x \wedge \neg(t_1 = t_2))) \Rightarrow$

$Spec\ Env_2(\exists t_2((y = 2t_2 + 20) \wedge u \geq x \wedge \neg(2t_2 + 10 = t_2))) \Rightarrow$

$Spec\ Env_2((y \bmod 2 = 0) \wedge u \geq x)$

Postcondition does not change *Spec Env₃*:

$Spec\ Env_3(Button = ON)$

Apply postcondition to *Spec Env₄*:

$Spec\ Env_4(z = (-\infty;+\infty))$

$Spec\ Env_4(z := z + 1)$

We obtain $Spec\ Env_4(z = (-\infty;+\infty))$

So, for the first conjunct of precondition after applying the postcondition we obtain:

$$Gen\ Env(Spec\ Env_1(v_1) = f(Spec\ Env_2(y)), Spec\ Env_1(v_2) =$$
$$= f(Spec\ Env_2(x)), Spec\ Env_1(v_3) = f(Spec\ Env_2(x+1))) \wedge$$
$$\wedge\ Spec\ Env_1((v_2 = v_3) \wedge (v_1 = OFF)) \wedge Spec\ Env_2((y \bmod 2 = 0) \wedge u \geq x) \wedge$$
$$\wedge\ Spec\ Env_3(Button = ON) \wedge Spec\ Env_4(z = (-\infty;+\infty))$$

Similar considerations should be applied to the second conjunct of disjunction of the precondition.

To summarize this section the results of comparison of modeling time using general satisfiability functions and functions with specialization are given.

Table 1. Results of the experiments

Group of tests	**General algorithm**	**With specializations**
1	930 sec	300 sec (memory usage/concrete values)
2	300 sec	280 sec (splitting by types)
3	300 sec	33 sec (memory usage/independent memory)

8 Conclusions

Symbolic modeling is a powerful technique for the automated reachability of deadlocks and violations of user-defined properties. The main complexity of the reachability problem is in the complexity of satisfiability and predicate transformer functions. There are a lot of SMT-based techniques which speed up the satisfiability of formulae that satisfy some particular theory. We propose a technique that allows to speedup classical symbolic modeling when formulae could be splitted in several parts and used some special theories for manipulations with them, which are called specializations. The mathematical description of the algorithm for constructing specializations is provided and the correctness of such specializations is proved.

Specializations by memory usage and functional symbols are considered and examples for each are given.

The nearest plans are the investigation of additional kinds of specialization, because the more specializations we have, the more speedup we obtain.

References

1. Symbolic Modeling, http://en.wikipedia.org/wiki/Model_checking
2. Letichevsky, A., Gilbert, D.: A Model for Interaction of Agents and Environments. In: Bert, D., Choppy, C., Mosses, P.D. (eds.) WADT 1999. LNCS, vol. 1827, pp. 311–328. Springer, Heidelberg (2000)
3. Letichevsky, A.: Algebra of Behavior Transformations and its Applications. In: Kudryavtsev, V.B., Rosenberg, I.G. (eds.) Structural Theory of Automata, Semigroups, and Universal Algebra. NATO Science Series II. Mathematics, Physics and Chemistry, vol. 207, pp. 241–272. Springer, Heidelberg (2005)
4. Letichevsky, A., Kapitonova, J., Kotlyarov, V., Letichevsky Jr., A., Nikitchenko, N., Volkov, V., Weigert, T.: Insertion Modeling in Distributed System Design. Problems of Programming (4), 13–39 (2008)
5. Letichevsky, A., Kapitonova, J., Volkov, V., Letichevsky Jr., A., Baranov, S., Kotlyarov, V., Weigert, T.: System Specification with Basic Protocols. Cybernetics and System Analysis (4), 3–21 (2005)
6. Letichevsky, A.A., Godlevsky, A.B., Letichevsky Jr., A.A., Potienko, S.V., Peschanenko, V.S.: Properties of Predicate Transformer of VRS System. Cybernetics and System Analyses (4), 3–16 (2010)
7. Letichevsky, A., Kapitonova, J., Letichevsky Jr., A., Volkov, V., Baranov, S., Kotlyarov, V., Weigert, T.: Basic Protocols, Message Sequence Charts, and the Verification of Requirements Specifications. In: ISSRE 2004, WITUL (Workshop on Integrated reliability with Telecommunications and UML Languages), Rennes (November 4, 2005)
8. Letichevsky, A.A., Letychevskyi, O.A., Peschanenko, V.S.: Insertion Modeling System. In: Clarke, E., Virbitskaite, I., Voronkov, A. (eds.) PSI 2011. LNCS, vol. 7162, pp. 262–273. Springer, Heidelberg (2012)
9. Bofill, M., Nieuwenhuis, R., Oliveras, A., Rodríguez-Carbonell, E., Rubio, A.: The Barcelogic SMT Solver. In: Gupta, A., Malik, S. (eds.) CAV 2008. LNCS, vol. 5123, pp. 294–298. Springer, Heidelberg (2008)
10. Barrett, C., Berezin, S.: CVC Lite: A New Implementation of the Cooperating Validity Checker. In: Alur, R., Peled, D.A. (eds.) CAV 2004. LNCS, vol. 3114, pp. 515–518. Springer, Heidelberg (2004)
11. Barrett, C., Tinelli, C.: CVC3. In: Damm, W., Hermanns, H. (eds.) CAV 2007. LNCS, vol. 4590, pp. 298–302. Springer, Heidelberg (2007)
12. Barrett, C., Conway, C.L., Deters, M., Hadarean, L., Jovanović, D., King, T., Reynolds, A., Tinelli, C.: CVC4. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 171–177. Springer, Heidelberg (2011)
13. Cotton, S., Asarin, E., Maler, O., Niebert, P.: Some Progress in Satisfiability Checking for Difference Logic. In: Proc. FORMATS-FTRTFT (2004)
14. Déharbe, D., Ranise, S.: Bdd-Driven First-Order Satisfiability Procedures (extended version). Research report 4630, LORIA (2002)
15. Bozzano, M., Bruttomesso, R., Cimatti, A., Junttila, T., van Rossum, P., Schulz, S., Sebastiani, R.: An Incremental and Layered Procedure for the Satisfiability of Linear Arithmetic Logic. In: Halbwachs, N., Zuck, L.D. (eds.) TACAS 2005. LNCS, vol. 3440, pp. 317–333. Springer, Heidelberg (2005)
16. Ganai, M.K., Talupur, M., Gupta, A.: SDSAT: Tight Integration of Small Domain Encoding and Lazy Approaches in a Separation Logic Solver. In: Hermanns, H., Palsberg, J. (eds.) TACAS 2006. LNCS, vol. 3920, pp. 135–150. Springer, Heidelberg (2006)

17. Audemard, G., Bertoli, P.G., Cimatti, A., Kornilowicz, A., Sebastiani, R.: A SAT based Approach for Solving Formulas over Boolean and Linear Mathematical Propositions. In: Voronkov, A. (ed.) CADE 2002. LNCS (LNAI), vol. 2392, pp. 195–210. Springer, Heidelberg (2002)
18. Bryant, R.E., Lahiri, S.K., Seshia, S.A.: Modeling and Verifying Systems using a Logic of Counter Arithmetic with Lambda Expressions and Uninterpreted Functions. In: Brinksma, E., Larsen, K.G. (eds.) CAV 2002. LNCS, vol. 2404, pp. 78–92. Springer, Heidelberg (2002)
19. Dutertre, B., de Moura, L.: A Fast Linear-Arithmetic Solver for DPLL(T). In: Ball, T., Jones, R.B. (eds.) CAV 2006. LNCS, vol. 4144, pp. 81–94. Springer, Heidelberg (2006)
20. Walther, C., Schweitzer, S.: About veriFun. In: Baader, F. (ed.) CADE 2003. LNCS (LNAI), vol. 2741, pp. 322–327. Springer, Heidelberg (2003)
21. Ball, T., Cook, B., Lahiri, S.K., Zhang, L.: ZAPATO: Automatic Theorem Proving for Predicate Abstraction Refinement. In: Alur, R., Peled, D.A. (eds.) CAV 2004. LNCS, vol. 3114, pp. 457–461. Springer, Heidelberg (2004)
22. de Moura, L., Bjørner, N.S.: Z3: An Efficient SMT Solver. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 337–340. Springer, Heidelberg (2008)
23. Barrett, C., de Moura, L., Ranise, S., Stump, A., Tinelli, C.: The SMT-LIB Initiative and the Rise of SMT (HVC 2010 Award Talk). In: Raz, O. (ed.) HVC 2010. LNCS, vol. 6504, pp. 3–3. Springer, Heidelberg (2010)
24. Hutter, F., Hoos, H.H., Leyton-Brown, K., Stuetzle, T.: ParamILS: an Automatic Algorithm Configuration Framework. JAIR 36, 267–306 (2009)
25. Peschanenko, V.S., Guba, A.A., Shushpanov, C.I.: Mixed Concrete-Symbolic Predicate Transformer. Bulletin of Taras Shevchenko National University of Kyiv, Series Physics & Mathematics 2 (2013) (in press)
26. Barrett, C., Sebastiani, R., Seshia, S., Tinelli, C.: Satisfiability Modulo Theories. Frontiers in Artificial Intelligence and Applications 185, 825–885 (2009)
27. Godlevsky, A.B.: Predicate Transformers in the Context of Symbolic Modeling of Transition Systems. Cybernetics and System Analysis 4, 91–99 (2010)
28. Satisfiability Modulo Theories Competition (SMT-COMP), `http://smtcomp.sourceforge.net`
29. The Satisfiability Modulo Theories Library, `http://www.smtlib.org/`

Extending Floyd-Hoare Logic for Partial Pre- and Postconditions

Andrii Kryvolap[1], Mykola Nikitchenko[1], and Wolfgang Schreiner[2]

[1] Taras Shevchenko National University of Kyiv, Kyiv, Ukraine
krivolapa@gmail.com, nikitchenko@unicyb.kiev.ua
[2] Johannes Kepler University, Linz, Austria
Wolfgang.Schreiner@risc.jku.at

Abstract. Traditional (classical) Floyd-Hoare logic is defined for a case of total pre- and postconditions while programs can be partial. In the chapter we propose to extend this logic for partial conditions. To do this we first construct and investigate special program algebras of partial predicates, functions, and programs. In such algebras program correctness assertions are presented with the help of a special composition called Floyd-Hoare composition. This composition is monotone and continuous. Considering the class of constructed algebras as a semantic base we then define an extended logic – Partial Floyd-Hoare Logic – and investigate its properties. This logic has rather complicated soundness constraints for inference rules, therefore simpler sufficient constraints are proposed. The logic constructed can be used for program verification.

Keywords: Program algebra, program logic, partial predicate, soundness, composition-nominative approach.

1 Introduction

Program logics are the main formalisms used for proving assertions about program properties. A well-known classical Floyd-Hoare logic (here also referred to as CFHL) [1, 2] is an example of such logics. Semantically, this logic is defined for the case of total predicates though programs can be partial (non-terminating). In this case program correctness assertions can be presented with the help of a special composition over total predicates called Floyd-Hoare composition (FH-composition). However, a straightforward extension of CFHL for partial predicates meets some difficulties. The first one is that the FH-composition will not be monotone with respect to partial predicates. Monotonicity is an important property that gives the possibility to reason about the correctness of the program based on the correctness of its approximations.

That is why the need of a modified definition of the classical Floyd-Hoare logic for the case of partial mappings arises. We construct such logics in this paper and called them Partial Floyd-Hoare Logics (PFHL). Here we will consider only a special case of partial mappings (predicates, ordinary functions, and program functions) defined over sets of named values (nominative sets). Mappings over classes of such sets are called quasiary mappings [3] and corresponding program algebras are called quasiary program algebras. They form the semantic component of PFHL.

The syntactic component of such logics is presented by their languages and systems of inference rules. We study the possibility to use classical rules for modified logics with a monotone Floyd-Hoare composition. Systems of such inference rules should be sound to be of a practical use. This could be achieved by adding proper restrictions (constraints) to the inference rules of the classical Floyd-Hoare logic that fail to be sound. Thus, the proposed scheme permits to define a Floyd-Hoare-like logic for partial pre- and postconditions.

The rest of the chapter is structured as follows. In Section 2 we analyze the traditional Floyd-Hoare logic and its potential to be extended for partial predicates. In Section 3 we describe program algebras of quasiary predicates and functions at different levels of abstraction, define a modified Floyd-Hoare composition and specify the syntax for the modified logic. In Section 4 we study the soundness of the system of inference rules for the introduced program algebras and define constraints for the rules of the systems. We prove that obtained logic is indeed an extension of classical Floyd-Hoare logic. Also simpler constraints are formulated. In Section 5 we describe related work, and finally, we formulate conclusions in Section 6.

2 Analysis of Classical Floyd-Hoare Logic

We first analyze the CFHL constructed for a very simple imperative language **WHILE** [4]. The grammar of the (slightly modified) language is defined as follows:

$a ::= k \mid x \mid a_1 + a_2 \mid a_1 * a_2 \mid a_1 - a_2 \mid (a)$
$b ::= T \mid F \mid a_1 = a_2 \mid a_1 \leq a_2 \mid b_1 \vee b_2 \mid \neg b \mid (b)$
$S ::= x{:=}a \mid \text{skip} \mid S_1 ; S_2 \mid \text{if } b \text{ then } S_1 \text{ else } S_2 \mid \text{while } b \text{ do } S \mid \text{begin } S \text{ end}$

where:

- k ranges over integers $Int=\{\ldots, -2, -1, 0, 1, 2, \ldots\}$,
- x ranges over variables (names) $V=\{N, R, X, Y, Z, \ldots\}$,
- a ranges over arithmetic expressions *Aexpr*,
- b ranges over Boolean expressions *Bexpr*,
- S ranges over statements (programs) *Stm*.

Semantics of arithmetical expressions is denoted as $[[a]]$ and of Boolean expressions as $[[b]]$. Program states (also called data) are considered as collections of named values. Program correctness assertions (referred to simply as assertions) are Floyd-Hoare triples of the form $\{p\}S\{q\}$ where p, q are predicates of some basic predicate logic and S is a statement. An assertion $\{p\}S\{q\}$ is said to be valid [4] if the following proposition holds: if S is started in a state satisfying p, and if S eventually terminates in some final state, then this final state will satisfy q.

Analyzing this definition of assertion validity we should admit that it permits semantic treatment of assertion $\{p\}S\{q\}$ as a certain predicate defined on states. This treatment of assertions will not be monotone under predicate extension. Indeed, consider informally the following assertion:

$$\{T\} \text{ while } T \text{ do skip } \{F\}.$$

This Floyd-Hoare triple will be true on all states because the infinite loop is undefined on all states, and thus on all states the condition of validity for this assertion is satisfied. Now consider a triple $\{T\}$ skip $\{F\}$ that is false on all states. However, the mapping 'skip' is an extension of 'while T do skip'. Thus, monotonicity of assertion validity fails.

Now we make a short analysis of the inference system for **WHILE** presented by rules of Table 1 [4].

Table 1. WHILE inference system for concrete syntax

$\{p[x \mapsto [\![a]\!]]\}\ x := a\ \{p\}$	*R_as*
$\{p\}\text{skip}\{p\}$	*R_skip*
$\dfrac{\{p\}S_1\{q\},\{q\}S_2\{r\}}{\{p\}S_1;S_2\{r\}}$	*R_seq*
$\dfrac{\{[\![b]\!] \wedge p\}S_1\{q\},\{\neg[\![b]\!] \wedge p\}S_2\{q\}}{\{p\}\text{ if } b \text{ then } S_1 \text{ else } S_2\{q\}}$	*R_if*
$\dfrac{\{[\![b]\!] \wedge p\}S\{p\}}{\{p\}\text{ while } b \text{ do } S\{\neg[\![b]\!] \wedge p\}}$	*R_wh*
$\dfrac{\{p'\}f\{q'\}}{\{p\}f\{q\}}$ if $p \rightarrow p'$ and $q' \rightarrow q$	*R_cons*

These rules are oriented on the concrete syntax of **WHILE,** and moreover they use semantic mappings [[a]] and [[b]]. We adopt the semantic-syntactic style, thus, we present these rules as constructed over special semantic program algebra. In this algebra (see the formal definitions in the next section) we semantically treat program structuring constructs as special operations called compositions. For **WHILE** the following compositions are introduced (we use notation of [3, 5]):

- superposition S_P^x ;
- assignment AS^x ;
- sequential execution • ;
- conditional *IF*;
- cycle (loop) *WH*.

In the sequel pre- and postconditions are denoted (possibly with indexes) as p, q, r; ordinary functions as h, s; program functions (semantics of statements) as f, g. Statement 'skip' is semantically represented by identity function *id*. Data (states) are usually denoted as d.

Note, that we do not make an explicit distinction between a formula and its interpretation. Thus, in the assertion $\{p\}f\{q\}$ we treat p and q syntactically as formulas of the logic language and semantically as predicates of the program algebra.

According to the introduced notations the inference system can be presented by rules of Table 2.

Table 2. WHILE inference system for semantic program algebra

$\{S_P^x(p,h)\}AS^x(h)\{p\}$	*R_AS*
$\{p\}id\{p\}$	*R_SKIP*
$\dfrac{\{p\}f\{q\},\{q\}g\{r\}}{\{p\}f\bullet g\{r\}}$	*R_SEQ*
$\dfrac{\{r\wedge p\}f\{q\},\{\neg r\wedge p\}g\{q\}}{\{p\}\ IF(r,f,g)\ \{q\}}$	*R_IF*
$\dfrac{\{r\wedge p\}f\{p\}}{\{p\}\ WH(r,f)\ \{\neg r\wedge p\}}$	*R_WH*
$\dfrac{\{p'\}f\{q'\}}{\{p\}f\{q\}}$ if $p\to p'$ and $q'\to q$	*R_CONS*

In the next sections we define a class of algebras of partial predicates (semantics of logic) and modify the inference system for such predicates, thus obtaining Partial Floyd-Hoare Logics.

3 Quasiary Program Algebras

To modify the classical Floyd-Hoare logic for partial quasiary mappings, we will use semantic-syntactic scheme [3, 5]. This means that we will first define the semantics in the form of classes of quasiary program algebras. Then the language of the logic will be defined as well as the interpretation mappings.

To emphasize a mapping's *partiality/totality* we write the sign $\xrightarrow{p}$ for partial mappings and the sign $\xrightarrow{t}$ for total mappings. Given an arbitrary partial mapping $\mu: D\xrightarrow{p}D'$, $d\in D$, $S\subseteq D$, $S'\subseteq D'$ we write:

- $\mu(d)\downarrow$ to denote that μ is defined on d;
- $\mu(d)\downarrow= d'$ to denote that μ is defined on d with a value d';
- $\mu(d)\uparrow$ to denote that μ is undefined on d;
- $\mu[S]=\{\mu(d)\mid\mu(d)\downarrow,d\in S\}$ to denote the image of S under μ;
- $\mu^{-1}[S']=\{d\mid\mu(d)\downarrow,\mu(d)\in S'\}$ to denote the preimage (inverse image) of S' under μ.

3.1 Classes of Quasiary Mappings

Let V be a set of *names (variables)*. Let A be a set of *basic values*. Given V and A, the class ${}^VA$ of *nominative sets* is defined as the class of all partial mappings from V to A, thus, ${}^VA=V\xrightarrow{p}A$. Informally speaking, nominative sets represent states of variables.

Though nominative sets are defined as mappings, we follow mathematical traditions and also use a set-like notation for these objects. In particular, the notation

$d = [v_i \mapsto a_i \mid i \in I]$ describes a nominative set d where $v_i \mapsto a_i \in_n d$, which means that $d(v_i)$ is defined and its value is a_i $(d(v_i)\downarrow = a_i)$. The main operation for nominative sets is the binary *total overriding operation* ∇: ${}^VA \times {}^VA \xrightarrow{t} {}^VA$ defined by the formula $d_1 \nabla d_2 = [v \mapsto a \mid v \mapsto a \in_n d_2 \vee (v \mapsto a \in_n d_1 \wedge \neg\exists a'(v \mapsto a' \in_n d_2))]$. Intuitively, given d_1 and d_2 this operation yields a new nominative set which consists of named pairs of d_2 and those pairs of d_1 whose names do not occur in d_2.

Let $Bool = \{F, T\}$ be the set of Boolean values. Let $Pr^{V,A} = {}^VA \xrightarrow{p} Bool$ be the set of all partial predicates over ${}^VA$. Such predicates are called *partial quasiary predicates*. Let $Fn^{V,A} = {}^VA \xrightarrow{p} A$ be the set of all partial functions from ${}^VA$ to A. Such functions are called *partial quasiary ordinary functions*. Here 'ordinary' means that the range of such functions is the set of basic values A. Let $FPrg^{V,A} = {}^VA \xrightarrow{p} {}^VA$ be the set of all partial functions from ${}^VA$ to ${}^VA$. Such functions are called *bi-quasiary functions*.

Quasiary predicates represent conditions which occur in programs, quasiary ordinary functions represent the semantics of program expressions, and bi-quasiary functions represent program semantics.

The terms 'partial' and 'ordinary' are usually omitted. In a general term, elements of $Pr^{V,A}$, $Fn^{V,A}$, and $FPrg^{V,A}$ are called *quasiary mappings*.

3.2 Hierarchy of Quasiary Program Algebras and Logics

Based on algebras with three carriers ($Pr^{V,A}$, $Fn^{V,A}$, and $FPrg^{V,A}$) we can define logics of three types (see details in [3, 5]):

1) *pure quasiary predicate logics based on algebras with one sort*: $Pr^{V,A}$;
2) *quasiary predicate-function logics based on algebras with two sorts*: $Pr^{V,A}$ and $Fn^{V,A}$;
3) *quasiary program logics based on algebras with three sorts*: $Pr^{V,A}$, $Fn^{V,A}$, and $FPrg^{V,A}$.

The basic compositions of logics of the first type are disjunction $\vee$, negation $\neg$, renomination $R^{\bar{v}}_{\bar{x}}$, and quantification $\exists x$.

The basic compositions of logics of the second type additionally include superpositions $S^{\bar{v}}_F$ and $S^{\bar{v}}_P$, and denomination function $'x$.

The basic compositions of logics of the third type additionally include the following program compositions: the parametric assignment composition $AS^x : Fn^{V,A} \xrightarrow{t} FPrg^{V,A}$, the composition of sequential execution $\bullet : FPrg^{V,A} \times FPrg^{V,A} \xrightarrow{t} FPrg^{V,A}$, the conditional composition $IF : Pr^{V,A} \times FPrg^{V,A} \times FPrg^{V,A} \xrightarrow{t} FPrg^{V,A}$, the cyclic (loop) composition $WH : Pr^{V,A} \times FPrg^{V,A} \xrightarrow{t} FPrg^{V,A}$, and identity function *id*: $FPrg^{V,A}$. Also we need compositions that describe properties of the programs. The Floyd-Hoare composition $FH : Pr^{V,A} \times FPrg^{V,A} \times Pr^{V,A} \xrightarrow{t} Pr^{V,A}$ is the most important of them. Its formal definition will be given in the next subsection.

3.3 Formal Definition of the Floyd-Hoare Composition

The required definition stems from the analysis of Floyd-Hoare assertions with total predicates (see, for example, [4]). Namely, an assertion $\{p\}f\{q\}$ is said to be valid if and only if

$$\text{for all } d \text{ from } {}^{V}A \text{ if } p(d) = T, f(d)\downarrow = d' \text{ for some } d' \text{ then } q(d') = T.$$

This definition permits to treat $\{p\}f\{q\}$ as a predicate because it is a pointwise definition. Rewriting this definition for different cases we get the following matrices (Table 3) specifying the logical values of $\{p\}f\{q\}$ for an arbitrary d.

Table 3. Logical values of $\{p\}f\{q\}$ for total predicates

a) $f(d)$ is defined

$p(d) \setminus q(f(d))$	F	T
F	T	T
T	F	T

b) $f(d)$ is undefined

$p(d)$	$\{p\}f\{q\}(d)$
F	T
T	T

The example given in Section 2 demonstrates that for partial predicates monotonicity fails for the case when $f(d)$ is undefined and $p(d) \downarrow = T$. Therefore, to define a monotone interpretation of Floyd-Hoare triples for partial predicates we should change the value of $\{p\}f\{q\}$ for this case. Also, we should specify the logical values of $\{p\}f\{q\}$ for the cases when pre- or postconditions are not defined. In Table 4 such unspecified logical values are denoted by the question marks.

Table 4. Logical values of $\{p\}f\{q\}$ for partial predicates, where question mark represents values that should be changed to a Boolean values

a) $f(d)$ is defined

$p(d) \setminus q(f(d))$	F	T	*undefined*
F	T	T	?
T	F	T	?
undefined	?	?	?

b) $f(d)$ is undefined

$p(d)$	$\{p\}f\{q\}(d)$
F	T
T	?
undefined	?

While defining a required composition we adopt the following requirements:

- partiality of mappings;
- monotonicity of a composition on all its arguments;
- maximal definiteness of the obtained predicates (we call this as Kleene's requirement).

To do this we use techniques for non-deterministic semantics described in [6]. We will treat the case when a predicate is '*undefined*' as non-deterministic values T and F. Thus, we can use Boolean values given in Table 4 to evaluate a set of values for cases with question marks. The obtained results are presented in Table 5.

Table 5. Logical values of $\{p\}f\{q\}$ for partial predicates presented as sets of Boolean values

a) $f(d)$ is defined

$p(d) \setminus q(f(d))$	$\{F\}$	$\{T\}$	$\{F,T\}$
$\{F\}$	$\{T\}$	$\{T\}$	$\{T\}$
$\{T\}$	$\{F\}$	$\{T\}$	$\{F,T\}$
$\{F,T\}$	$\{F,T\}$	$\{T\}$	$\{F,T\}$

b) $f(d)$ is undefined

$p(d)$	$\{p\}f\{q\}(d)$
$\{F\}$	$\{T\}$
$\{T\}$	$\{F,T\}$
$\{F,T\}$	$\{F,T\}$

Now, replacing non-deterministic results $\{F, T\}$ on *undefined* we get the final results (Table 6).

Table 6. Logical values of $\{p\}f\{q\}$ for partial predicates with undefined values

a) $f(d)$ is defined

$p(d)\setminus q(f(d))$	F	T	*undefined*
F	T	T	T
T	F	T	*undefined*
undefined	*undefined*	T	*undefined*

b) $f(d)$ is undefined

$p(d)$	$\{p\}f\{q\}(d)$
F	T
T	*undefined*
undefined	*undefined*

The obtained matrices define an interpretation of $\{p\}f\{q\}$ for partial predicates. As was said earlier, we formalize such triples as a Floyd-Hoare composition $FH : Pr^{V,A} \times FPrg^{V,A} \times Pr^{V,A} \xrightarrow{t} Pr^{V,A}$ ($p, q \in Pr^{V,A}, f \in FPrg^{V,A}, d \in {}^{V}A$):

$$FH(p,f,q)(d) = \begin{cases} T, & \text{if } q(f(d)) \downarrow= T \text{ or } p(d) \downarrow= F, \\ F, & \text{if } p(d) \downarrow= T \text{ and } q(f(d)) \downarrow= F, \\ & \text{undefined in other cases.} \end{cases}$$

3.4 Formal Definition of Program Algebra Compositions

In the previous subsection the formal definition of FH-composition was presented. In this subsection we give definitions of other compositions (see details in [3, 5]).

Propositional compositions are defined as follows ($p, q \in Pr^{V,A}, d \in {}^{V}A$):

$$(p \vee q)(d) = \begin{cases} T, & \text{if } p(d) \downarrow= T \text{ or } q(d) \downarrow= T, \\ F, & \text{if } p(d) \downarrow= F \text{ and } q(d) \downarrow= F, \\ \text{undefined in other cases.} \end{cases} \quad (\neg p)(d) = \begin{cases} T, \text{ if } p(d) \downarrow= F, \\ F, \text{if } p(d) \downarrow= T, \\ \text{undefined if } p(d) \uparrow. \end{cases}$$

Unary parametric composition of existential quantification $\exists x$ with the parameter $x \in V$ is defined by the following formula ($p \in Pr^{V,A}, d \in {}^{V}A$):

$$(\exists x\, p)(d) = \begin{cases} T, & \text{if } b \in A \text{ exists: } p(d \nabla x \mapsto b) \downarrow= T, \\ F, & \text{if } p(d \nabla x \mapsto a) \downarrow= F \text{ for each } a \in A, \\ \text{undefined in other cases.} \end{cases}$$

Here $d\nabla x \mapsto a$ is a shorter form for $d\nabla[x \mapsto a]$.

Parametric n-ary superpositions with $\bar{x} = (x_1,...,x_n)$ as the parameter are defined by the following formulas ($h, s_1,\ldots, s_n \in Fn^{V,A}$, $p \in Pr^{V,A}$, $d \in {}^VA$):

$$(S_F^{\bar{x}}(h, s_1,\ldots,s_n))(d) = h(d\nabla[x_1 \mapsto s_1(d),\ldots,x_n \mapsto s_n(d)]) \,,$$
$$(S_P^{\bar{x}}(p, s_1,\ldots,s_n))(d) = p(d\nabla[x_1 \mapsto s_1(d),\ldots,x_n \mapsto s_n(d)]) \,.$$

Null-ary parametric denomination composition with the parameter $x \in V$ is defined by the following formula ($d \in {}^VA$): $'x(d) = d(x)$.

Identical program composition $id \in FPrg^{V,A}$ is simple: $id(d) = d$ ($d \in {}^VA$).

Assignment composition is defined as follows ($h \in Fn^{V,A}$, $d \in {}^VA$):

$$AS^x(h)(d) = d\nabla[x \mapsto h(d)] \,.$$

Composition of sequential execution is introduced in the ordinary way (f, $g \in FPrg^{V,A}$, $d \in {}^VA$):

$$f \bullet g(d) = g(f(d)) \,.$$

Note, that we define $\bullet$ by commuting arguments of conventional functional composition: $f \bullet g = g \circ f$.

Conditional composition depends on the value of the first argument which is the condition itself ($p \in Pr^{V,A}$, f, $g \in FPrg^{V,A}$, $d \in {}^VA$):

$$IF(p,f,g)(d) = \begin{cases} f(d), \text{ if } \; p(d) \downarrow = T, \\ g(d), \text{ if } \; p(d) \downarrow = F, \\ \text{undefined in other cases.} \end{cases}$$

Cycle (loop) composition is defined by the following formulas: $WH(p,f)(d) = d_n$, if there exists a sequence $d_0,...,d_n$ such that $d_0 = d$, $f(d_0) \downarrow = d_1$, …, $f(d_{n-1}) \downarrow = d_n$, $p(d_0) \downarrow = T$, … , $p(d_{n-1}) \downarrow = T$, $p(d_n) \downarrow = F$ ($p \in Pr^{V,A}$, $f \in FPrg^{V,A}$, $d \in {}^VA$).

It means that we have defined the following *quasiary program algebra*:

$QPA(V, A) = < Pr^{V,A}, Fn^{V,A}, FPrg^{V,A}; \vee, \neg, S_F^{\bar{v}}, S_P^{\bar{v}}, 'x, \exists x, id, AS^x, \bullet, IF, WH, FH>$.

The class of such algebras is the main object of our investigation.

3.5 Formal Definition of Program Algebra Terms

Terms of the algebra $QPA(V, A)$ defined over sets of predicate symbols *Ps*, ordinary function symbols *Fs*, program symbols *Prs*, and variables V specify the syntax (the language) of the logic. We now give inductive definitions for terms $Tr(Ps, Fs, Prs, V)$, formulas $Fr(Ps, Fs, Prs, V)$, program texts $Pt(Ps, Fs, Prs, V)$, and Floyd-Hoare assertions $FHFr(Ps, Fs, Prs, V)$.

First we will define terms:

- if $F \in Fs$ then $F \in Tr(Ps, Fs, Prs, V)$;
- if $v \in V$ then $'v \in Tr(Ps, Fs, Prs, V)$;

– if $F \in Fs$, $t_1,\ldots,t_n \in Tr(Ps,Fs,Prs,V)$, and $v_1,\ldots,v_n \in V$ $(n \geq 0)$ are distinct variables then $S_F^{v_1,\ldots,v_n}(F,t_1,\ldots,t_n) \in Tr(Ps,Fs,Prs,V)$.

Then we will define program texts:

– if $Pr \in Prs$ then $Pr \in Pt(Ps,Fs,Prs,V)$;
– $id \in Pt(Ps,Fs,Prs,V)$;
– if $v \in V$ and $t \in Tr(Ps,Fs,Prs,V)$ then $AS^v(t) \in Pt(Ps,Fs,Prs,V)$;
– if $pr_1, pr_2 \in Pt(Ps,Fs,Prs,V)$ then $pr_1 \bullet pr_2 \in Pt(Ps,Fs,Prs,V)$;
– if $pr_1, pr_2 \in Pt(Ps,Fs,Prs,V)$ and $p \in Fr(Ps,Fs,Prs,V)$ then $IF(p, pr_1, pr_2) \in Pt(Ps,Fs,Prs,V)$;
– if $pr \in Pt(Ps,Fs,Prs,V)$ and $p \in Fr(Ps,Fs,Prs,V)$ then $WH(p, pr) \in Pt(Ps,Fs,Prs,V)$.

Finally, formulas and Floyd-Hoare triples are defined:

– if $P \in Ps$ then $P \in Fr(Ps,Fs,Prs,V)$;
– if $\Phi, \Psi \in Fr(Ps,Fs,Prs,V)$ then $\Phi \vee \Psi \in Fr(Ps,Fs,Prs,V)$;
– if $\Phi \in Fr(Ps,Fs,Prs,V)$ then $\neg\Phi \in Fr(Ps,Fs,Prs,V)$;
– if $P \in Ps$, $t_1,\ldots,t_n \in Tr(Ps,Fs,Prs,V)$, and $v_1,\ldots,v_n \in V$ $(n \geq 0)$ are distinct variables then $S_P^{v_1,\ldots,v_n}(P,t_1,\ldots,t_n) \in Fr(Ps,Fs,Prs,V)$;
– if $\Phi \in Fr(Ps,Fs,Prs,V)$ and $v \in V$ then $\exists v\Phi \in Fr(Ps,Fs,Prs,V)$;
– if $f \in Pt(Ps,Fs,Prs,V)$ and $p, q \in Fr(Ps,Fs,Prs,V)$ then $\{p\}f\{q\} \in FHFr(Ps,Fs,Prs,V)$.

After syntax and semantics have been defined, we need to specify the interpretation mappings, assuming that interpretation mappings for the predicate symbols $I_{Ps} : Ps \xrightarrow{t} Pr^{V,A}$, function symbols $I_{Fs} : Fs \xrightarrow{t} Fn^{V,A}$, and program symbols $I_{Prs} : Prs \xrightarrow{t} FPrg^{V,A}$ are given. Let $J_{Fr} : Fr(Fs,Ps,Prs,V) \xrightarrow{t} Pr^{V,A}$ denote an interpretation mapping for formulas, $J_{Tr} : Tr(Fs,Ps,Prs,V) \xrightarrow{t} Fn^{V,A}$ denote an interpretation mapping for terms and $J_{Pt} : Pt(Fs,Ps,Prs,V) \xrightarrow{t} Prg^{V,A}$ denote an interpretation mapping for programs (statements). They are all defined in a natural way, only the case with assertions needs special consideration:

$$J_{FHFr}(\{p\}f\{q\}) = FH(J_{Fr}(p), J_{Pt}(f), J_{Fr}(q)).$$

Thus, an interpretation J is defined by some algebra $QPA(V, A)$ and interpretation mappings I_{Ps}, I_{Fs}, and I_{Prs}. An assertion is said to be *valid* (*irrefutable*) *in an interpretation* J (denoted $J \models_{IR} \{p\}f\{q\}$ or simply $\models \{p\}f\{q\}$) if a predicate obtained under interpretation J is *not refutable*. An assertion is said to be *valid* (denoted $\models \{p\}f\{q\}$) if for any interpretation J we have $J \models \{p\}f\{q\}$. In this chapter we do not define interpretations explicitly expecting that they are clear from the context.

Thus, in the text the main reasoning steps are described for program algebras though to be precise we had to define an interpretation first and then consider predicates of the corresponding program algebra.

3.6 Monotonicity and Continuity of the Floyd-Hoare Composition

In the previous subsections a function-theoretic style of composition definitions was used. To prove properties of the FH-composition, it is more convenient to use a set-theoretic style of definitions.

The following sets are called respectively *truth*, *falsity*, and *undefinedness domains* of the predicate p over D:

$$p^T = \{d \mid p(d) \downarrow = T\},$$

$$p^F = \{d \mid p(d) \downarrow = F\},$$

$$p^\perp = \{d \mid p(d) \uparrow\}.$$

The following definitions introduce various images and preimages involved in Floyd-Hoare composition:

$$q^{-T,f} = f^{-1}[q^T],$$

$$q^{-F,f} = f^{-1}[q^F],$$

$$q^{-\perp,f} = f^{-1}[q^\perp],$$

$$p^{T,f} = f[p^T],$$

$$p^{F,f} = f[p^F],$$

$$p^{\perp,f} = f[p^\perp].$$

Using these notations we can define FH-composition by describing the truth and falsity domains of the predicate that is the result of the composition application:

$$FH(p,f,q)^T = p^F \cup q^{-T,f},$$

$$FH(p,f,q)^F = p^T \cap q^{-F,f}.$$

Validity of formulas (predicates) is considered as irrefutability, that is

$$\models p \Leftrightarrow p^F = \varnothing.$$

From this follows that

$$\models FH(p,f,q) \Leftrightarrow p^T \cap q^{-F,f} = \varnothing.$$

Let us give a formal definition of a monotone composition.

Composition $C : (FPrg^{V,A})^n \times (Pr^{V,A})^k \times (Fn^{V,A})^m \xrightarrow{t} Pr^{V,A}$ (with the class of predicates as its range) is called *monotone* if the following condition holds for all arguments of C:

$$f_1 \subseteq g_1, \ldots, f_n \subseteq g_n, p_1 \subseteq q_1, \ldots, p_k \subseteq q_k, h_1 \subseteq s_1, \ldots, h_m \subseteq s_m \Rightarrow$$

$$C(f_1, \ldots, f_n, p_1, \ldots, p_k, h_1, \ldots, h_m) \subseteq C(g_1, \ldots, g_n, q_1, \ldots, q_k, s_1, \ldots, s_m).$$

Here relation of partial order $\subseteq$ is defined as inclusion of graphs of the arguments (which are mappings) of this relation.

Theorem 1. Floyd-Hoare composition is monotone for every argument.

Let us prove monotonicity for every argument separately, examining their truth and falsity domains. For the first argument (precondition) we have:

$$p_1 \subseteq p_2 \Rightarrow p_1^F \subseteq p_2^F \Rightarrow p_1^F \cup q^{-T,f} \subseteq p_2^F \cup q^{-T,f} \Rightarrow$$
$$FH(p_1,f,q)^T \subseteq FH(p_2,f,q)^T$$

and

$$p_1 \subseteq p_2 \Rightarrow p_1^T \subseteq p_2^T \Rightarrow p_1^T \cap q^{-F,f} \subseteq p_2^T \cap q^{-F,f} \Rightarrow$$
$$FH(p_1,f,q)^F \subseteq FH(p_2,f,q)^F .$$

Thus, $p_1 \subseteq p_2 \Rightarrow FH(p_1,f,q) \subseteq FH(p_2,f,q)$.

For the third argument (postcondition) the proof is similar:

$$q_1 \subseteq q_2 \Rightarrow q_1^T \subseteq q_2^T \Rightarrow q_1^{-T,f} \subseteq q_2^{-T,f} \Rightarrow p^F \cup q_1^{-T,f} \subseteq p^F \cup q_2^{-T,f} \Rightarrow$$
$$FH(p,f,q_1)^T \subseteq FH(p,f,q_2)^T$$

and

$$q_1 \subseteq q_2 \Rightarrow q_1^F \subseteq q_2^F \Rightarrow q_1^{-F,f} \subseteq q_2^{-F,f} \Rightarrow p^T \cap q_1^{-F,f} \subseteq p^T \cap q_2^{-F,f} \Rightarrow$$
$$FH(p,f,q_1)^F \subseteq FH(p,f,q_2)^F .$$

Thus, $q_1 \subseteq q_2 \Rightarrow FH(p,f,q_1) \subseteq FH(p,f,q_2)$.

Let us show the monotonicity of FP-composition for the second argument. For the truth domains we have:

$$f_1 \subseteq f_2 \Rightarrow q^{-T,f_1} \subseteq q^{-T,f_2} \Rightarrow p^F \cup q^{-T,f_1} \subseteq p^F \cup q^{-T,f_2}$$
$$\Rightarrow FH(p,f_1,q)^T \subseteq FH(p,f_2,q)^T .$$

Similar, for the falsity domains:

$$f_1 \subseteq f_2 \Rightarrow q^{-F,f_1} \subseteq q^{-F,f_2} \Rightarrow p^T \cap q^{-F,f_1} \subseteq p^T \cap q^{-F,f_2} \Rightarrow$$
$$FH(p,f_1,q)^F \subseteq FH(p,f_2,q)^F .$$

Therefore, $f_1 \subseteq f_2 \Rightarrow FH(p,f_1,q) \subseteq FH(p,f_2,q)$.

Thus, it was shown that the composition is monotone for every component, what was needed to prove.

For the constructed composition even stronger result is true, it is continuous. To show this, the following definitions are made and the notion of continuity is given (see, for example, [4]).

An infinite set of indexed mappings $\{\mu_0, \mu_1, \ldots\}$ with $\mu_i \subseteq \mu_{i+1}, i \in \omega$ is called a *chain* of mappings.

The supremum of the above-mentioned set of indexed mappings is called *limit* of the chain, denoted as $\coprod_i \mu_i$.

The composition $C:(Prg^{V,A})^n \times (Pr^{V,A})^m \times (Fn^{V,A})^l \xrightarrow{t} Pr^{V,A}$ is called *continuous* on the first argument if for arbitrary chain $\{f_i \mid i \in \omega\}$ the following property holds:

$$C(\coprod_i f_i, g_2, \dots, g_n, p_1, \dots p_m, h_1, \dots h_l) = \coprod_i C(f_i, g_2, \dots, g_n, p_1, \dots p_m, h_1, \dots h_l).$$

Continuity on the other arguments is defined in a similar manner.

Theorem 2. Floyd-Hoare composition is continuous on every argument.

Though this result follows from the general consideration, we give here its direct proof. Let us show the continuity on the first argument. In the case of other arguments the proof will be similar.

Consider a chain of predicates $\{p_i \mid i \in \omega\}$. Since Floyd-Hoare composition is monotone, $\{FH(p_i, f, q) \mid i \in \omega\}$ will also be a chain. We need to show that $FH(\coprod_i p_i, f, q) = \coprod_i FH(p_i, f, q)$. To do this we demonstrate that $FH(\coprod_i p_i, f, q)(d)$ is defined iff $(\coprod_i FH(p_i, f, q))(d)$ is defined, and in this case $FH(\coprod_i p_i, f, q)(d) = (\coprod_i FH(p_i, f, q))(d)$ for the arbitrary data d.

There are two different possibilities – $(\coprod_i p_i)(d)\uparrow$ and $(\coprod_i p_i)(d)\downarrow$.

In the first case none of the elements of the chain is defined on d. Therefore $(\coprod_i FH(p_i, f, q))(d)$ is defined iff $q(f(d))\downarrow = T$. But this means that both sides of the equality have the same value.

In the second case ($(\coprod_i p_i)(d)\downarrow$), an element of the chain that is also defined on this data could be found. Otherwise the limit would have been undefined on d, what is guaranteed by the inclusion relation on the elements of the chain. Thus, there exists k such that $p_k(d)\downarrow$. Therefore

$$FH(\coprod_i p_i, f, q)(d) = FH(p_k, f, q)(d) \text{ and}$$

$$FH(p_k, f, q)(d) = (\coprod_i FH(p_i, f, q))(d),$$

since for any *i*: $i > k, p_i(d)\downarrow = p_k(d)$ by the definition of the chain.

The following equality is obtained: $FH(\coprod_i p_i, f, q)(d) = (\coprod_i FH(p_i, f, q))(d)$. Since the data was chosen arbitrary, we get $FH(\coprod_i p_i, f, q) = \coprod_i FH(p_i, f, q)$, what was needed to prove.

The proof for the other arguments is similar. Thus, the monotone Floyd-Hoare composition is continuous on every argument.

The theorems 1 and 2 often permit to consider instead of programs with cycles their cycle-free approximations.

4 Inference System for PFHL

In this section we investigate possibility to use inference rules of Table 2 for PFHL.

4.1 Soundness of Classical Inference System for PFHL

Analysis of inference rules shows that soundness fails for the rules *R_SEQ*, *R_WH*, and *R_CONS*. This can be demonstrated with the following examples.

First, let us show that for some interpretation there can be such p, q, r, f, and g that $\models \{p\}f\{q\}, \models \{q\}g\{r\} \Rightarrow \models \{p\}f \bullet g\{r\}$ is false.

Consider $p(d)\downarrow = T$, $q(d)\uparrow$ and $r(d)\downarrow = F$ for arbitrary d and $f = g = id$. In this case $\models \{p\}f\{q\}$ and $\models \{q\}g\{r\}$ because $q^T = q^F = \varnothing$, but $\not\models \{p\}f \bullet g\{r\}$. Thus, $\models \{p\}f\{q\}, \models \{q\}g\{r\} \Rightarrow \models \{p\}f \bullet g\{r\}$ is false.

Next example concerns the rule *R_WH*. We need to show that for some r, f, p $\models \{r \wedge p\}f\{p\}$ and $\not\models \{p\}\,WH(r, f)\{\neg r \wedge p\}$.

In this case we need at least three different data (states) $d_1 \neq d_2 \neq d_3$. Then r, f, p are defined in the following manner:

$$r(d) = \begin{cases} T, \text{if } d = d_1 \text{ or } d = d_2, \\ F, \text{if } d = d_3, \\ \text{undefined in other cases.} \end{cases}$$

$$f(d) = \begin{cases} d_3, \text{if } d = d_3, \\ d_2, \text{if } d = d_1, \\ d_3, \text{if } d = d_2, \\ \text{undefined in other cases.} \end{cases}$$

$$p(d) = \begin{cases} T, \text{if } d = d_1, \\ \text{undefined, if } d = d_2, \\ F, \text{if } d = d_3, \\ \text{undefined in other cases.} \end{cases}$$

It is not hard to prove that $\models \{r \wedge p\}f\{p\}$, because $(r \wedge p)(d)\downarrow = T$ only when $d = d_1$, but in this case $f(d)\downarrow = f(d_1)\downarrow = d_2$ and $p(d_2)\uparrow$. This means that there is no such d that $FH(r \wedge p, f, p)(d)\downarrow = F$ in the case of abovementioned interpretations.

Let us show that $\not\models \{p\}WH(r, f)\{\neg r \wedge p\}$. Consider the value of $FH(p, WH(r, f), \neg r \wedge p)(d_1)$.

We have that $p(d_1)\downarrow=T$, $WH(r,f)(d_1)\downarrow=d_3$, $(\neg r\wedge p)(d_3)\downarrow=F$.

Thus, $FH(p,WH(r,f),\neg r\wedge p)(d_1)\downarrow=F$. This gives $\not\models\{p\}WH(r,f)\{\neg r\wedge p\}$.

So, $\models\{r\wedge p\}f\{p\}\Rightarrow\models\{p\}WH(r,f)\{\neg r\wedge p\}$ is false.

The case with the rule *R_CONS* is similar to the previous ones. Consider $p(d)\downarrow=T$, $q'(d)\downarrow=q(d)\downarrow=F$ and $p'(d)\uparrow$ for arbitrary d. Then $\models\{p'\}id\{q'\}$, $p\to p'$ and $q'\to q$, but $\not\models\{p\}id\{q\}$.

Thus, $\{p'\}f\{q'\}, p\to p', q'\to q\Rightarrow\{p\}f\{q\}$ is false.

Given examples prove that additional constraints should be introduced in order for inference system to be sound in the case of partial predicates.

4.2 Composition of Preimage Predicate Transformer

To introduce constraints for the rules of PFHL we need new compositions. They are inspired by the weakest precondition and the strongest postcondition predicate transformers introduced by Dijkstra. But in the case of partial mappings there can be more than one definition what predicate should be considered as weakest (or strongest) therefore more adequate definitions are required. In this chapter we restrict ourselves by introducing only one composition called *composition of preimage predicate transformer* (*preimage composition*). This composition is a generalization of the weakest precondition predicate transformer and is defined in the following way:

$$PC(f,q)(d)=\begin{cases}T, \text{ if } f(d)\downarrow \text{ and } q(f(d))\downarrow=T,\\ F, \text{ if } f(d)\downarrow \text{ and } q(f(d))\downarrow=F,\\ \text{undefined in other cases.}\end{cases}$$

In set-theoretic terms this composition can be defined as follows:

$$PC(f,q)^T=q^{-T,f},\ PC(f,q)^F=q^{-F,f}.$$

Semantically, $PC(f,q)$ can be treated as *backward predicate transformer.*

Introduction of this composition means that now we work with algebras of the form

$$QPAT(V,A)=<Pr^{V,A}, Fn^{V,A}, FPrg^{V,A};$$
$$\vee, \neg, S_F^{\bar{v}}, S_P^{\bar{v}}, 'x, \exists x, id, AS^x, \bullet, IF, WH, FH, PC>.$$

Also, the introduced composition permits to reformulate the assertion validity definition. Preliminary, we define $p\models q$ as $\models p\to q$.

Theorem 3. For any assertion $\{p\}f\{q\}$ the following equivalence holds:

$$\models\{p\}f\{q\}\Leftrightarrow p\models PC(f,q).$$

To prove this theorem we first recall that

$$\models\{p\}f\{q\}\Leftrightarrow\ p^T\cap q^{-F,f}=\varnothing.$$

Therefore $p \models PC(f,q) \Leftrightarrow \models p \to PC(f,q) \Leftrightarrow (p \to PC(f,q))^F = \varnothing \Leftrightarrow$

$$\Leftrightarrow p^T \cap PC(f,q))^F = \varnothing \Leftrightarrow \ p^T \cap q^{-F,f} = \varnothing \Leftrightarrow \models \{p\}f\{q\}.$$

4.3 Constraints for Partial Predicate Inference System

Analysis of the constraint problem demonstrates that for an inference rule different constraints can be formulated. We start with the constraints that practically are reformulations of conditions of assertion validity. Such constraints will be called *trifling constraints* because they do not give additional knowledge of assertion validity. Constraints will be formulated in terms of the preimage predicate transformer.

The examples showed that validity constraints are required for the rules *R_SEQ*, *R_WH*, and *R_CONS*. Trifling constraints are the following:

- $p \models PC(f \bullet g, r)$ for *R_SEQ*,
- $p \models PC(WH(r,f), \neg r \wedge p)$ for *R_WH*,
- $p \models PC(f,q)$ for *R_CONS*.

These constraints in a quite natural sense are necessary and sufficient. But in this form such constraints are not very useful, especially the constraint for *R_CONS* because it does not relate premises with conclusions. Therefore we formulate a more stronger constraint for this rule, which will be sufficient but not necessary.

At first we introduce two *special logical consequence relations*: *over the truth domain* $\models_T$ and *over the falsity domain* $\models_F$ in the following way:

- $p \models_T q$ iff $p_J{}^T \subseteq q_J{}^T$ for every interpretation *J*,
- $p \models_F q$ iff $q_J{}^F \subseteq p_J{}^F$ for every interpretation *J*.

In these terms a new constraint for *R_CONS* is $p \models_T p', q' \models_F q$. This gives us a PFHL inference system with constraints for **WHILE** presented in Table 7.

Table 7. PFHL inference system for **WHILE** with constraints in backward form

$\{S_P^x(p,h)\}AS^x(h)\{p\}$	*R_AS'*
$\{p\}id\{p\}$	*R_SKIP'*
$\dfrac{\{p\}f\{q\},\{q\}g\{r\}}{\{p\}f \bullet g\{r\}}, p \models PC(f \bullet g, r)$	*R_SEQ'*
$\dfrac{\{r \wedge p\}f\{q\},\{\neg r \wedge p\}g\{q\}}{\{p\}IF(r,f,g)\{q\}}$	*R_IF'*
$\dfrac{\{r \wedge p\}f\{p\}}{\{p\}WH(r,f)\{\neg r \wedge p\}}, p \models PC(WH(r,f), \neg r \wedge p)$	*R_WH'*
$\dfrac{\{p'\}f\{q'\}}{\{p\}f\{q\}}, p \models_T p', q' \models_F q$	*R_CONS'*

In this table a constrained rule consists of two parts: pure inference rule and rule constraint written on the right side of the pure rule.

Theorem 4. PFHL inference rules of Table 7 are sound. That means:

1. $\models \{S_P^x(p,h)\} AS^x(h)\{p\}$,
2. $\models \{p\}\, id\, \{p\}$,
3. $\models \{p\} f\{q\}, \models \{q\} g\{r\}, p \models PC(f \bullet g, r) \Rightarrow\ \models \{p\} f \bullet g\{r\}$,
4. $\models \{r \wedge p\} f\{q\}, \models \{\neg r \wedge p\} g\{q\} \Rightarrow\ \models \{p\} IF(r,f,g)\{q\}$,
5. $\models \{r \wedge p\} f\{p\},\ p \models PC(WH(r,f), \neg r \wedge p) \Rightarrow\ \models \{p\}\, WH(r,f)\{\neg r \wedge p\}$,
6. $\models \{p'\} f\{q'\}, p \models_T p', q' \models_F q \Rightarrow\ \models \{p\} f\{q\}$.

Let us prove this for each rule. Recall our assumption that such properties are proved for an implicitly given arbitrary interpretation *J*.

1. For $\models \{S_P^x(p,h)\} AS^x(h)\{p\}$ to hold it is required that

$$FH(S_P^x(p,h), AS^x(h), p)^F = (S_P^x(p,h))^T \cap p^{-F, AS^x(h)} = \varnothing .$$

Let d be any data such that $d \in (S_P^x(p,h))^T$. Then $p(d\nabla[x \mapsto h(d)]) \downarrow = T$. By definition of assignment composition it means that $d \in p^{-T, AS^x(h)}$. So, $(S_P^x(p,h))^T \cap p^{-F, AS^x(h)} = \varnothing$ and $\models \{S_P^x(p,h)\} AS^x(h)\{p\}$.

2. $\models \{p\} id \{p\}$ follows from the definition of *id*:

$$FH(p, id, p)^F = p^T \cap p^{-F, id} = p^T \cap p^F = \varnothing .$$

3. Soundness condition for rule *R_SEQ'* is obvious by theorem 3.

4. Let us prove $\models \{r \wedge p\} f\{q\}, \models \{\neg r \wedge p\} g\{q\} \Rightarrow\ \models \{p\} IF(r,f,g)\{q\}$.

Since $\models \{r \wedge p\} f\{q\}$, $\models \{\neg r \wedge p\} g\{q\}$ we have:

$$(r \wedge p)^T \cap q^{-F,f} = \varnothing; (\neg r \wedge p)^T \cap q^{-F,g} = \varnothing .$$

We need to show that $p^T \cap q^{-F, IF(r,f,g)} = \varnothing$.

Let d be any data such that $p(d) \downarrow = T$ and $IF(r,f,g)(d) \downarrow$. If $r(d) \downarrow = T$ then *q(f (d))* $\downarrow =$ *T* or is undefined by the first premise; if *r(d)* $\downarrow =$ *F* then *q(g(d))* $\downarrow =$ *T* or is undefined by the second premise. Therefore $\models \{p\} IF(r,f,g)\{q\}$.

5. Soundness condition for rule *R_WH'* is obvious by theorem 3.

6. Let us prove $\models \{p'\} f\{q'\}, p \models_T p', q' \models_F q \Rightarrow\ \models \{p\} f\{q\}$.

We have $\models \{p'\} f\{q'\}$ what means $p'^T \cap q'^{-F,f} = \varnothing$.

Also we have $p \models_T p'$ and $q' \models_F q$; using definitions we get $p^T \subseteq p'^T$ and $q^F \subseteq q'^F$.

We need to show that $p^T \cap q^{-F,f} = \varnothing$.

Let d be any data such that $p(d) \downarrow = T$, $f(d) \downarrow$, and $q'(f(d)) \downarrow$. By the second premise $p'(d) \downarrow = T$, by the first premise $q'(f(d)) \downarrow = T$. If $q(f(d)) \downarrow$ then

$q(f(d))\downarrow = T$ by the third premise; therefore $d \notin q^{-F,f}$. If $q(f(d))\uparrow$ then also $d \notin q^{-F,f}$. Thus, in both cases $p^T \cap q^{-F,f} = \varnothing$.

So, all rules are inspected and the theorem is proved.

Now we need to show that for total predicates properties of the classical Floyd-Hoare logic will be preserved and that defined logic will be an extension of the Floyd-Hoare logic. This means that for total predicates a derivation of a Floyd-Hoare assertion in PFHL can be transformed to a derivation of this assertion in CFHL and vice versa: a derivation in CFHL can be presented as derivation in PFHL. This property holds because constraints of rules *R_SEQ'* and *R_WH'* will be satisfied automatically in the case of total predicates; as to *R_CONS'* its constraint can be reduced to the constraint of the rule *R_cons* of CFHL. This will be granted by Theorem 5. But before that we show that for total predicates assertion validity in PFHL ($\models$) is equivalent to validity in CFHL ($\models_{CL}$).

If we recall definitions of the classical (denoted FH_{CL}) and monotone compositions *FH* we will have:

$$FH_{CL}(p,f,q)=\begin{cases} T, & \text{if } p(d)=F \text{ or } f(d)\uparrow \text{ or } (f(d)\downarrow \text{ and } q(f(d))=T),\\ F, & \text{if } p(d)=T,\ f(d)\downarrow, \text{ and } q(f(d))=F.\end{cases}$$

$$FH(p,f,q)=\begin{cases} T, & \text{if } p(d)\downarrow = F \text{ or } (f(d)\downarrow \text{ and } q(f(d))\downarrow = T)\ ,\\ F, & \text{if } p(d)\downarrow = T\ , f(d)\downarrow, \text{ and } q(f(d))\downarrow = F,\\ \text{undefined in other cases.} \end{cases}$$

By the definitions, for total predicates $FH(p,f,q)^F = FH_{CL}(p,f,q)^F$.

Thus, $FH(p,f,q)^F = \varnothing \Leftrightarrow FH_{CL}(p,f,q)^F = \varnothing$. But

$$\models \{p\}f\{q\} \Leftrightarrow \models FH(p,f,q)^F = \varnothing \text{ and}$$
$$\models_{CL} \{p\}f\{q\} \Leftrightarrow \models_{CL} FH(p,f,q)^F = \varnothing.$$

So, we obtain $\models \{p\}f\{q\} \Leftrightarrow \models_{CL} \{p\}f\{q\}$. It means that for total predicates classes of valid assertions in PFHL and CFHL are the same.

Theorem 5. For total predicates the inference rules of PFHL (Table 7) can be reduced to the inference rules of CFHL (Table 2).

To prove the theorem we should demonstrate that for total predicates the constraints of *R_SEQ'* and *R_WH'* hold. It means that

$$\models \{p\}f\{q\}, \models \{q\}g\{r\} \Rightarrow p \models PC(f \bullet g, r) \text{ and}$$
$$\models \{r \wedge p\}f\{p\} \Rightarrow p \models PC(WH(r,f), \neg r \wedge p).$$

Let us prove that $\models \{p\}f\{q\}, \models \{q\}g\{r\} \Rightarrow p \models PC(f \bullet g, r)$.

This means that

$$p^T \cap q^{-F,f} = \varnothing; q^T \cap r^{-F,g} = \varnothing \Rightarrow p^T \cap r^{-F,f\bullet g} = \varnothing.$$

Indeed, $r^{-F,f\bullet g} = f^{-1}[r^{-F,g}]$. Since $q^T \cap r^{-F,g} = \varnothing$ and q is total, we have that $r^{-F,g} \subseteq q^F$. And since $p^T \cap q^{-F,f} = \varnothing$ we obtain that $p^T \cap r^{-F,f\bullet g} = \varnothing$.

Let us prove that $\models \{r \wedge p\} f \{p\} \Rightarrow p \models PC(WH(r,f), \neg r \wedge p)$.

Using the definition of validity we have: $(r \wedge p)^T \cap p^{-F,f} = \varnothing$ and $p^T \cap PC(WH(r,f), \neg r \wedge p)^F \neq \varnothing$.

By definition of *PC*,

$$p^T \cap PC(WH(r,f), \neg r \wedge p)^F = p^T \cap (\neg r \wedge p)^{-F, WH(r,f)}.$$

Let *d* be any data such that $p(d) \downarrow = T$ and $WH(r,f)(d) \downarrow$. Then there exists a sequence $d_0, d_1, \ldots, d_n$ such that $d_0 = d$, $f(d_0) \downarrow = d_1$, …, $f(d_{n-1}) \downarrow = d_n$, $r(d_0) \downarrow = T$, … , $r(d_{n-1}) \downarrow = T$, $r(d_n) \downarrow = F$. Also, $WH(r,f)(d) \downarrow = d_n$ and $p(d) \downarrow = T$. This gives $(r \wedge p)(d_0) \downarrow = T$. Also, $f(d_0) \downarrow = d_1$, thus $p(d_1) \downarrow = T$. With $r(d_1) \downarrow = T$ and $f(d_1) \downarrow = d_2$ we obtain $p(d_2) \downarrow = T$. By induction we have $p(d_n) \downarrow = p(WH(r,f)(d)) \downarrow = T$ and $r(d_n) \downarrow = r(WH(r,f)(d)) \downarrow = F$. Thus, $d \in (\neg r \wedge p)^{-T, WH(r,f)}$. Therefore $p^T \cap (\neg r \wedge p)^{-F, WH(r,f)} = \varnothing$ and consequently $p \models PC(WH(r,f), \neg r \wedge p)$.

4.4 Simpler Constraints for Partial Predicate Inference System

The trifling constraints introduced for rules *R_SEQ'* and *R_WH'* of PFHL in some cases can be changed to more stronger but simpler constraints. Such simpler constraints considered here stem from the following observation for properties of assertion validity for total predicates. In this case $\models \{p\} f \{q\}$ implies $p^{T,f} \subseteq q^T$, $q^F \subseteq p^{F,f}$ and, dually, $p^T \subseteq q^{-T,f}$, $q^{-F,f} \subseteq p^F$ because $\models \{p\} f \{q\}$ means that $p^{T,f} \cap q^F = \varnothing$ and predicates are total.

In terms of special consequence relations these properties can be reformulated as $p \models_T PC(f,q)$, $PC(f,q) \models_F p$.

Using these properties we can strengthen constraints for *R_SEQ'* and *R_WH'*.

Theorem 6. For PFHL the following properties hold:

1. $\models \{p\} f \{q\}, \models \{q\} g \{r\}, p \models_T PC(f,q) \Rightarrow p \models PC(f \bullet g, r)$,
2. $\models \{p\} f \{q\}, \models \{q\} g \{r\}, PC(f,q) \models_F p \Rightarrow p \models PC(f \bullet g, r)$,
3. $\models \{p\} f \{q\}, \models \{q\} g \{r\}, q \models_F PC(g,r) \Rightarrow p \models PC(f \bullet g, r)$,
4. $\models \{p\} f \{q\}, \models \{q\} g \{r\}, PC(g,r) \models_F q \Rightarrow p \models PC(f \bullet g, r)$,
5. $\models \{r \wedge p\} f \{p\}, PC(f,p) \models_T (r \wedge p) \Rightarrow p \models PC(WH(r,f), \neg r \wedge p)$,
6. $\models \{r \wedge p\} f \{p\}, (r \wedge p) \models_F PC(f,p) \Rightarrow p \models PC(WH(r,f), \neg r \wedge p)$.

To prove the first property recall, that $\models\{p\}f\{q\}$ means $p^T \cap q^{-F,f} = \varnothing$, $\models\{q\}g\{r\}$ means $q^T \cap r^{-F,g} = \varnothing$, $p \models_T PC(f,q)$ means $p^T \subseteq q^{-T,f}$, and $p \models PC(f \bullet g, r)$ means $p^T \cap r^{-F,f\bullet g} = \varnothing$. Thus, we should prove

$$p^T \cap q^{-F,f} = \varnothing,\ q^T \cap r^{-F,g} = \varnothing,\ p^T \subseteq q^{-T,f} \Rightarrow p^T \cap r^{-F,f\bullet g} = \varnothing .$$

Let d be any data such that $p(d)\downarrow= T,\ f\bullet g(d)\downarrow,\ r(f\bullet g(d))\downarrow$. By the first premise $q(f(d))\downarrow= T$. By the second premise $r(f\bullet g(d))\downarrow= T$. Thus, $d \notin r^{-F,f\bullet g}$. Therefore $p^T \cap r^{-F,f\bullet g} = \varnothing$.

Other properties related with *R_SEQ'* are proved in the same manner.

Consider properties related with *R_WH'*.

Property

$$\models\{r\wedge p\}f\{p\}, PC(f,p) \models_T (r\wedge p) \Rightarrow p \models PC(WH(r,f), \neg r \wedge p)$$

can be represented as

$$(r\wedge p)^T \cap p^{-F,f} = \varnothing,\ p^{-T,f} \subseteq (r\wedge p)^T,\ p^T \cap (\neg r\wedge p)^{-F,WH(r,f)} = \varnothing .$$

Let d be any data such that

$$p(d)\downarrow= T,\ WH(r,f)(d)\downarrow, (\neg r\wedge p)(WH(r,f)(d))\downarrow .$$

By the definition of the loop composition we have that there exists a sequence $d_0, d_1, \ldots d_n$ such that $d_0 = d$, $f(d_0)\downarrow= d_1$, …, $f(d_{n-1})\downarrow= d_n$, and $r(d_1)\downarrow= T$, … , $r(d_{n-1})\downarrow= T$, $r(d_n)\downarrow= F$. By induction on n taking into consideration the second premise we get that $p(d_0)\downarrow= T$, $p(d_1)\downarrow= T$, …, $p(d_n)\downarrow= T$. That means that $(\neg r\wedge p)(d_n)\downarrow= T$. Therefore $p^T \cap (\neg r\wedge p)^{-F,WH(r,f)} = \varnothing$.

Another property related with *R_WH'* is proved in the same manner.

This theorem permits to consider

$$p \models_T PC(f,q),\ PC(f,q) \models_F p,\ q \models_F PC(g,r),\ PC(g,r) \models_F q$$

(or any their combination) as constraints for *R_SEQ'* and

$$PC(f,p) \models_T (r\wedge p),\ (r\wedge p) \models_F PC(f,p)$$

as constraints for *R_WH'*. These constraints are simpler than initial trifling constraints.

We can go further trying to identify cases in which these constraints hold automatically. In other words to find cases in which the pure part of the inference rules can be used in derivation without proving validity of constraints.

One of such cases is described by the following definitions.

Assertion $\{p\}f\{q\}$ is called *T-increasing* if $p \models_T PC(f,q)$ holds, and *F-decreasing* if $PC(f,q) \models_F p$ holds.

Theorem 7. Let assertion $\{p\}f\{q\}$ be T-increasing or F-decreasing. Then $\models \{p\}f\{q\}$.

Consider the case $p \models_T PC(f,q)$. It means that $p^T \subseteq q^{-T,f}$ therefore $p^T \cap q^{-F,f} = \varnothing$. Other cases are considered in the same manner.

Theorem 8. All pure (with constraints omitted) inference rules of PFHL except rule *R_CONS'* (Table 7) preserve the classes of T-increasing and F-decreasing assertions.

Proofs for both properties is similar, thus consider the class of T-increasing assertion.

1. For *R_AS'* the proof that $\{S_P^x(p,h)\}AS^x(h)\{p\}$ is T-increasing can be easily obtained from the proof of the corresponding item of theorem 4.
2. For *R_ID'* the proof is obvious.
3. For *R_SEQ'* we should prove

$$p \models_T PC(f,q),\ q \models_T PC(g,r) \Rightarrow\ p \models_T PC(f \bullet g, r).$$

This means $p^T \subseteq q^{-T,f}, q^T \subseteq r^{-T,g} \Rightarrow p^T \subseteq r^{-T,f\bullet g}$. The proof of this fact is trivial.

4. For *R_IF'* we need to prove

$$r \wedge p \models_T PC(f,q),\ \neg r \wedge p \models_T PC(g,q) \Rightarrow\ p \models_T PC(IF(r,f,g),q).$$

This means $(r \wedge p)^T \subseteq q^{-T,f}$, $(\neg r \wedge p)^T \subseteq q^{-T,g} \Rightarrow p^T \subseteq q^{-T,\,IF(r,f,g)}$. Let d be any data such that $p(d)\downarrow = T$ and $IF(r,f,g)(d)\downarrow$. If $r(d)\downarrow = T$ then $r(f(d))\downarrow = T$ by the first premise; if $r(d)\downarrow = F$ then $r(\mathrm{g}(d))\downarrow = T$ by the second premise. Therefore $d \in q^{-T,\,IF(r,f,g)}$ and $p \models_T PC(IF(r,f,g),q)$.

5. For *R_WH'* we need to prove

$$r \wedge p \models_T PC(f,p) \Rightarrow\ p \models_T PC(WH(r,f),p).$$

From this point the proof coincides with the corresponding part of the proof of theorem 5 therefore it is omitted. So, we can conclude that $p \models_T PC(WH(r,f))$.

The theorem is proved.

As to rule *R_CONS'* we can change it to rule *R_CONS''* with the following new constraint:

$$p \models_T p' \text{ and } q' \models_T q.$$

It is easy to prove that the rule *R_CONS''* with this constraint is sound, and being restricted on the class of total predicates it is reduced to the rule *R_CONS*.

Theorem 9. Rule *R_CONS''* preserves the class of T-increasing assertions.

The proof is obvious.

The proved theorems permit to write Table 8 for simple **WHILE** inference system which is valid and is an extension of the inference system given in Table 2. In the new system only rule *R_CONS''* has a constraint. Simplicity of this system is explained by the fact that rules *R_AS'* and *R_SKIP'* (being axioms) specify T-increasing assertions and the constraint of *R_CONS''* is simple sufficient constraint (though it is rather expressive being an extension of *R_cons*).

Table 8. Simple PFHL **i**nference system for **WHILE** with T-increasing assertions

$\{S_P^x(p,h)\}AS^x(h)\{p\}$	*R_AS*
$\{p\}id\{p\}$	*R_SKIP*
$\dfrac{\{p\}f\{q\},\{q\}g\{r\}}{\{p\}f \bullet g\{r\}}$	*R_SEQ*
$\dfrac{\{r \wedge p\}f\{q\},\{\neg r \wedge p\}g\{q\}}{\{p\}IF(r,f,g)\{q\}}$	*R_IF*
$\dfrac{\{r \wedge p\}f\{p\}}{\{p\}WH(r,f)\{\neg r \wedge p\}},$	*R_WH*
$\dfrac{\{p'\}f\{q'\}}{\{p\}f\{q\}}, p \models_T p', q' \models_T q$	*R_CONS''*

Identification and investigation of other simple inference systems should be continued. One of such cases is induced by acyclic programs.

4.5 PFHL for Acyclic Programs

If we consider acyclic programs (loop-free programs), the preimage predicate transformer composition can easily be presented via formulas of predicate logic. This simplifies constraints and reduces the problem of their validity to the validity problem of formulas of composition-nominative predicate logics. These problems were investigated in [3, 5, 7]. For the cyclic programs, their acyclic approximations can be considered. Details are not presented here.

5 Related Work

The seminal work on a logical characterization of programs by Floyd [1] and Hoare [2] was purely axiomatic, i.e., not yet backed by a formal semantics of programs. While also Dijkstra followed this tradition with his weakest precondition calculus [8], he also systematically investigated the necessary properties of his predicate transformer “wp”. In particular, he explicitly required its *monotonicity* and realized (after a hint of J.C. Reynolds) the importance of its *continuity* for expressing effectively implementable calculations (by ruling out unbounded nondeterminism).

The crucial importance of monotonicity and continuity of functions for the set-theoretic modeling of programs was exhibited by Scott’s and Strachey’s denotational semantics where unbounded repetition is modeled as the fixed point of a continuous functional [9,10]. Similar considerations of monotonicity and continuity play a role in those approaches to program semantics that are based on the formal representation of programs as state relations (predicates), e.g. Back’s and White’s Refinement Calculus [11], Hoare’s and He’s Unifying Theory of Programming [12] and Boute’s Calculational Semantics [13]. However, this work was typically performed in a context where functions and predicates were basically assumed to be total, i.e., well-defined for all kinds of arguments (apart from the result of infinite loops which is usually represented by a special “non-termination” value).

From a logical perspective, *partial predicates* [14] give rise to *three-valued logics* where the additional value may represent “unknown” or “error”. Depending on the exact interpretation of this additional value, numerous variants of such logics have been developed by Łukasiewicz [15], Kleene [16], Bochvar [17] and others, see e.g. Bergmann [18] for a survey. Moisil [19] provided by the “Łukasiewicz-Moisil Algebras” an axiomatic algebraic framework for their formalization. A particular interest in these non-standard logics arose in the context of the theory of computation (McCarthy [20]), the modeling of processes (Bergstra and Ponse [21]), and in particular in the formal specification and verification of computer programs (Blikle [22], Konikowska et al [23]).

Especially in the context of the algebraic specification of abstract datatypes [24], the handling of *partial functions* (whose execution may not terminate or yield an error) plays an important role [25, 26]. Within a classical framework these may be handled by explicitly restricting the domain of a partial function by a predicate and treat the function result for arguments outside the domain as a definite (but unknown) value in the range of the function; this was also the basis of the work of one of the author’s of the present chapter [27, 28].

On the other hand, one may also introduce explicit support for partial functions within the logic itself such as in the Vienna Development Method (VDM) which introduces a corresponding “logic of partial functions” [29]. Broy and Wirsing devised in the CIP project the concept of “partial algebras” [30] where each carrier may contain unacceptable values (e.g. “undefined”) where special rules are given to deal with the application of functions to unacceptable elements; thus even non-strict functions may be specified that produce acceptable results for unacceptable arguments. This concept has become the basis of a lot of subsequent work [31–33] and also forms the semantic basis of the “Common Algebraic Specification Language” CASL [34].

6 Conclusions

In the chapter we have considered questions concerning extension of traditional Floyd-Hoare logic for partial pre- and postconditions. We have adopted a semantic-syntactic style of logic definition. Therefore we first have constructed and investigated special program algebras of partial predicates, functions, and programs. In such algebras program correctness assertions can be presented with the help of a special composition called Floyd-Hoare composition. We have proved that this composition is monotone and continuous. Considering the class of constructed algebras as a semantic base we then have defined an extended logic – Partial Floyd-Hoare Logic – and investigated its properties. This logic has rather complicated soundness constraints for inference rules, therefore somewhat simpler but also sufficient constraints have been proposed. The logics constructed can be used for program verification.

This chapter can be considered as a first step in developing composition-nominative program logics. The major directions of further investigation are the question of relative completeness of the system of inference rules, invariants for cycles, and types for variables and functions. Also the authors plan to construct a prototype of a program reasoning system oriented on the constructed logics.

References

1. Floyd, R.W.: Assigning meanings to programs. Proceedings of the American Mathematical Society Symposia on Applied Mathematics 19, 19–31 (1967)
2. Hoare, C.A.R.: An axiomatic basis for computer programming. Communications of the ACM (12), 576–580 (1969)
3. Nikitchenko, M.S., Shkilniak, S.S.: Mathematical logic and theory of algorithms. Publishing house of Taras Shevchenko National University of Kyiv, Kyiv (2008) (in Ukrainian)
4. Nielson, H.R., Nielson, F.: Semantics with Applications: A Formal Introduction, p. 240. John Wiley & Sons Inc. (1992)
5. Nikitchenko, M.S., Tymofieiev, V.G.: Satisfiability in Composition-Nominative Logics. Central European Journal of Computer Science 2(3), 194–213 (2012)
6. Avron, A., Zamansky, A.: Non-Deterministic Semantics for Logical Systems. Handbook of Philosophical Logic 16, 227–304 (2011)
7. Nikitchenko, M., Tymofieiev, V.: Satisfiability and Validity Problems in Many-sorted Composition-Nominative Pure Predicate Logics. In: Ermolayev, V., Mayr, H.C., Nikitchenko, M., Spivakovsky, A., Zholtkevych, G. (eds.) ICTERI 2012. CCIS, vol. 347, pp. 89–110. Springer, Heidelberg (2013)
8. Dijkstra, E.W.: A Discipline of Programming. Prentice-Hall, Englewood Cliffs (1976)
9. Schmidt, D.A.: Denotational Semantics – A Methodology for Language Development. Allyn and Bacon, Boston (1986)
10. Scott, D., Strachey, C.: Towards a Mathematical Semantics for Computer Languages. In: Proc. Symp. on Computers and Automata, Polytechnic Institute of Brooklyn; also Tech. Mon. PRG-6, pp. 19–46. Oxford U. Computing Lab. (1971)
11. Back, R.-J., von Wright, J.: Refinement Calculus: A Systematic Introduction. Springer, New York (1998)
12. Hoare, C.A.R., He, J.: Unifying Theories of Programming. Prentice Hall, London (1998)
13. Boute, R.T.: Calculational Semantics: Deriving Programming Theories from Equations by Functional Predicate Calculus. ACM Transactions on Programming Languages and Systems 28(4), 747–793 (2006)
14. Wang, H.: The Calculus of Partial Predicates and Its Extension to Set Theory. Zeitschr. F. Math. Logik und Grundlagen D. Math. 7, 283–288 (1961)
15. Łukasiewicz, J.: O logice trójwartościowej. In: Borkowski, L. (ed.) Ruch Filozoficzny 5:170–171. English Translation: On Three-Valued Logic. Selected works by Jan Łu-kasiewicz, pp. 87–88. North–Holland, Amsterdam (1970)
16. Kleene, S.C.: On Notation for Ordinal Numbers. Journal Symbolic Logic 3, 150–155 (1938)
17. Bochvar, D.A.: On a 3-valued Logical Calculus and its Application to the Analysis of Contradictions. Matematiceskij Sbornik 4, 287–308 (1939) (in Russian)
18. Bergmann, M.: An Introduction to Many-Valued and Fuzzy Logic: Semantics, Algebras, and Derivation Systems. Cambridge University Press, Cambridge (2008)

19. Moisil, G.: Recherches sur les logiques nonchrysippiennes. Ann. Sci. Univ. Jassy 26, 431–436 (1940)
20. McCarthy, J.: A Basis for a Mathematical Theory of Computation. In: Braffort, P., Hirshberg, D. (eds.) Computer Programming and Formal Systems, pp. 33–70. North-Holland, Amsterdam (1963)
21. Bergstra, J.A., Ponse, A.: Bochvar-McCarthy Logic and Process Algebra. Notre Dame Journal of Formal Logic 39(4), 464–484 (1988)
22. Blikle, A.: Three-Valued Predicates for Software Specification and Validation. In: Bloomfield, R.E., Jones, R.B., Marshall, L.S. (eds.) VDM 1988. LNCS, vol. 328, pp. 243–266. Springer, Heidelberg (1988)
23. Konikowska, B., Tarlecki, A., Blikle, A.: A Three-valued Logic for Software Specification and Validation. Fundam. Inform. 14(4), 411–453 (1991)
24. Sannella, D., Tarlecki, A.: Foundations of Algebraic Specification and Formal Software Development. Monographs in Theoretical Computer Science. Springer (2012)
25. Cheng, J.H., Jones, C.B.: On the Usability of Logics which Handle Partial Functions. In: Morgan, C., Woodcock, J.C.P. (eds.) 3rd Refinement Workshop, pp. 51–69 (1991)
26. Jones, C.B.: Reasoning about Partial Functions in the Formal Development of Programs. Electronic Notes in Theoretical Computer Science 145, 3–25 (2006)
27. Schreiner, W.: Computer-Assisted Program Reasoning Based on a Relational Semantics of Programs. In: Quaresma, P., Back, R.-J. (eds.) Proceedings First Workshop on CTP Components for Educational Software (THedu 2011), Wrocław, Poland. Electronic Proceedings in Theoretical Computer Science (EPTCS), vol. 79, pp. 124–142 (July 31, 2012) ISSN: 2075-2180
28. Schreiner, W.: A Program Calculus Technical Report. Research Institute for Symbolic Computation (RISC), Johannes Kepler University, Linz, Austria (2008), `http://www.risc.uni-linz.ac.at/people/schreine/papers/ProgramCalculus2008.pdf`
29. Jones, C.B., Middelburg, C.A.: A Typed Logic of Partial Functions Reconstructed Classically. Acta Informatica 31(5), 399–430 (1994)
30. Broy, M., Wirsing, M.: Partial Abstract Data Types. Acta Informatica 18(1), 47–64 (1982)
31. Burmeister, P.: A Model Theoretic Oriented Approach to Partial Algebras. Akademie-Verlag (1986)
32. Kreowski, H.-J.: Partial Algebra Flows from Algebraic Specifications. 14th Int. Colloquium on Automata, Languages and Programming. In: Ottmann, T. (ed.) ICALP 1987. LNCS, vol. 267, pp. 521–530. Springer, Heidelberg (1987)
33. Reichel, H.: Initial Computability, Algebraic Specifications, and Partial Algebras. Oxford University Press (1987)
34. Mosses, P.D. (ed.): CASL Reference Manual: The Complete Documentation of the Common Algebraic Specification Language. LNCS, vol. 2960. Springer, Heidelberg (2004)

Author Index