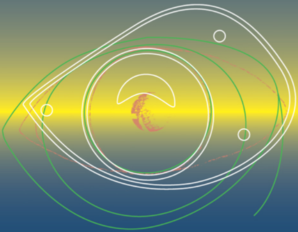
KI 2009

32nd Annual Conference on Artificial Intelligence

Paderborn, Germany
September 15 – 18, 2009



Workshop Proceedings

Klaus-Dieter Althoff, Kerstin Bach & Meike Reichle (Eds.)

Preface

We are pleased to present the Workshop Proceedings of the 32^{nd} Annual Conference on Artificial Intelligence (KI 2009), which is held on September 15-18 in Paderborn.

This year the volume includes papers or abstracts of ten workshops: 3^{rd} Workshop on Behavior Monitoring and Interpretation - Well Being, Complex Cognition, 1^{st} International Workshop on Distributed Computing in Ambient Environments, 4^{th} Workshop on Emotion and Computing - Current Research and Future Impact, 5^{th} Workshop on Knowledge Engineering and Software Engineering, Human-Machine-Interaction, Machine Learning in Real-time Applications, 23^{rd} Workshop on Planning, Scheduling, Design, and Configuration, Relational Approaches to Knowledge Representation and Learning, and Self-X in Mechatronics and other Engineering Applications.

Our thanks for these valuable contributions go to the workshop chairs Hamid Aghajan, M. Zaheer Aziz, Joachim Baumeister, Christoph Beierle, André Brinkmann, Stefan Edelkamp, Heinz-Josef Eikerling, Björn Gottfried, Benjamin Klöpper, Gabriele Kern-Isberner, Markus Knauff, Volker Lohweg, G. J. Nalepa, Oliver Niggemann, Marco Ragni, Dirk Reichardt, Jürgen Sauer, Bernd Schattenberg, Ute Schmid, Gerhard Rigoll, and Frank Wallhoff. Of course, we would also like to thank all the presenters and authors of the workshop papers as well as the members of the various program committees.

We very much appreciate the support of the chair of this year's conference Bärbel Mertsching and her organizing team - especially Christine Harms, Marcus Hund, and Zaheer Aziz - and value the opportunity to organize the workshops.

We sincerely hope that the participants enjoyed this year's workshop program and that this collection of papers will inspire and encourage more AI-related research in the future.

Hildesheim, September 2009

Klaus-Dieter Althoff,
Workshop Chair

Kerstin Bach & Meike Reichle,
Co-Editors

Table of Contents

WS 1: Behavior Monitoring and Interpretation - Well Being

WS 2: Complex Cognition

WS 3: Distributed Computing in Ambient Environments (DiComAe) 2009

WS 4: Emotion and Computing

WS 5: Knowledge Engineering and Software Engineering

WS 6: Human-Machine-Interaction

WS 7: Machine Learning in Real-time Applications (MLRTA 09)

WS 8: Planning, Scheduling, Design, and Configuration (PuK2009)

WS 9: Relational Approaches to Knowledge Representation and Learning

WS 11: Self-X in Mechatronics and other Engineering Applications

The 32nd Annual Conference on
Artificial Intelligence (KI-2009)
September 15-18, 2009, Paderborn, Germany

Workshop on

Behavior Monitoring and Interpretation - Well Being

http://www.tzi.de/ bjoerng/BMI-KI-09

Björn Gottfried & Hamid Aghajan

KI 2009

AI And Automation

Workshop Organizers and Co-Chairs

Björn Gottfried	Centre for Computing Technologies, University of Bremen, Germany bg@tzi.de
Hamid Aghajan	Wireless Sensor Networks Laboratory, Stanford University, CA, USA aghajan@stanford.edu

Program Committee

Timothy D. Adlam	University of Bath, UK
Stephen Balakirsky	National Institute of Standards and Technology, USA
Matt Duckham	University of Melbourne, Australia
Sylvain Giroux	University of Sherbrooke, Canada
Thomas Kirste	University of Rostock, Germany
Karin Klabunde	Philips Research, The Netherlands
Joyca Lacroix	Philips Research, The Netherlands
Paul McCullagh	University of Ulster at Jordanstown, UK
Ubbo Visser	University of Miami, USA
Howard D. Wactlar	Carnegie Mellon University, USA

Preface

Björn Gottfried[1] and Hamid Aghajan[2]

[1] Centre for Computing Technologies
University of Bremen
bg@tzi.de
[2] Wireless Sensor Networks Lab
Stanford University
aghajan@stanford.edu

The third workshop on Behaviour Monitoring and Interpretation (BMI) is held in conjunction with the 32nd German Conference on Artificial Intelligence, in Paderborn, on September 15, 2008. The collection of submitted papers covers a broad spectrum of issues concerned with behaviour monitoring and interpretation in general and with the thematic focus section on Well Being in particular.

Monitoring what happens in the environment, what people do and how they interact with their surroundings is of interest in several areas, such as in ambient intelligence, health care applications, or mobile services. This workshop focuses on methods analyzing and interpreting the behaviour of individuals, or of small groups of people. This is for the purpose of intention recognition, triggering of smart home environment services, life routine logging, or generally for the investigation of how humans deal with specific problems in their everyday life.

While technological advances in sensing and processing have ushered in an unprecedented opportunity for realizing behaviour monitoring applications, much effort remains needed for the development of methods to integrate and exploit the available data for addressing specific applications. In addition to the general BMI topic, part of this year's workshop features a thematic focus section on Well Being. While including areas such as ambient assisted living, Well Being is more general and captures different interfaces and methods that advance modern living. Techniques and approaches in formulating and addressing application needs in Well Being will be presented and discussed.

Björn Gottfried
Hamid Aghajan

Behaviour in Physical Information Spacetime: Present-Directed Intention and Its Local Recognition

Stefan Artmann

Frege Centre for Structural Sciences
Friedrich Schiller University
Jena, Germany
stefan.artmann@uni-jena.de

The talk introduces the concept of physical information spacetime to capture the medium of interaction between intelligent agents as conceived by strong Pervasive Computing. The representation of behaviour of such agents involves the concept of intention. An intelligent agent that interacts with another one in physical information spacetime, must have at least some components of this concept at its disposal in order to represent its own and the other's behaviour. The problem of intention recognition is, thus, a central problem of strong Pervasive Computing, so that general prerequisites of its solution ought to be described precisely. In the case of small-scale engineering of physical information spacetime, present-directed intentions must be recognized locally. Three postulates are discussed that must be accepted if intention recognition in a minimum scenario of one agent trying to recognize the intention of another one, shall be successful: the postulate of data structures common to both agents, the postulate of continuity between intentional and intended actions, and the postulate of distributed algorithms for acting with shared intentions.

Abnormal Behavior Classification and Alerting through Detection, Identification, Prediction (DIPR) System Applied to a Multi-Camera Network

Deborah Goshorn[1], Rachel Goshorn[2], Joshua Goshorn[3], Lawrence Goshorn[4]

[1] CSE Department, UC San Diego and and MOVES Institute, Naval Postgraduate School, Monterey, Ca., USA
[2] Department of Sys. Eng., Naval Postgraduate School Monterey, Ca., USA
[3] ECE Department, UC San Diego and JLG Tech., Inc, Pebble Beach, Ca., USA
[4] JLG Technologies, Inc, Pebble Beach, Ca., USA

Cameras are becoming part of our everyday life, whether, for example on a cell phone, a security system, or an airport. The world is also installing multi-camera systems for monitoring; these systems tend to be monitored through "human intelligence". In order to install these multi-camera systems and automate "human intelligence", multi-camera systems deployed to monitor large-scale areas, requires an artificial intelligence (AI) systems approach where the amount of data/information to monitor, what to monitor, and the outcome of the monitored data must be defined up front, therefore yielding a AI systems engineering problem. In this paper, a multi-camera system is developed and implemented for the purpose of detecting abnormal behavior and provide corresponding alarms, and in parallel provides an automatic mustering system for "personnel check-in" (e.g. for a university class or a work environment). The AI software system to automate the "human intelligence" is broken up into four software subsystems of Detection, Identification, Prediction, and Reaction (DIPR), creating the DIPR system. The DIPR system is superimposed onto the infrastructure of networks, multiple cameras, computer nodes, server, etc. In this DIPR system, the raw sensor data comes from about thirty IP cameras, where up to six IP cameras connect to one PC (Node), where the system has six PCs (Nodes); the Detection subsystem is carried out at each PC (Node), where the input into Detection is the raw video data and the output is the personnel features. The Identification subsystem forms intelligent states (symbols) based on intelligence rules fusing features to select the camera with the best field of view per person at each time instance. The Prediction subsystem forms sequences of symbols, classifies these symbols into a behavior (based on a syntactical sequential behavior classifier) and then infers a predicted behavior outcome. The "Reaction" subsystem carries out actions; the main action carried out is to alarm if an abnormal behavior is detected; if a normal behavior is detected, no action is recommended. This paper will demonstrate the infrastructure designed and the DIPR System superimposed onto the infrastructure for the purpose of abnormal behavior detection and alarms, and in parallel an automatic mustering.

Towards an Intelligent Agent Framework to Manage and Coordinate Collaborative Care

Christian Guttmann[1], Michael Georgeff[1], Kumari Wickramasinghe[1], Ian Thomas[1], Hamid Gharib[2], Simon Thompson[2], and Heinz Schmidt[3]

[1] Faculty of Med., Nursing and Health Sci., Monash University, Melbourne, Australia
[2] British Telecom Research, Adastral Park, Ipswich, United Kingdom
[3] School of Computer Science and Information Tech., RMIT University, Australia
christian.guttmann@med.monash.edu.au

Our project investigates computational models of collaboration between entities involved in the coordination and management of complex, consumer care provision systems. We use the agent paradigm to assist service providers in the coordination and decision processes that enable the management of patient care. One important aspect is to adapt dynamically to changing patient situations, service provider characteristics and changing quality requirements. This paper discusses how we have built a Multi-Agent System to (a) plan the delivery of healthcare services, (b) negotiate the responsibility of healthcare providers to provide services (reaching agreements), and (c) ensure that the selected team of providers and patients adhere to careplans (maintaining agreements). Coordination of healthcare services is a powerful example of a complex domain requiring models of collaborating agents that make agreements and adhere to them. The scope of our project is on investigating settings where collaborators are unreliable and non-conformant, i.e., where agreements are made but are not always maintained. This issue not only arises in collaborative healthcare management, but also in many other service industries, including the telecommunications and electricity supply. In the healthcare domain, better knowledge of patients and their needs can lead to dramatic improvements in the quality, safety, and efficiency of care provision, particularly in relation to patients suffering from chronic disease. However, a realisation of these benefits requires an adequate technological infrastructure. This infrastructure must support effective monitoring, analysis, distribution and utilisation of this knowledge. To date, there has been little analysis of such healthcare management systems from an information-theoretic viewpoint. Despite the similarities of care management in both healthcare and commercial domains, current research has not offered a unified model of consumer care or determined how the lessons and technologies used in the one domain are transferable to the other. We report on building software architectures, knowledge representations, and machine reasoning techniques that enable consumer care provision across various organisational structures and settings.

‡ This project is supported in part by ARC Linkage Grant LP0774944 and British Telecom Agreement CT1080050530. We wish to thank Dr Kay Jones, Professor Leon Piterman, Professor Peter Shattner and Mr Akuh Adaji for their advice and contribution to the project.

Autonomous Learning of User's Preference of Music and Light Services in Smart Home Applications

Amir Hossein Khalili, Chen Wu, and Hamid Aghajan

Wireless Sensor Networks Laboratory
Stanford University
Stanford, CA, USA
{akhalili|chenwu|aghajan}@stanford.edu

There has been extensive research on smart homes with a variety of applications including monitoring systems for independent living of elderly and accident detection. In most current applications, the system provides its services based on a set of embedded fixed rules. This results in a rigid operation which may not be desirable for many users because different users have different behavior and habits. When an ambient intelligence system is designed for a smart home environment, such variety in the user activity patterns and preferences in environment settings has to be taken into account for the system to adapt to and satisfy the user. Automatic learning of the behavior and preferences of the user is essential for automatic regulation of services such as lighting and music in order for the system to be useful and practical.

In our approach, the system gains intelligence through observing the user, interacting with the user, and exploring the user's interests via mutual discovery mechanisms. A "sensory-motor" module monitors the activity and location of the user at home, and activates a "decision-maker" module in case of a change or when a time slot lapses. According to the state of the environment (which includes the user's current activity, location and current time), "decision-maker" learns and selects the satisfactory features of a service and makes "actor" to take the desired action with the objective of maximizing the user satisfaction. Rewards and penalties are received either in an explicit form through a handheld device or in an implicit form by observing the user's feedback such as stopping a music which is recognized as a penalty.

We use the Temporal Differential (TD) class of reinforcement learning for the "decision-maker" module since TD, which is an unsupervised learning method, does not require a predefined model and continuously learns the best action for each state through interactions with the user. The interaction is in a trial-and-error form to increase an objective function of contradictive goals including the user's comfort and the minimized total cost function defined explicitly or implicitly as penalty. We demonstrate the effect of each parameter of TD in quality and speed of learning and inference, even in case of uncertainty in the user's feedback or when the user changes his preferences over time.

Analysis and Prediction of Daily Physical Activity Level Data Using Autoregressive Integrated Moving Average Models

Xi Long [a,c], Steffen Pauws [a], Marten Pijl [a], Joyca Lacroix [a], Annelies Goris [b], and Ronald M. Aarts [a,c]

[a] User Experiences Group, Philips Research Laboratories, Eindhoven, The Netherlands
[b] Philips New Wellness Solutions, Eindhoven, The Netherlands
[c] Department of Electrical Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands
{Xi.Long, Steffen.Pauws, Marten.Pijl, Joyca.Lacroix, Annelies.Goris, Ronald.M.Aarts}@philips.com

Results are provided on predicting daily physical activity level (PAL) data from past data of participants of a physical activity lifestyle program aimed at promoting a healthier lifestyle consisting of more physical exercise. The PAL data quantifies the level of a person's daily physical activity and reflects the daily energy expenditure of this person. In this wellbeing program, a mobile body-worn activity monitor with a built-in triaxial accelerometer was used to record the PAL data of an individual for a period of 13 weeks. The autoregressive integrated moving average (ARIMA) models were employed to predict future PAL data of every next week. This paper proposes a categorized ARIMA prediction method which achieves a large reduction in computation time without significant loss in prediction accuracy compared with the traditional ARIMA. In the current method, PAL data were categorized as being stationary, trend or seasonal via assessing their autocorrelation functions. The most appropriate ARIMA model for these three categories was automatically selected by applying the objective penalty function criterion. The results show that our categorized ARIMA method performed well in terms of PAL prediction accuracy (~9% mean absolute percentage error), model parsimony and robustness.

Parsing correctly nested event sequences with linear time complexity

Hartmut Messerschmidt and Björn Gottfried

Centre for Computing Technologies
University of Bremen
28359 Bremen, Germany
{hmess|bg}@tzi.de

In recent years great advances have been made regarding the development of sensor technologies. With such technologies areas like *smart environments* and *ambient assisted living* evolved. Analysing event sequences occurring in both areas is of interest in planning, quality assurance, as well as behaviour interpretation. In this paper, a class of event sequences is identified which can be frequently observed and which are therefore of some importance. It is shown that these sequences can be parsed within linear time complexity.

One application field that could essentially benefit from sensor technologies are hospital environments; essentially, an increase of patient safety by means of the tracking of drug-treatment, in other words, for reducing medical errors.

A sensor rich environment, however, requires a lot of efforts to efficiently make use of the new data, in this case of *RFID* tags. Which objects should be tagged for which purposes? Where are *RFID* readers to be installed? As founds out: A typical 800-bed hospital administers approximately 15,000 doses of medication a day.

It is therefore the idea of the present paper to look for interesting patterns that arise when controlling drug-treatment and other businesses of nurses. According to the huge amounts of data that are generated throughout a day in a hospital, a particular focus of the presented research lies in methods that are as efficient as possible.

To determine the efficiency the tasks or patterns in the hospital environment are described and translated into formal concepts such as the interval relations by Allen and formal languages. These languages include deterministic context free languages such as the language of correctly nested brackets (Dyck language), as well as many mildly context sensitive languages such as cross reverences ($a^n b^m c^n d^m$) and repetitions ($a^n b^n c^n$). The latter are known not to be context free. Nevertheless the word problem for all these languages can be solved in linear time. To prove this, we present the so-called restarting automaton, and construct specific restarting automata accepting these languages. Different restricted variants of restarting automata are known to accept exactly the contextfree, the deterministic contextfree, and the Church-Rosser languages. For deterministic context free and Church-Rosser languages the word problem is decidable in linear time.

The paper finishes with a discussing about the possibilities and limitations of this approach together with future research topics and its relevance for BMI.

Make room for me

A study outline on a spatial and situational concept in HRI

Annika Peters, Thorsten P. Spexard, Petra Weiß and Marc Hanheide

Cognitive Interaction Technology Centre of Excellence,
Applied Informatics and Department of Linguistics and Literature,
Bielefeld University, Germany
{apeters}@techfak.uni-bielefeld.de

Mobile robots are already applied in factories and hospitals, merely to do a distinct task. It is envisioned that robots assist in households, soon. Those service robots will have to cope with several situations and tasks and of course with sophisticated Human-Robot Interaction (HRI).

Consider a narrow place, e.g. hallways, door frames or a small kitchen. A robot might block the way or drives towards you, pursuing its own goal like you. Humans do not even speak to each other in order to pass by and avoid bumping into each other even if the space is narrow.

The complexity for a social robot comprises not only the detection of a particular situation but also the distinction between several situations. These could be for example, a “passing by” situation, an “approaching”- or a “giving information” situation. Consequently, the detection of humans and collision avoiding is necessary but not enough, when a robot is modelled for assistance in households.

Social signals or unconscious cues are sent and received by interaction partners. These signals and cues influence the interaction partner. It is crucial to be aware of those implicit cues, when interaction between a robot and a human is modelled. The general goal is to use non-verbal communication, namely implicit body- and machine movements to make HRI smoother and simpler.

A robot should not only consider social rules with respect to proxemics and spatial prompting in communication or in approaching people. It should also be able to signal and understand spatial constraints. The spatial and situational constraint, which this paper focuses on, is “making room for each other” in a “passing by” scenario. It is important to consider both sides of an interaction – the human’s and the side of the robot, while modelling.

Therefore, a study is outlined and currently conducted to find out, which behaviour is the most appropriate avoiding strategy and how participants express their wish to pass by. Hence, the non-humanoid robot BIRON2 blocks the way in a corridor in the Bielefeld University and initiates a variety of defensive and more offensive passing strategies towards an approaching person. To track humans, the demonstrator is equipped with a person representation, which comprises faces, legs, sound directions.

The results of the study will be used to provide data under the focus of proxemics and spatial prompting and to subsequently improve the system, accordingly.

GPS Technology for Measuring Out-of-Home Mobility of Cognitively Impaired Elderly People: Preliminary Findings

Noam Shoval[1], Hans-Werner Wahl[2], Gail Auslander[3], Michal Isaacson[1], Frank Oswald[2], Tamar Edry[1], Ruth Landau[3] and Jeremia Heinik[4]

[1]Department of Geography
The Hebrew University of Jerusalem
Mount Scopus, Jerusalem 91905, Israel
{noamshoval|michal.isaacson|tamar.edry}@huji.ac.il

[2]Department of Psychological Aging Research
University of Heidelberg, Germany.
{hans-werner.wahl|frank.oswald}@psychologie.uni-heidelberg.de

[3]School of Social Work
The Hebrew University of Jerusalem
Mount Scopus, Jerusalem 91905, Israel
{msgail|mslanda}@pluto.mscc.huji.ac.il

[4]The Margoletz Psychogeriatric Center
Tel-Aviv Sourasky Medical Center, Israel
heinik@post.tau.ac.il

One of the behavioral manifestations of age-related cognitive impairment is a severe problem with out-of-home mobility. Until recently, the assessment of outdoor mobility relied on the reports of family caregivers and institutional staff, using observational approaches, activity monitoring, or behavioral checklists. In recent years, however, the rapid development and availability of small, inexpensive, and reliable tracking devices has led to a growing volume of research that uses this technology as a tool for data collection in various disciplines.

This paper presents findings based on 41 participants from the Tel-Aviv Metropolitan area (Israel), who took part in an interdisciplinary project, involving researchers in geography, social work, gerontology, psychology, and medicine, that studies the outdoor activities of elderly people in time and space using advanced tracking technologies.

Demented, mildly cognitive impaired and healthy men and women aged 64–90 were tracked for 28 consecutive days using a location kit that combined GPS with RFID technology. The high-resolution spatial and temporal data obtained made it possible to analyze the differences in the timing and distance of daily outdoor mobility patterns of participants with different levels of cognitive function.

The results of this analysis demonstrate that as part of the general decline in out-of-home mobility of elderly people with cognitive impairment, the spatial range of mobility of these people is severely compromised and they tend to spend most of their out-of-home time in close proximity to their residences.

Medical Knowledge Representation and Reasoning in the CHRONIOUS Project

Xin Xing and Hagen Langer

Center for Computing and Communication Technologies (TZI)
Universität Bremen, Germany
{xing|hlanger}@tzi.de

In this paper we will outline the function of ontological knowledge representation and reasoning within the CHRONIOUS[1] system and comparable platforms. CHRONIOUS, a project funded by the European Union under its 7th Framework Program, proposes an open, ubiquitous and adaptive chronic disease management platform. The primary goal of this project is to develop a smart wearable solution, based on multi-parametric sensor data processing, for continuous monitoring of patients suffering from chronic obstructive pulmonary disease (COPD) and chronic kidney disease (CKD). Both state-of-the-art environmental and activity sensors as well as diverse vital signs sensors are employed to constantly monitor patients' behavior and track their current health condition. If invalid medical data is detected or if current activity and behavior lay outside the predefined individual activity ranges and locomotion behavior, the system will generate alerts to inform the patient and/or the health-care professionals. In addition, monitoring services for drug intake and dietary habits, as well as post-hoc services for multi-lingual document search and medical guideline representation and execution are under consideration by the CHRONIOUS team.

In such environments different sources of information have to be used in an integrated manner in order to enable the desired complex functionality. The spectrum of heterogeneous information sources includes raw sensor data generated by the patient's wearable device, natural language documents (scientific publications on the patient's disease and medical guidelines), and semi-structured sources, such as electronic health records (EHR). The information must be represented language-independently and be formalized at the semantic level.

We propose a hierarchical knowledge representation system with three levels of ontologies to represent the various sources of information mentioned above: (1) A top-level ontology which provides general domain-independent concepts and roles, (2) biomedical ontologies represent general medical terms, anatomy, physiology, pharmacology, and other concepts, which are independent from the patient's particular disease, and (3) disease-specific ontologies for COPD and CKD. These three levels of the ontology form a *default inheritance* network, so that more specific information such as concepts defined in the disease-specific subontology, override the default specifications of the same concept defined at the upper levels. Although the reasoning *within* each subontology remains strictly monotonic, drawing inferences between different levels of the ontology, as described above, introduces a limited form of non-monotonic reasoning.

[1] Web site: *http://www.chronious.eu/* (visited: August 13, 2009).

The 32nd Annual Conference on
Artificial Intelligence (KI-2009)
September 15-18, 2009, Paderborn, Germany

Workshop on

Complex Cognition

http://www.cogsys.wiai.uni-bamberg.de/KI09WSCoCo/

Ute Schmid, Marco Ragni & Markus Knauff (Eds.)

KI 2009

AI And Automation

Workshop Organizers and Co-Chairs

Ute Schmid	University of Bamberg, Germany ute.schmid@uni-bamberg.de
Marco Ragni	University of Freiburg, Germany ragni@informatik.uni-freiburg.de
Markus Knauff	Justus Liebig University Gießen markus.knauff@psychol.uni-giessen.de

Program Committee

Berry Claus	Psycholinguistik, Universität des Saarlandes
Joachim Funke	Allgemeine/Theoretische Psychologie, Universität Heidelberg
Günther Görz	Künstliche Intelligenz, Universität Erlangen
Helmar Gust	Künstliche Intelligenz, Universität Osnabruck
Hilde Haider	Allgemeine Psychologie, Universität zu Köln
Christoph Hölscher	Kognitionswissenschaft, Universität Freiburg
Markus Knauff	Allgemeine Psychologie/Kognitionsforschung, Universität Gießen
Josef Krems	Allgemeine Psychologie und Arbeitspsychologie, TU Chemnitz
Marco Ragni	Kognitionswissenschaft, Universität Freiburg
Thomas Reineking	Kognitive Neuroinformatik, Universität Bremen
Christoph Schlieder	Kulturinformatik, Universität Bamberg
Ute Schmid	Kognitive Systeme, Universität Bamberg
Klaus Stein	Kulturinformatik, Universität Bamberg

Preface

Ute Schmid[1] and Marco Ragni[2] and Markus Knauff[3]

[1] Cognitive System Group, Faculty Information Systems and Applied Computer Science, University of Bamberg
ute.schmid@uni-bamberg.de
[2] Center for Cognitive Science, Institute of Computer Science and Social Research, University of Freiburg
ragni@cognition.uni-freiburg.de
[3] Experimental Psychology and Cognitive Science, University of Gießen
markus.knauff@psychol.uni-giessen.de

The KI'09 workshop *Complex Cognition* was a joint venture of the Cognition group of the Special Interest Group Artificial Intelligence of the German Computer Science Society (Gesellschaft für Informatik) and the German Cognitive Science Association.

Dealing with complexity has become one of the great challenges for modern information societies. To reason and decide, plan and act in complex domains is no longer limited to highly specialized professionals in restricted areas such as medical diagnosis, controlling technical processes, or serious game playing. Complexity has reached everyday life and affects people in such mundane activities as buying a train ticket, investing money, or connecting a home desktop to the internet.

Research in cognitive AI can contribute to support people navigating through the jungle of everyday reasoning, decision making, planning and acting by providing intelligent support technology. Lessons learned from expert system research of the nineteen-eighties are that the aim should not be to provide for fully automated systems which can solve specialized tasks autonomously but instead to develop interactive assistant systems where user and system work together by taking advantages of the respective strenghts of human and machine.

To accomplish a smooth collaboration between humans and intelligent systems, basic research in cognition is a necessary precondition. Insights in cognitive structures and processes underlying successful human reasoning and planning can provide suggestions for algorithm design. Even more important, insights in restrictions and typical errors and misconceptions of the cognitive systems provide information about that parts of a complex task from which the human should be relieved. For successful human-computer interaction in complex domains furthermore it has to be decided which information should be presented when in what way to the user.

We strongly believe that symbolic approaches of AI and psychological research of higher cognition are at the core of success for the endeavor to create intelligent assistant system for complex domains. While insight in the neurological processes of the brain and in the realization of basic processes of perception, attention and sensu-motoric coordination are important for the basic understanding of the basis of human intelligence, these processes have a much too fine granularity for the design and realization of intercative systems which must communicate with the user on knowledge level. If human system users should not be incapacitated by a system, system decisions must be transparent for the user and the system must be able to provide explanations for the reasons of its proposals and recommendations. Therefore, even when some of the underlying algorithms are based on statistical or neuronal approaches, the top-level of such systems must be symbolical and rule-based.

The papers presented at this workshop on complex cognition give an inspiring and promising overview of current work in the field which can provide first building stones for our endeavor to create knowledge level intelligent assistant systems for complex domains. The topics cover modeling basic cognitive processes, interfacing subsymbolic and symbolic representations, dealing with continuous time, Bayesian identification of problem solving strategies, linguistic inspired methods for assesing complex cognitive processes and complex domains such as recognition of sketches, predicting changes in stocks, spatial information processing, and coping with critical situations.

Ute Schmid
Marco Ragni
Markus Knauff

Accessing complex cognitive processes via linguistic protocol analysis

Thora Tenbrink and Linn Gralla

FB 10 Faculty of Linguistics and Literary Sciences, Universität Bremen, Germany
{tenbrink | gralla@uni-bremen.de}

Abstract. Complex cognitive processes are often investigated via elicitation of natural language data. While traditional psychological research typically focuses on the analysis and interpretation of content that is directly expressed in verbal reports, linguistic discourse analytic methods can contribute deeper insights into the processes involved, via highlighting linguistic structures and patterns that the speakers themselves may not be consciously aware of. In this paper, we first present the general method of "Cognitive Discourse Analysis", outlining its main features and analysis procedures in the light of requirements from cognitive science and artificial intelligence. In the second part we turn to a more detailed, exemplary presentation of a study of thought processes involved in object assembly. A process model developed on the basis of the verbal data represents the main steps of the generalized abstract problem solving procedure. Furthermore, the linguistic data reflect a complex interplay of structural and functional object conceptualizations and mapping processes between them.

Introduction

A great variety of everyday tasks involve complex cognitive processes: these include route planning and event scheduling, decision making, using household appliances for specific purposes, and many more. What kinds of thought processes are involved in dealing with such tasks? Much research in the area of cognitive science, in particular cognitive psychology and – increasingly – artificial intelligence has been devoted in the past decades to accessing cognitive processes across various types of task, often for purposes of modelling human ways of thinking, and reproducing them in artificial agents. Quite often, such research involves the elicitation of natural language, either as external representations of current internal processes while solving a particular task in so-called think-aloud protocols, or as retrospective reports which are suitable reflections of the earlier thought processes (Ericsson & Simon, 1984).

Currently, this particular type of linguistic data interpretation remains largely uninformed by linguistic expertise. Usually, cognitive scientists elicit and analyse language for the purposes demanded by the task at hand, without consideration of the particular features of the discourse type they are dealing with. The aim of this paper is to show the extent to which linguistic tools for discourse analysis are suitable for capturing and highlighting aspects of language in use that may be of crucial interest to cognitive scientists, both for purposes of investigating psychological procedures involved in problem solving and other complex cognitive processes, and for purposes of modelling such procedures formally and computationally. As one potential outcome, the design of artificial agents that share particular aspects of human thought may profit greatly from a structured, in-depth understanding of the language used to externalize complex cognitive processes. Natural language is unquestionably the most common medium required and used to convey information between agents; employed in an informed way, it can serve as a fruitful mediator and representation method bridging the gap between computational issues and human thought.

We will start out by presenting the main features of a newly developed method called Cognitive Discourse Analysis (CODA), discussing elicitation as well as analysis procedures that have been successfully adopted so far. The second part of this paper concerns a more detailed, exemplary presentation of our current study concerned with object assembly.

CODA – Cognitive Discourse Analysis

Ericsson and Simon (1984) provide a broad and exhaustive account of previous literature in the area of language data collection along with cognitively complex tasks. Along with this, they discuss the question of the validity of verbal data as such, i.e., the extent to which – and the circumstances under which – participants' accounts of thought processes can be trusted. Their work contains a detailed account of the recommended data collection, annotation, and analysis procedures for verbal data, particularly think-aloud protocols and verbal reports. This approach has since been established as a kind of paradigm which is regularly re-used and adopted for a great variety of purposes.

In this tradition, linguistic features are only seldom accounted for in any way. The analysis of verbal protocols generally focuses on extracting aspects that the speakers are themselves aware of, i.e., the conceptual strategies and processes that they report explicitly. However, linguistic representations may reflect conceptual aspects that the participants take for granted, being unaware of the significance of particular ways of framing a verbal representation. This is the basic motivation for adopting discourse analytic tools in addition to the content-based interpretation of verbal protocols.

The approach of CODA targets a systematic analysis of linguistic patterns by addressing the way *how* some content is expressed or structured in addition to *what* is said. As such, this idea envelops a wide range of achievements and ideas from the field of linguistic expertise; indeed the CODA methodology is flexible enough to allow for, and unite, various different perspectives. However, certain procedures of elicitation as well as analysis may be more suitable for particular purposes than others – both in terms of identifying cognitive processes in general, and in terms of addressing specific research questions in a particular study. Crucially, *text-type related* and *task related* aspects need to be differentiated carefully. On the one hand, some types of linguistic patterns are systematically related to the usage of a particular text type (e.g., Biber, 1989), yielding standard and less standard ways of representing information. On the other hand, a range of systematic aspects in language involve cognitively relevant phenomena such as presuppositional aspects, semantic under-specification, and conceptual categorizations, building a bridge between the available linguistic system and the current topic represented during a problem solving task. Such insights support the interpretation of those aspects of the language data that are in fact peculiar to the task at hand, i.e., that reflect cognitive processes related to the participants' behaviour.

In the following, we briefly sketch a range of elicitation issues as well as analysis procedures that have been usefully adopted in CODA-based studies, along with examples. Concerning elicitation, a main focus will be on the significance of linguistic data types. Analysis procedures, on the other hand, center around systematic patterns in language that may be cognitively relevant for a particular task. Following this overview we will turn to a more detailed discussion of a set of think-aloud data collected during a problem solving task: assembling a dollhouse with limited prior information about the functions of the available parts.

Elicitation in CODA: Significance of linguistic data types

Ericsson & Simon's (1984) framework provides a good basis for identifying the cognitive significance of particular text types. For instance, information verbalized *during* the task and retrospective probing is likely to reflect cognitive processes within short-term memory, while generalized questions *after* the task require intermediate processing influenced by long-term memory. Therefore, *think-aloud protocols* and *retrospective reports* are best suited to elicit unbiased verbalizations of cognitive processes. While this insight motivates a focus on these particular text types, other types of verbalizations have different effects which may also be welcome under certain circumstances. For some purposes, slightly enhanced discourse goals – if well understood and systematically accounted for in the interpretation of the elicited language data – may lead to further useful insights. If the instruction given to elicit verbal protocols along with complex cognitive tasks is formulated in a less neutral way, inducing some kind of bias, this will influence not only the elicited language as such but may also affect the way the participant perceives the task, and thus have an impact on behavior. Under certain circumstances, the requirement to verbalize may promote a better understanding of the task itself – or it may lead to an impairment (Schooler, Ohlsson, & Brooks, 1993). In the following, we will briefly discuss three further widely used text types, which may be suitable for different purposes.

Instructions for other people may trigger intermediate processes of verbalization, such as explanations. Such data may provide insights into how cognitive processes can be conveyed from an expert (in solving a complex task) to a novice. Clark & Krych (2004) present a relevant analysis of dialogues concerned with a joint problem solving task (building a LEGO model), showing how experts adjust their instructions according to their partners' reactions. One important field of investigation within spatial cognition research concerns the analysis of route directions. Here, participants are typically not asked to describe what they were thinking when finding their way, but use a verbal representation to enable another person to find their way (e.g., Denis, 1997). This opens up further possibilities for eliciting language under consideration of different perspectives. Apart from the text type itself, the precise nature of the (perceived) discourse goal (i.e., why language is produced) plays a decisive role, with systematic influences on the level of granularity or detail expressed in language as well as the trains of thought that are triggered by the way the current linguistic aims are understood. A recent study by Wiener, Tenbrink, Henschel, and Hölscher (2008), which involved three different types of linguistic data (think-aloud protocols and written route descriptions "for themselves" and "for a stranger"), revealed that the way a route information is conveyed depends on the perceived relevance of the question for the route receiver, based on previous knowledge, presumed preferences (nice routes vs. shortest option), and so on. Such issues have consequences not only for the way a route instruction is formulated but also on the information itself, i.e., the choice of a route. Moreover, the think-aloud protocols highlighted the incremental cognitive processes involved in the actual wayfinding process, drawing on visual information. Thus, variation in the elicitation of language data led to enhanced insights about a range of crucial cognitive aspects.

A recent linguistic in-depth comparison (Tenbrink, 2008a) of three different text types produced by a single study participant (a think-aloud protocol with a subsequently produced retrospective report plus an instruction "for a friend") in relation to a variant of the *Traveling Salesperson Problem* addressed the distinct perspectives of each data type on the conceptualizations of the problem solving task at hand. The linguistic features of the think-aloud data reflected cognitive chunking and a gradual shift of attention focus with respect to perception and action. The retrospective reports coherently represented those cognitive processes that after a number of trials turned out to be most decisive for this particular person. The instructions formulated for an addressee additionally revealed potentially useful ideas that were not necessarily decisive for the participant's own actions.

Interview questions. Ericsson & Simon (1984) pointed out that questions posed by the experimenter, if not formulated in a very general way, lead to filtering processes and may address aspects that the subjects never actually attended to themselves during the problem solving process (such as reasons and motivations). However, this may not necessarily be a disadvantage. In the analysis of strategies used in particular problem solving tasks, intermediate thought processes may lead to the mention of strategies that could have been used but were not; due to conscious reflection, participants may realize that better performance on the current task could have been achieved. Such a recognition of further possible strategies would in most cases also be reflected linguistically, highlighting the need for detailed linguistic analysis. However, after the task, the motivation for improving performance may be reduced, as is the perceptual input; thus, it becomes even more difficult to imagine good ways of solving the problem. Thus, the main danger consists in participants wrongly believing that they solved the task in a particular way; therefore, a particular kind of verbal data always needs to be controlled against other ways of verbalization as well as against behavioral data. Generally, relying on think-aloud data alone may often not be sufficient since verbalizations during the task may influence behavior under certain circumstances, and they may be incomplete in systematic respects (Ericsson & Simon, 1984). Similarly, Someren et al. (1994) point out that retrospective reports may sometimes omit false leads, i.e., fruitless thought processes that the problem solver discarded after a while.

Dialogue. Apart from the possibility of eliciting dialogues between experts and novices as already mentioned, further variations are possible. Boren and Ramey (2000) suggest extending Ericsson & Simon's approach to a *communication-based* one: they argue for allowing the experimenter to communicate in a fairly natural way with the participant in order to elicit more information and to support the user in exploring the ideas and issues at stake. Krahmer and Ummelen (2004) compare this approach directly with Ericsson and Simon's and find that dialogic interaction during performance appears to have an influence on task success but not necessarily on the contents of the comments being produced (thinking aloud vs. dialogue).

Clearly, when engaging participants in dialogue, or when using questionnaires, one should avoid questions that are theory-driven to such a high degree that they bias participants to the kinds of answers that the researcher is looking for. In CODA, various different verbalizations are triggered, not in the first place by specific questions, but by suggesting *different discourse tasks* to the participant. Thus, participants may be asked to produce verbal representations not only for the purpose of revealing thought processes, but primarily for a different purpose in which these thought processes are again put to use, this time not for behavioral purposes but in order to create a linguistic product. This includes monologic and dialogic discourse, as well as spoken and written language. Spoken language differs from written language, for example, with respect to the usage of certain markers of hesitation (see below), repetitions and self-corrections, lexical choices, typical syntactic patterns, and so on. With the presence of an (active) addressee, dialogue patterns such as alignment, clarification, and adaptation to the interaction partner come into play that influence the amount and representation mode of the information to be conveyed, and therefore highlight different aspects as compared to other discourse types. By systematically eliciting and comparing several such accounts, it is possible to approach the thought processes underlying verbalizations from different perspectives.

Analysis procedures in CODA

Structure and information presentation. The way in which texts (of any type) are structured can be expected to relate systematically to the way the underlying cognitive processes are structured. This concerns both the text as a whole, revealing for instance temporal and causal relationships developing gradually, and smaller portions of the text, for example information packaging within single clauses. Insights from linguistic theory such as Functional Grammar (Halliday, 1994) support the identification of parts of the text that are represented as Given or New, based not only on linear order but also on a range of grammatical features such as presenting vs. presuming reference types. Information presented as Given is linguistically taken for granted, which (if not supported by the previous text) may serve rhetorical purposes or reflect the underlying trains of thought. Information presented as New is apparently "newsworthy" for the speaker. Such effects may be supported by the usage of explicit discourse markers (see next paragraph). Related to our study of route planning under diverse circumstances (Wiener et al., 2008), we analyzed the way in which information about landmarks was packaged in think-aloud protocols in various conceptual situations (Tenbrink, 2008b). The analysis revealed a high amount of occurrences of presuppositions and non-anchored spatial references. For example, the utterance "At the concert hall take the Sedan street in the direction of the theatre" presupposes the location of both *the concert hall* and *the theatre* (i.e., their location cannot be derived from this utterance, though it may be derivable from the earlier discourse); in contrast, due to the spatial anchoring of *the Sedan street* within the utterance, its spatial location can be mentally integrated directly. This reveals the underlying spatial representation on the part of the speaker, where the presupposed locations are firmly anchored but not made prominent, leading to necessary inference processes on the part of the hearer.

Discourse markers. In a line of work on an approach called "psychopragmatics" (Caron-Pargue & Caron, 1991), Caron (1996) identified a number of linguistic markers that may reflect cognitive processes. Particularly interesting in this respect is the usage of connectives: On the one hand, connectives (such as *before, because, while*) serve to explicitly structure the represented contents, revealing how the participant construes the concepts and relations involved. On the other hand, certain markers that are particularly prominent in spoken language may reflect hierarchical thought processes; for instance, occurrences of "Okay, now…" may signal the completion of a subprocess together with the start of a new one. In Tenbrink & Seifert (under review), a route planning task involved the mental combination of two domains, *planning* (based on a map) and *travelling* (in the real world); this combination was systematically reflected by modal markers in retrospective reports.

Lexical choices. The way words and concepts (typically, nouns) are used may be revealing about the role of a particular semantic or conceptual field during a problem solving task. In the analysis of a version of the Traveling Salesperson Problem (Tenbrink & Wiener, 2009), we were interested in the impact of *colour* and *shape* on the path planning processes required for this particular problem solving task. While strategies focusing directly on either one of these concepts were rarely formulated explicitly (which is not surprising since attending to colour or shape did not support the problem solving process in any direct way), the lexical analysis revealed that participants actually relied heavily on concepts of colour, but not shape. In Tenbrink & Seifert (under review), on the other hand, a detailed lexical analysis supported the differentiation of planning and travelling domains based on choices and combinations of words for particular thought processes.

Activity sequences. A focus on the verbs used in verbal protocols reveals the types of activities that are prominent for a participant during a complex cognitive task. According to Halliday (1994), verbs can be classified into a limited number of types according to their basic semantic function; the three main types are verbs of *being* representing abstract relations, verbs of *sensing* representing consciousness, and verbs of *doing* representing the physical world. Each of these types (and some further subgroups) have their own grammatical restrictions as well as functions in discourse. Starting from this classification, a close examination of the development of processes (i.e., usage of verbs and possible nominalizations of verbs) can reveal the particular types of activities that the participants attend to during the task. Such analysis always focuses on whole constructions with verbs at their center, rather than attempting to interpret decontextualized usages. In Tenbrink & Wiener (2009), this type of analysis led to the proposal of an accumulated procedure for solving the Traveling Salesman Problem, generalizing over all collected protocols.

Exemplary study: Object assembly

The lasting success of companies like IKEA suggests that people are willing to assemble their furniture on their own. In general they are aided in their effort by a manual that is supplied by the manufacturer; however, some people are reluctant to use these, or the manual may be missing. Moreover, a situation may occur in which object parts are discovered without information about the composite object that may be assembled from the parts. In such situations, object assembly turns into a problem solving task involving an interesting variety of cognitive processes, resembling earlier findings in other domains (Tversky, Heiser, Lee, & Daniel, 2009). A range of studies have addressed the conveyance of information relevant to an assembly process in situated communication (e.g., Rickheit & Wachsmuth, 2006). In our explorative study, we collected think-aloud data and retrospective reports in an object assembly task, so as to learn more about the cognitive processes involved in solving such problems. A number of studies have shown the impact of prior knowledge on recall (Bransford & Johnson, 1972) and comprehension (Dixon, 1987). In order to address the impact of the amount of prior information on the cognitive processes involved and their linguistic reflections, we tested participants in three conditions. The participants in the first condition were told nothing about the nature of the composite object and thus lacked contextual information altogether. Those in the second condition were told that a dollhouse should be assembled and thus provided with domain knowledge (the general context of the assembly). Those in the third condition were given very specific contextual information on the object and the actual goal state by combining verbal and visual information. In the following we sketch the procedure and analysis involved in this project in order to illustrate procedures of the CODA methodology in practice. As this is work in progress, the analysis is not yet complete; however, we report a range of patterns emerging from the procedure of analyzing think-aloud data.

Procedure

52 participants (graduate and under-graduate Bremen University students, 28 female, 24 male) were presented with a box containing 10 object parts, plus a large roof piece and 2 wooden boards, all of which belonged to a wooden two-story dollhouse by the German toy brand “Selecta”. They were randomly assigned to three conditions. In the first condition (*no goal* condition) they were asked to assemble all given parts in a sensible way. Participants in the second condition were asked to use all parts to assemble a two-story dollhouse in a sensible way (*verbal goal* condition). In the third condition participants were shown a picture of the complete two-story dollhouse for 30 seconds and asked to assemble the depicted dollhouse (*verbal & visual goal* condition). All of the participants were trained and instructed to think aloud during

the assembly, based on Ericsson and Simon's (1984) methodology. After they indicated completion of the task, they were asked to give a retrospective report on the assembly procedure (not analyzed here). The participants were video-taped and their speech was recorded and later transcribed.

Analysis

Structure. Tversky et al. (2009) identified a common structure involved in explanations of construction tasks; across modes (gestures, diagrams, and words), a clear beginning, middle, and end could consistently be identified. We were interested in examining whether similar structures would emerge in think-aloud protocols, which differ from explanations by the lack of an explicit addressee. Based on our data we defined three stages as follows:

- The *beginning* was defined as utterances produced after entering the room and before starting the actual assembly process. Two main categories in regard to content were identified: repetition of parts of the instructions and first perceptual remarks. The first category contained reminders of thinking aloud or repetition of object parts mentioned in the instruction (e.g. box, table, parts). The majority of these utterances included the linguistic marker ‘okay, well’ signaling that the passive part of receiving instructions was finished and the active part started.
- The major *middle* part directly concerns the assembly process. It contains a *local structure* of sub-processes (also referred to as *episodes*).
- The *end* was defined as utterances following the actual assembly process, expressing completion of the task.

All of the 22 protocols analyzed in this respect so far exhibited this structure. Beginning and end parts were analyzed in regard to content as well as linguistic form; this will not be pursued further here. The analysis of the middle part focuses on the sub-processes of the assembly and their linguistic representation. In the following we will briefly present two aspects of this procedure: first, the content-based derivation of a process model representing the cognitive processes involved in object assembly, and second, a lexical analysis highlighting the mental representations of objects and their functions, and mappings between these, as part of the assembly process.

Process model. Given the explorative nature of the analysis a preliminary process model was derived by a context based analysis of the verbalized actions in a pilot protocol (cf. Someren et al., 1994), drawing on Palmer's (1977) account for working definitions. This model was validated and expanded by the analysis of (so far) 10 further protocols from all three conditions. According to Palmer (1977), problem solving consists of *explorative hypotheses, false leads, dead end, backtracking,* and *fresh starts*. For our current purposes these categories were more specifically defined as follows. *Hypotheses* are ideas and assumptions about objects, moves or consequences of actions. Actions that are evaluated as wrong moves are called *false leads*. *Dead end* states are temporary impasses or states of frustration. *Fresh starts* are instances of disassembly of parts or the whole object and their reassembly in a new way. In addition to utterances expressing these states, some participants also comment on the nature of the task (*meta-level*) or verbalize thoughts that are not directly task related (*aside*). All verbalizations in the middle part of the 11 protocols could be classified as representing one of these categories. Possibly related to the fact that this particular discourse did not serve a communicative intention, some states are not explicitly verbalized. For instance, positive evaluations are seldom stated, but they are implicit in a new hypothesis which shows that the assembly process proceeds. The current version of the process model is shown in Figure 1. The processes that were identified in the think-aloud data are similar to the search-control process described by Newell and Simon (1972). The next step in this analysis procedure will be to spell out the particular linguistic representations used for each of the actions and states. This will provide further insights about their nature and about the patterns of verbalization, which may be useful for computational purposes as explained above.

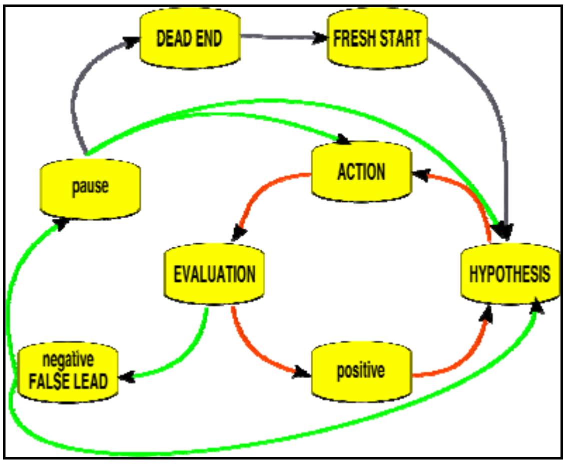


Figure 1. Process model derived from verbal protocol data in object assembly

Lexical analysis. The analysis of the *nouns* used by participants allows for conclusions about the current mental representation of an object part at a particular stage in the assembly process. The nouns can be distinguished in regard to their goal specificity; they can either be *generic*, i.e. not goal object related (e.g. *thing, wood, board*) or *specific* to the goal domain (e.g. *roof, wall, window*). A clear mental representation of the target object should be reflected in a frequent use of goal specific nouns. Participants who were given little or no prior information should therefore use goal specific nouns less regularly, or only later on in the assembly process. Participants who were given a picture of the goal object, in contrast, could draw upon an existing external representation of the object from the start. As an outcome, the distribution of generic and specific nouns should differ systematically between conditions. Our analysis of 22 protocols so far supports this assumption.

While the analysis of the usage of nouns highlights the existence of two levels of representation (generic / specific), the *mapping process* between object parts and functions in the targeted dollhouse is particularly interesting. A *generic* noun (e.g. *thing*) or a *deictic expression* (such as *this (one)*) refers to a particular object in the stock; a *specific* name of a role within the dollhouse (e.g. *wall*) assigns a function to it. These two levels of conceptualization may be connected by *comparison, modal verbs* or *relational verbs*. Altogether, explicit mapping occurs 77 times in the 22 think-aloud protocols analyzed in this respect so far (distributed approximately evenly across individual protocols and conditions, with a slightly higher relative frequency in the *verbal goal* condition as compared to the other conditions). An analysis of the patterns of its occurrence highlights the impact of prior information on mapping processes as follows.

We were particularly interested in the amount of *certainty* concerning the mapping, as this sheds light on the stability of the mental representation of a currently focused object. Linguistic markers expressing high certainty should reflect clearer mental representations on the part of the speaker than linguistic markers expressing neutral or tentative mapping processes and uncertainty. We identified three categories of linguistic representations of mapping processes. First, a high degree of certainty is expressed by the use of relational verbs (present tense of *be*) and a particular set of modals known to signal a high level of certainty (*must*, *will*) (Martin & Rose 2003). Second, another set of modals such as the German equivalents of *may* (*müsste*, *könnte, sollte*) expresses a lower, though still positive level of certainty. The third way in which objects may be assigned functions linguistically is via comparison (such as *(looks) like, (use) as*). Such markers neutrally reflect a tentative assignment of a function to an object. In some cases, hedges such as *a bit* in *this looks a bit like a roof* add an element of uncertainty to the assignment.

According to our analysis of 22 protocols so far, it appears that participants in the *verbal goal* condition assign meaning by using linguistic markers of high certainty most often. These participants were given information about the nature of the target object but not its particular appearance; therefore, they may have had features of typical dollhouses in mind (e.g. open front, walls, roof) and simply matched those to the objects at hand in some suitable way. Mappings via modals expressing a lower degree of certainty were most often used by participants in the *verbal & visual goal* condition. These participants were shown a picture of a correctly assembled dollhouse which they were asked to match. This may have led to a lower degree of certainty if the object parts could not readily be matched to the target picture in memory. Mappings via comparison were most frequent in the *no goal* condition, reflecting the fact that participants were altogether uncertain about the object's functions and tentatively explored mapping options. The analysis of the remaining protocols will shed more light on these issues. However, already at this intermediate stage, a pattern emerges showing that the amount of prior information systematically affects the ways in which object parts are referred to. These results highlight how the cognitive process of assigning functions to previously undefined object parts is linguistically expressed in various ways exhibiting a scale of changing certainty. This systematic variety in linguistic expressions is not necessarily part of the participants' conscious assembly process, but reflects how mental representations change through time, mediated by the amount and nature of prior knowledge.

Conclusion

The linguistically based analysis of verbal protocols enhances the range of insights that can be gained about the cognitive processes involved in complex tasks. In this paper, we have discussed a range of issues concerned with data elicitation, analysis, and interpretation. Two general conclusions can be drawn from this account. On the one hand, diverse types of discourse may be useful for gaining diverse types of insight about thought processes that are externalized in language for diverse purposes. This fact can be made use of for implementation in artificial agents both with respect to computational modelling of thought processes, and in the usage of language for purposes of mediation between different ways of processing (in machines and humans). On the other hand, knowledge about the particular linguistic features involved in texts of any kind may support the analysis of verbal reports effectively, by enabling a focus on those kinds of linguistic items that potentially reflect cognitive processes of interest for scientific progress. While a content-based analysis of language data is suitable for highlighting the conscious processes that study participants verbalize, the structure and linguistic choices involved in these verbalizations contain much more information than one might suspect at first sight. This kind of subtle reflection of cognitive processes becomes informative whenever linguistic evidence exhibits systematic patterns in language use. Particularly if these patterns can be matched to other types of evidence (such as behavioral results, eye movements, and the like), the linguistic data analysis can be trusted as a particularly valuable tool for accessing complex cognitive processes in problem solving tasks.

Acknowledgements

Funding by the Volkswagen Foundation for the Tandem project "Wayfinding Strategies in Behavior and Language", granted to Jan Wiener and Thora Tenbrink, is gratefully acknowledged; also, we thank Universität Bremen for funding the doctoral project "Psychopragmatic analysis of problem solving processes". We thank Katja Abeln for her support in carrying out the assembly study and transcribing the data.

References

Biber, D. 1989. A typology of English texts. *Linguistics* 27:3-43.

Boren, M. and J. Ramey. 2000. Thinking aloud: Reconciling theory and practice. *IEEE Transactions on Professional Communication 43*(3), 261–278.

Bransford, John D. and Marcia K. Johnson. 1972. Contextual Prerequisite for Understanding: Some Investigations of Comprehension and Recall. *Jounral of verbal learning and verbal behavior*, 11: 717-726.

Caron, Jean. 1996. Linguistic markers and cognitive operations. In: Caron-Pargue, Josiane and Steven Gillis (eds), 1996. *Verbal Production and Problem Solving*. Antwerp Papers in Linguistics 85, Universiteit Antwerpen, pp. 11-28.

Caron-Pargue, Josiane and Jean Caron. 1991. Psychopragmatics vs. sociopragmatics: the function of pragmatic markers in thinking-aloud protocols. In Jef Verschueren (ed.), *Pragmatics at issue: Selected papers of the International Pragmatics Conference*, Antwerp, August 17-22, 1987, Volume I. Amsterdam: Benjamins, pp. 29-36.

Clark, Herbert H. and Meredyth A. Krych. 2004. Speaking while monitoring addressees for understanding. *Journal of Memory and Language 50*, 62–81.

Dixon, Peter. 1987. The Structure of Mental Plans for Following Directions. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 13(1):18-26.

Denis, Michel. 1997. The description of routes: A cognitive approach to the production of spatial discourse. *Cahiers de Psychologie Cognitive, 16*(4):409-458.

Ericsson, K. Anders, and Simon, Herbert A. 1984. *Protocol analysis –Verbal reports as data*. Cambridge, MA: Bradford books/MIT Press.

Halliday, M.A.K. 1994 (2nd ed). *An introduction to functional grammar*. London: Edward Arnold.

Krahmer, Emiel and Nicole Ummelen. 2004. Thinking About Thinking Aloud: A Comparison of Two Verbal Protocols for Usability Testing. *IEEE Transactions on Professional Communication, Vol. 47, No. 2*, 105-117.

Martin, J.R., & David Rose, 2003. *Working with Discourse: Meaning Beyond the Clause*. Continuum International Publishing Group Ltd.

Newell, Allen and Herbert E. Simon. 1972. *Human Problem Solving*. NJ: Prentice Hall.

Palmer, S.E. 1977. Hierarchical structure in perceptual representation. *Cognitive Psychology 9*(4), 441-474.

Rickheit, Gert and Ipke Wachsmuth (eds.). 2006. *Situated Communication.* Mouton de Gruyter.

Schooler, Jonathan W., Stellan Ohlsson, and Kevin Brooks. 1993. Thoughts Beyond Words: When Language Overshadows Insight. *Journal of Experimental Psychology: General*, 122:2, 166-183.

Someren, W. M., Barnard, F. Y., & A. C. J. Sandberg. 1994. The think aloud method. A practical guide to modeling cognitive process. London: Harcourt Brace & Company.

Tenbrink, Thora. 2008a. The verbalization of cognitive processes: Thinking-aloud data and retrospective reports. In Wiebke Ramm & Cathrine Fabricius-Hansen (eds.), *Linearisation and Segmentation in Discourse. Multidisciplinary Approaches to Discourse 2008 (MAD 08)*, Feb 20-23 2008, Lysebu, Oslo. Oslo: Dept. of Literature, Area Studies and Europ. Languages, Univ. of Oslo, pp. 125-135.

Tenbrink, Thora. 2008b. Wayfinding in an urban environment: A discourse-analytic approach to thinking-aloud data. *Space, Interaction, and Discourse*, Aalborg, 12.-14.11.2008.

Tenbrink, Thora and Inessa Seifert (under review). Conceptual Levels and Strategies in Tour Planning.

Tenbrink, Thora and Jan Wiener. 2009. The verbalization of multiple strategies in a variant of the traveling salesperson problem. *Cognitive Processing 10*:2, 143-161.

Tversky, Barbara, Julie Heiser, Paul Lee, and Marie-Paule Daniel. 2009. Explanations in gesture, diagram, and word. In Kenny Coventry, Thora Tenbrink, and John Bateman (eds), *Spatial Language and Dialogue.* Oxford: Oxford University Press, pp. 119-131.

Wiener, Jan, Thora Tenbrink, Jakob Henschel, and Christoph Hölscher. 2008. Situated and Prospective Path Planning: Route Choice in an Urban Environment. In B. C. Love, K. McRae, & V. M. Sloutsky (eds.), *Proceedings of the 30th Annual Conference of the Cognitive Science Society* (pp. 851-856). Austin, TX: Cognitive Science Society.

Learning and Recognition of Sketches for Complex Computational Cognition – Position Paper –

Angela Schwering[1] and Ulf Krumnack[2] and Helmar Gust[2] and Kai-Uwe Kühnberger[2]

[1] University of Münster, Institute for Geoinformatics
angela.schwering@uni-muenster.de
[2] University of Osnabrück, Institute of Cognitive Science
{krumnack,gust,kkuehnbe}@uos.de

Abstract. In order to enable machines to operate intelligently in their environment, it is important that they do not only collect sensory input about their environment, but also recognize and understand objects. Analogical reasoning is considered fundamental for many complex cognitive processes. In this paper, we present an experiment which gives empirical support of our hypothesis that object recognition and concept formation rely fundamentally on analogical similarities. Similar object sketches with the same structure are recognized faster and more frequently than similar object sketches with different structure. Afterwards, we introduce our analogy-making framework Heuristic-Driven Theory Projection (HDTP) and explain how HDTP can be used for object recognition.

1 Introduction

In order to enable machines to operate intelligently in our world, it is important that they do not only collect sensory input and observe the environment, but also recognize and understand it. The correct classification of perceived objects allows a machine to use its background knowledge about the world to reason on it. Sketches, i.e. freehand schematized drawings, are an intuitive medium for people to communicate about objects in the world. In this paper, we focus on learning and recognition of sketched objects. We present empirical evidence for our hypothesis that structural similarities are important in the human recognition process. We propose a computational model how machines recognize new sketches by detecting common structures to known sketches and classify the objects according to their ontological knowledge. We examine how concepts change over time and develop an analogy-based approach for learning and revising conceptual knowledge and for explaining the creation of new and abstract knowledge.

Realizing learning and recognition of sketched objects on a machine requires an appropriate language for describing spatial objects in their environments. It must be possible to capture the geometry of all elements in a scene and the spatial relations between them. Furthermore, the representational formalism must be adaptable to change representations of the same scene according to the different perceptions in varying contexts. Recognition requires the ability of comparing new stimuli to already known stimuli in the memory. The structural composition of the object parts is very important, in particular for sketches of spatial objects. Analogical mapping is used to compare two stimuli–a new stimulus and a well-known stimulus–for structural similarities. In a recognition task, the well-known stimulus can be a typical instance of a concept or the specification of a concept from memory.

The model of computational cognition proposed in this paper uses knowledge gained through recognition tasks to learn new and revise old concepts. The two main mechanisms for learning constitute learning via transfer and learning by abstraction [10]. Once a new stimulus is successfully classified, either additional knowledge about the concept can be transferred to the newly classified stimulus, or features observed about the new stimulus can be transferred and integrated in the existing concept description. This additional knowledge leads to a richer and more precise concept description. Moreover, the comparison process aligns analogous elements in both stimuli, i.e. reveals the commonalities of both stimuli at an abstract level. These analogous commonalities describe the essential characteristics defining a concept.

This paper is structured as follows: in Section 2, psychological evidence is provided that structural changes of a visual stimulus do influence object categorization of humans stronger than non-structural changes. Section 3 proposes some ideas for a model of object recognition based on the analogy engine Heuristic-Driven Theory Projection. Section 4 provides a vision how adaptations of representations for analogy-based stimulus recognition can be used for learning new concepts. Section 5 concludes the paper.

2 Object Categorization and Structural Alignment

2.1 The Experiment

A lot of common everyday objects are made up of several, distinct components. The same is true for the kitchen stove depicted by the line drawing in Figure 1. Some components typical for the outward appearance of such a stove have been highlighted in grey color. Obviously, these core elements are spatially related to each other. It is possible to describe these relationships in a qualitative manner. Commonly used spatial relations are topological, directional, or metric relations [1] and may involve other qualities such as symmetry and repetition of elements.

When applying this general idea to the stove in Figure 1, its highlighted components might be regarded as separate regions with certain underlying topological relations. The four hotplates on top could be regarded as four disjoint regions all of which are in turn situated inside Area 1. Underneath, Area 2 contains six disjoint temperature regulators. Similar relationships can be found as to the front handle and the spy window both of which are disjoint and situated

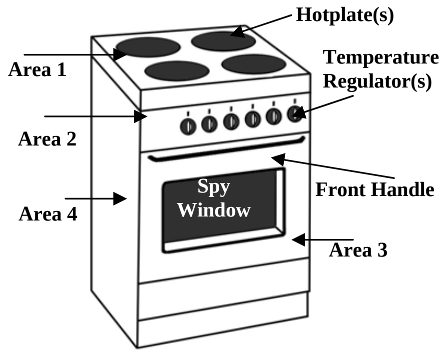


Fig. 1. Line drawing of a typical kitchen stove.

within another area (Area 3) on the stove's foreside. Furthermore, the lateral Area 4 directly meets Area 2, and so forth.

To investigate the role of structured representation in human object recognition, an experiment was set up, in which subjects had to recognize line drawings of different objects.[3] 132 line drawings were selected for the experiment. Of these, 72 functioned as filler items, whereas the remaining 60 drawings acted as the so-called "basic" experimental stimuli. The latter served as a basis for the development of four additional variations, namely two versions of non-structural modifications and two versions of structural modifications (cf. Figure 2). Generally speaking, each experimental condition was conceptualized as a pair of two experimental stimuli, henceforward referred to as item pairs.

Basically, a single experimental trial was composed of a source image stimulus and a subsequent target image stimulus. First, the source stimulus was shown and all subjects were expected to name the object that they thought to have identified in the black and white line drawing by an oral answer. Then, subjects had to press the keyboard's down-arrow key to call up the target image. In preparation for the imminent stimulus, a fixation cross with a duration of 250 ms was shown in the middle of the monitor prior to the occurrence of the target image. Finally, the target image stimulus appeared for maximally 650 ms. This time, the subjects' task consisted in deciding as quickly as possible by pressing the "yes" or "no" button whether the object they were just seeing was an instance of the same concept as the object they had named in the step before.

Due to the five experimental conditions, we created equally many stimulus lists that counterbalanced item pairs and conditions. Each subject saw 36 filler item pairs, 12 MAT items pairs, 12 NS1 item pairs, 12 NS2 item pairs, 12 S1 item pairs, and 12 S2 item pairs yielding 96 experimental trials in total. Figure 2 specifies the modified versions of the original stimulus.[4]

[3] The interested reader is referred to [20] for a complete presentation of the experiments.

[4] 75 native German subjects, 50 females and 25 males, volunteered for the experiment and confirmed normal or corrected normal vision. The vast majority of participants

MAT: The match condition was conceptualized as an item pair with identical source and target images. Solely the 60 basic experimental stimuli served as basis to set up this condition. Furthermore, this condition served as a baseline with respect to the reaction time measurements and required a clear ”yes” response from the subjects.

NS1: This condition entailed the movement of significant picture elements. These manipulations were not taken for a structural change since it was made sure that the topological relationships between the manipulated and unaffected picture elements remained untouched. It was anticipated that the subjects would show a high tendency to give a ”yes” response.

NS2: This condition entailed the resize of picture elements without moving them to another position. Simple resize was not taken for a structural change as long as the topological relations between the manipulated and other picture elements remained constant. It was anticipated that the subjects would show a high tendency to give a ”yes” response.

S1: As for the first structural condition, it exclusively implicated the removal and/or addition of selected picture elements. Adding to or removing significant elements from the overall scene was regarded as a clear structural change. It was decided to accept both a “yes” and a “no” response as “potentially correct”.

S2: The second structural condition likewise implied the movement of significant picture elements as with condition NS1. However, this time a structural change was deliberately caused by moving selected elements into another area. Alternatively, this condition involved the resize of desired picture elements as with condition NS2. Both ”yes” and ”no” were accepted as potentially correct answers.

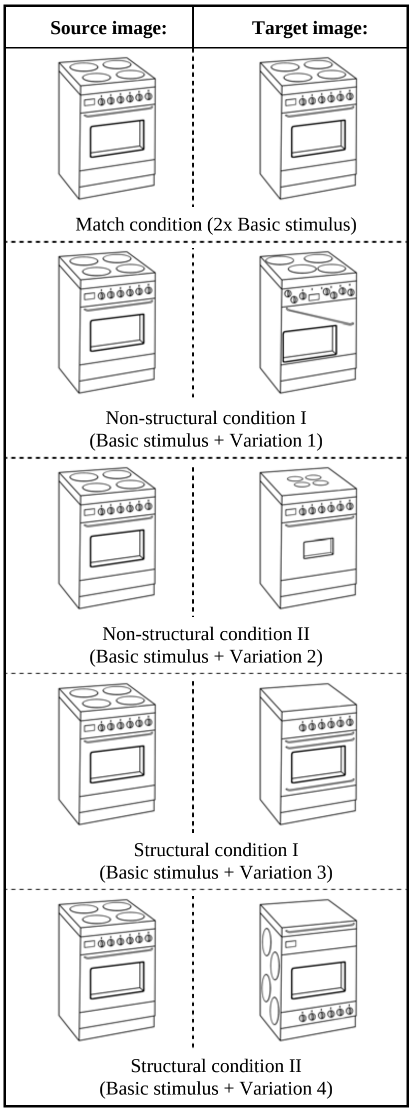


Fig. 2. The types of stimuli used in the experiment: Match condition, non-structural condition I (NS1), non-structural condition II (NS2), structural condition I (S1), and structural condition II (S2).

2.2 Results

For the goals of this paper, it suffices to find evidence for the assumption that humans would need more time to recognize structurally manipulated objects compared to non-structurally manipulated objects. As a consequence, it was decided to combine both non-structural (NS1 & NS2) as well as the two structural

consisted of undergraduate students who were enrolled in Psychology or Cognitive Science at the University of Osnabrück. The mean age was 23.2 years, ranging from age 18 to age 58. The experiment was conceptualized and generated with the aid of the software suite E-Prime 2.0 by Psychology Software Tools Inc.

Condition	RT in ms (Std. Dev)	ACC in %	Yes / No Ratio in %
MAT	618 (147)	95.6	—
NSCOM	708 (182)	—	82.1 / 17.9
SCOM	752 (200)	—	61.3 / 38.7

Table 1. Descriptive statistics results - analyses by subjects ("Yes" and "No" responses).

conditions (S1 & S2), essentially because of their strong relatedness.[5] The relevant reaction times per subject were summed up and averaged afterwards. The same holds for the "yes"/"no" response ratios yielding the numbers shown in Table 1.

On that basis, a 1 (source image) × 3 (target image type: MAT, NSCOM, SCOM) factorial analysis of variance (ANOVA) including repeated measures was conducted on the response latencies by subjects and by items. Only data points that were maximally two standard deviations away from their corresponding mean were taken into account to reduce the quantity of outliers in the first place. A confidence interval of 95% was consistently used.

As a result, the main effect for target image type was highly significant in the analysis by subjects (F1) and by items (F2) with $F1(1.61, 112.56) = 87.51$, $p < .001$ (Huynh-Feldt corrected); $F2(2, 110) = 69.15$, $p < .001$. Concerning the main effect for list, it was only significant in the analysis by items, $F1(4, 70) = .52$, $p > .72$; $F2(4, 55) = 7.50$, $p < .001$. By contrast, the two-way interaction between list and target image type was not significant at all with $F1(8, 138) = 1.21$, $p > .30$; $F2(8, 108) = 2.00$, $p > .05$.

Several pairwise comparisons (MAT vs. NSCOM; MAT vs. SCOM; NSCOM vs. SCOM) were carried out. In all pairwise comparisons, the main effect for target image types was highly significant in the analysis by subjects and by items. As an example the results for NSCOM vs. SCOM are mentioned[6]. The main effect for target image type was highly significant by subjects and by items with $F1(1, 70) = 34.82$, $p < .001$; $F2(1, 55) = 15.90$, $p < .001$. The main effect for list was only significant in the analysis by items, $F1(4, 70) = .41$, $p > .80$; $F2(4, 55) = 3.40$, $p < .05$. The two-way interaction between list and target image type was not significant ($F1(4, 70) = 1.52$, $p > .21$; $F2(4, 55) = 1.16$, $p > .34$).

2.3 Discussion

The experiment provides two results that are relevant for the discussion in this paper. First, the relation of "yes"/"no" responses shows that the degree of recognition is significantly higher if the structure of the visual stimulus is not changed (NSCOM), compared to the cases where it is changed (SCOM). This indicates that subjects are more willing to accept an object as belonging to a category, if its relational structure stays intact. Second, the reaction time is shorter in these

[5] A detailed presentation of the results with separate treatment of all conditions can be found in [20].

[6] The complete results can be found in [20].

cases, indicating that the task is cognitively less complex if a structural match of stimuli can be found.

Both results back the claim, that object recognition seems to be based, at least partly, on matching structural representations of the provided stimuli. A cognitive plausible model of object recognition should therefore incorporate such representations and matching mechanisms. In the rest of the paper, we sketch a model for recognizing visual stimuli that is driven by analogical mapping and that furthermore allows to introduce a learning mechanism based on recognition.

3 Analogy-Based Recognition of Visual Stimuli

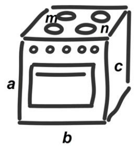
The model we propose is based on Heuristic-Driven Theory Projection (HDTP), a formal framework to compute analogies. This section gives a brief introduction to analogies and HDTP focussing on those aspects relevant to the intended application. A more comprehensive description of HDTP can be found in [19].

3.1 Syntactic Basis of HDTP

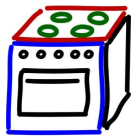
Classically, an analogy is established between two domains of knowledge, called *source* and *target* domain. By discovering corresponding structures in both domains, an analogical relation can be constructed. Such a relation can be used to identify commonalites and differences between the domains. Furthermore, gaps discovered in one domain can be filled by transferring knowledge from the other domain, based on the analogical relation. Such analogical inferences, though possibly incorrect from a logical point of view, can be a basis to explain certain aspects of cognitive phenomena like creativity and learning.

HDTP provides a formal framework to compute analogical relations and inferences, for domains represented in first-order logic. Both, source and target domain, are given by axiomatizations, i.e. finite sets of first-order formulae. The basic idea is to associate pairs of formulae from the domains in a systematic way. HDTP uses anti-unification to identify common patterns in formulae. In anti-unification, two formulae are compared and the most specific generalization subsuming both formulae is identified. As a result, besides the generalized formula a pair of substitutions is computed, that expresses the analogical relation between the two formulae.

This process of generalization by anti-unification can be iteratively applied to formulae of the two axiomatizations. However, it might be the case that for some axiom no good corresponding axiom exists on the other side. Nevertheless, there might still exist a good formula in the theory spanned by the axiomatization, i.e. among the formulae that can be derived from the axioms. In this case, HDTP will try to prove such a formula. This process can be considered as a kind of re-representation [11], since the originally given axiomatization is adapted to match the needs of the analogy considered. As a consequence HDTP does not compute an analogy between two specific axiomatizations, but between the theories spanned by these axiomatizations.



$line(a, (1, 1), (1, 4))$
$line(b, (1, 1), (4, 1))$
$line(c, (1, 4), (4, 4)) \ldots$
$round(m, (2, 5), (3, 5), 0.5)$
$round(n, (4, 5), (5, 5), 0.5) \ldots$



$polygon(p1, [a, b, c, d])$
$polygon(p2, [d, e, f, g])$
$above(p2, p1)$
$group(g2, [m, n, o, p])$
$inside(g1, p2)\ldots$

Fig. 3. Representation of a stove with its primitive elements in an unstructured way (left) and in a structured way (right).

HDTP distinguishes between domain knowledge (facts and laws holding for the source or the target domain) and background knowledge, which is true across domains. The background knowledge is of special importance in the context of re-representation, as it may be used to derive further formulae in the two domains, which then can be used again for generalization.

Uncovered parts of the source and the target domain, i.e. formulae that are not part of the analogical relation and therefore cannot be derived from the generalized formulae, are candidates for analogical transfer. The established analogical relation is used to translate these formulae. If the result does not lead to a contradiction in the other domain, it can be considered as an analogical inference, i.e. new knowledge that might be added to the axiomatization of that domain.

3.2 A Formal Language to Represent Spatial Objects

We now apply the ideas of HDTP to the processing and recognition of visual stimuli. In this setting, source and target are both from the same domain, i.e. sketch drawings. We distinguish between flat and structured representations of visual stimuli. A flat representation covers all features of a stimulus without any relational structure between them (e.g. the left side of Figure 3 listing the primitive visual elements of the stove). A structured representation captures regularities of a stimulus, like symmetry, iterations, Gestalt groupings etc. It furthermore comprises geometrical and topological relations. The structured representation on the right side of Figure 3 replaces the lines by a description of closed shapes such as polygons. Although the flat and the structured representation contain the same information, the structured representation is closer to the way humans perceive the visual stimuli. Our computational model of cognition shall take a flat representation as input and automatically compute a structured representation of the sketch reflecting human perception. A structured representation can be build from a flat representation according to a certain set of rules.

The application of HDTP as a framework for object recognition requires the development of a suitable language to represent spatial objects, the ability to adapt these representations such that analogous structures between the source and the target object become visible, and finally a mechanism for analogy-based learning of concepts. As a consequence the language has to meet two major requirements: it must describe all elements in a spatial scene with respect to the aspects relevant in human perception, but it must describe as well the spatial relationships which are important to compare and recognize objects. To reflect human perception, the language must comprise significant perceptual features, but also vocabulary to specify visual structures. When the human visual sensory system observes a spatial object, it transforms the unstructured information into a structured representation of coherent shapes and patterns. Human perception tends to follow a set of Gestalt principles: stimuli are experienced as a possibly good Gestalt, i.e. as regular, simplistic, ordered, and symmetrical as possible. Therefore the language focuses on basic Gestalt principles of perception, i.e. it allows for groupings according to the principle of similarity, the principle of proximity, closure, and good continuation.

The second requirement refers to spatial features: the geometry of elements in a scene and their spatial relations have to be represented in a way that allows for cognitively plausible reasoning. Common calculi for qualitative spatial reasoning such as RCC 8 for topological relations [14] and TPCC calculus [12] or neighborhood-based approaches [6, 15] for directional relations are integrated in the formal language.

In [17], we developed first steps towards a language for representing simple figures in geometric proportional analogies. Figure 3 shows exemplary a formal language representing a stove. On the left is an unstructured representation of the stove listing its primitive elements (lines and round elements). On the right is a structured representation of a stove: The four connected lines are represented as closed polygon. The four hotplates are grouped together according to the Gestalt principle of similarity and proximity. The topological relation inside and the directional relation above are captured as well. The groups of hotplates are inside the polygon $p2$ and polygon $p2$ is above polygon $p1$. In the following section, we explain how HDTP automatically adapts the unstructured representation to form a structured one.

3.3 Adaptation of the Representation for Analogy-Based Stimulus Recognition

The cognition of spatial objects involves the construction of a consistent and meaningful overall picture of the environment. Gestalt Psychology argues that human perception is holistic: instead of collecting every single element of a spatial object and afterwards composing all parts to one integrated picture, we experience things as an integral, meaningful whole. The whole contains an internal structure described by relationships between the individual elements.

In HDTP, a visual stimulus is described via a set of axioms specifying the features of all elements at a basic level (Figure 4). A set of perception rules and rules for spatial reasoning form the background knowledge of the system. The set of all formulae that can be inferred from the axioms comprises all possible re-representations of the same visual stimulus, but at different structural levels.

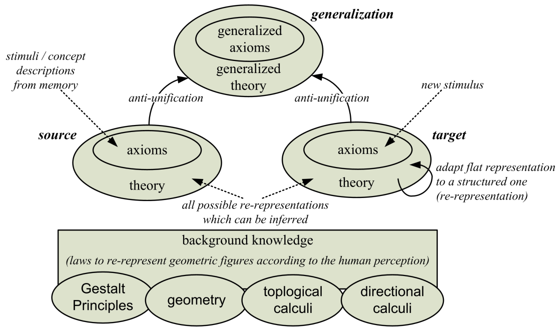


Fig. 4. Analogical comparison in the logical framework HDTP.

The initially flat representation can be transformed into a structured one by means of logical inference.

In the recognition task, a new stimulus (target) is compared to a known stimulus (source). The source stimulus is described via a structured representation recalled from the memory or knowledge base. The structural commonality between the flat representation of the target and the structured representation of the source is initially not obvious. To successfully classify a new stimulus, a mapping between the target stimulus and the source stimulus must be established, i.e. an analogous structure has to be established on the target stimulus. During the analogy-based mapping process the target must be re-represented such that common structures become visible. The re-representation process building a structure on the target side can be driven by heuristics motivated by human perception, like Gestalt principles.

Figure 5 shows adaptation rules as they can be found in the background knowledge: The first rule is applied to detect closed shapes such as a polygon and the second one is applied to compute topological relations such as inside. The re-representation process is driven by heuristics based on properties of human perception and by building a structure on the target side analogously to the structured stimulus on the source side. Experimental data shall give the necessary insight for creating appropriate heuristics reflecting human strategies in spatial object recognition. The heuristics have a great influence on the efficiency of the whole computational approach.

4 Analogy-based Learning, Concept Formation, and Creativity

Similarity judgment is one of the most central constructs in cognitive processes. Organization of conceptual knowledge in memory, recognition of new stimuli, and learning hinge crucially on similarity comparisons [8]. In particular, the role of structural similarity in relational categories has been considered as important [7]. We argue that structural similarity as detected in analogies is particularly important for learning spatial concepts. Our approach for computational cognition

Closed Shape (adapted from Gestalt principle)
$lineConnection(A, B)$:- $line(A, (\_, \_), (X, Y)), line(B, (X, Y), (\_, \_))$.
$lineConnection(A, B)$:- $line(A, (X, Y), (\_, \_)), line(B, (X, Y), (\_, \_))$.
$polygon(P, [\ldots])$:- ...

Topological Relation proper part (adapted from RCC8)
$regionConnection(X, Y)$:- $region(X), region(Y), not(disjoint(X, Y))$.
$part(X, Y)$:- $not(regionConnection(Z, X), not(regionConnection(Z, Y)))$.
$properPart(X, Y)$:- $part(X, Y), not(part(Y, X))$.
$overlap(X, Y)$:- $part(Z, X), part(Z, Y)$.

Fig. 5. Adaptation rules are stored in the background knowledge and define how unstructured descriptions can be re-represented to structured ones.

shall learn to classify spatial objects, i.e. the system shall be able to revise and refine its ontological knowledge during a training phase. Although researchers agree that analogy-making is central for human learning, there does not exist a comprehensive theory for analogical learning. Our own first ideas for a learning model based on HDTP were outlined in [18].

HDTP supports learning at two levels: analogical transfer and abstraction. Learning via analogical transfer means gaining new knowledge by applying additional knowledge about the source to the target. The system transfers knowledge about the concept (e.g. knowledge about the functionality) and applies this to the new stimulus. This enables the system to draw new inferences on the target side. Transfer also happens from the target to the source: the system observes characteristics about the new stimulus which leads to a revised concept definition. Learning via abstraction refers to the generalization process that is essential to derive abstract concept definitions. Existing approaches apply classical inductive learning which requires large set of data samples to create general laws. However, humans can generalize already over a small set of samples. Applying analogical comparison and describing structural commonalities at a general level is one possible way to make the essence defining a concept apparent. Reflecting this analogical generalization process is one of the strengths of HDTP [16]: during the analogical mapping, anti-unification automatically constructs a generalization for every aligned pair of formulae. This way, HDTP creates an explicit generalized theory over two theories – the source and the target theory. We argue that this generalized theory captures exactly the essential commonalities of the instances of a concept at an abstract level and therefore is an ideal mechanism for extracting the defining elements of a concept.

The following example illustrates how HDPT functions in concept formation and concept learning (cf. Figure 6). HDTP has a structural description of a stove in its knowledge repository. Presenting a new stove in a recognition task, HDTP detects the analogous structure and constructs a generalization containing the commonalities (i.e. common aspects about the geometry and spatial relation such as the temperature regulators being situtated in the front polygon). The generalization represents the concept "stove" at an abstract level. If again a new stove is presented in a second recognition task (e.g. the third one in the above figure), it could be classified as a stove, however the new generalization is not so

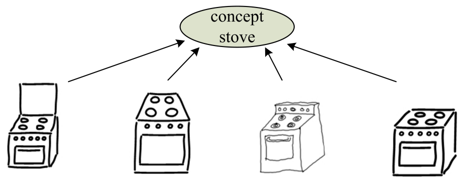


Fig. 6. A structural comparison of these stoves reveal that all stoves sketches have the form of a 3D cube with four hotplates on top and a spy window at the front. Three sketches show stoves with temperature regulators at the front.

specific on the position of the temperature regulators. First steps towards this incremental analogy-based learning have been sketched in [9].

5 Conclusions and Critical Evaluation

Analogies play a major role for cognition. We have shown empirically, that structural commonalities are important in object comparison and recognition: In a recognition task, subjects have recognized sketches of non-structurally varied objects faster and easier than sketches of objects which were structurally varied.

We have suggested an approach using HDTP, a symbolic analogy-making framework, to compute analogies between sketches of objects. HDTP is a promising framework, because it supports adaptation and learning at an abstract level. Many times analogical structures are not visible per se, but result from a comparison and mapping task. HDTP combines the representation of basic elements in a sketch with background knowledge on human perception. Therefore, HDTP can reveal commonalities in different contexts and different perceptions. It re-represents an unstructured flat representation of a sketch and determines a structured representation of the target stimulus which possibly matches the structured representation of the source stimulus. Furthermore, HDTP compares structures of source and target stimuli and computes a generalization of the shared structures. This supports concept learning.

Lately, various approaches have been developed to describe visual stimuli and detect analogous structures. *CogSketch* (comprising GeoRep and nuSketch) [4, 5] is a powerful tool for sketch understanding. A sketch consists of glyphs, which are the primitive elements. The spatial structure of the overall sketch is analyzed by topological, metric and directional relations between glyphs. A glyph is a piece of ink, i.e. a glyph can be a simple point but also a complex drawing. The approach proposed in this paper considers primitive elements as the most basic entity in a sketch, which itself can be re-represented as more complex figures by re-representation rules such as the ones depicted in Figure 5. The *Languages of Perception* [2] developed for Indurkhya's algebraic Interactionist Theory has a similar idea of re-representing simple geometric elements. It incorporates Gestalt-motivated mechanisms for re-representation such as groupings and iterations. The approach presented here goes beyond the Languages of Perception: We also aim at the explicit description of spatial relations and the integration of existing qualitative spatial reasoners. *Galatea* and the Proteus analogy model [3] was developed to describe visualizations in the context of problem solving. It aims at detecting visual similarities and transferring problem solving solutions, but not at the re-representation for perceptual understanding of sketches.

References

1. A. Cohn and J. Renz. Qualitative spatial representation and reasoning. In F. van Harmelen, V. Lifschitz, and B. Porter, editors, *Handbook of Knowledge Representation*. Elsevier, 2007.
2. M. Dastani and R. Scha. Languages for Gestalts of line patterns. *Journal of Mathematical Psychology*, 47:429–449, 2003.
3. J. Davies and A. K. Goel. Representation issues in visual analogy. In *Proceedings of the 25th Annual Conference of the Cognitive Science Society*, pages 300–305, 2003.
4. R. W. Ferguson and K. D. Forbus. GeoRep: A flexible tool for spatial representations of line drawings. In *18th National Conference on Artificial Intelligence*, Austin, Texas, 2000. AAAI Press.
5. K. D. Forbus, K. Lockwood, M. Klenk, E. Tomai, and J. Usher. Open-domain sketch understanding: The nuSketch approach. In *AAAI Fall Symposium on Making Pen-based Interaction Intelligent and Natural*, Washington, DC, 2004.
6. C. Freksa. Using orientation information for qualitative spatial reasoning. In A. U. Frank, I. Campari, and U. Formentini, editors, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, volume 639 of *Lecture Notes in Computer Science*, pages 162–178, Pisa, Italy, 1992. Springer.
7. D. Gentner and K. J. Kurtz. Relational categories. In W.-K. Ahn, R. L. Goldstone, B. C. Love, A. B. Markman, and P. W. Wolff, editors, *Categorization inside and outside the laboratory*, pages 151–175. APA, Washington DC, 2005.
8. R. L. Goldstone and A. Kersten. Concepts and categorization. In A. F. Healy and R. W. Proctor, editors, *Comprehensive Handbook of Psychology*, volume 4, pages 599–621. Wiley, New Jersey, 2003.
9. H. Gust, U. Krumnack, K.-U. Kühnberger, and A. Schwering. Integrating analogical and inductive learning at different levels of generalization. In *Workshop on Learning from Non-Vectorial Data (LNVD2007) at the 30th Annual German Conference on Artificial Intelligence (KI07)*, Publications of the Institute of Cognitive Science (PICS), Osnabrück, 2007. University of Osnabrück.
10. H. Gust, U. Krumnack, K.-U. Kühnberger, and A. Schwering. Analogical reasoning: A core of cognition. *KI - Zeitschrift für Künstliche Intelligenz, Sonderausgabe KI und Kognition*, pages 8–12, 2008.
11. U. Krumnack, H. Gust, K.-U. Kühnberger, and A. Schwering. Re-representation in a logic-based model for analogy making. In *AI 2008: Advances in Artificial Intelligence*, volume 5360/2008 of *Lecture Notes in Computer Science*, pages 42–48. Springer, 2008.
12. R. Moratz, B. Nebel, and C. Freksa. Qualitative spatial reasoning about relative position. In *Spatial Cognition III*, volume 2685 of *Lecture Notes of Computer Science*, pages 385–400. Springer, 2003.
13. E.-C. Mullally and D. P. O'Donoghue. Spatial inference with geometric proportional analogies. *Artificial Intelligence Review*, 26(1-2):129–140, 2006.
14. D. A. Randell, Z. Cui, and A. Cohn. A spatial logic based on regions and connection. In B. Nebel, W. Swartout, and C. Rich, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 3rd International Conference*, pages 165–176, Los Alloss, CA, USA, 1992. Morgan Kaufmann.
15. R. Rauh, C. Hagen, M. Knauff, T. Kuss, C. Schlieder, and G. Strube. Preferred and alternative mental models in spatial reasoning. *Spatial Cognition and Computation*, 5(2&3):239–269, 2005.
16. A. Schwering, U. Krumnack, H. Gust, and K.-U. Kühnberger. Abstraction in analogy. In *8. Fachtagung der Gesellschaft für Kognitionswissenschaft (KogWis2007)*, Saarbrücken, Germany, 2007.
17. A. Schwering, U. Krumnack, K.-U. Kühnberger, and H. Gust. Using Gestalt principles to compute analogies of geometric figures. In D. S. McNamara and J. G. Trafton, editors, *29th Annual Conference of the Cognitive Science Society (CogSci07)*, pages 1485–1490, Austin: TX, 2007.
18. A. Schwering, U. Krumnack, K.-U. Kühnberger, and H. Gust. Analogy as integrating framework for human-level reasoning. In *1st Conference on Artificial General Intelligence (AGI08)*, pages 419–423, Memphis, TN, 2008. IOS Press.
19. A. Schwering, U. Krumnack, K.-U. Kühnberger, and H. Gust. Syntactic principles of heuristic-driven theory projection. *Journal of Cognitive Systems Research*, 2009. Special Issue on Analogies - Integrating Cognitive Abilities.
20. R. Stollinski, A. Schwering, K.-U. Kühnberger, and U. Krumnack. Structural alignment of visual stimuli influences human object recognition. In *Proc. of Analogy'09*. NBU Press, 2009.

Neural Networks and Continuous Time

Frieder Stolzenburg and Florian Ruh

Hochschule Harz (FH), University of Applied Sciences, Department of Automation and Computer Sciences, 38855 Wernigerode, GERMANY, {fstolzenburg,fruh}@hs-harz.de

Abstract. The fields of neural computation and artificial neural networks have developed much in the last decades. Most of the works in these fields focus on implementing and/or learning discrete functions or behavior. However, technical, physical, and also cognitive processes evolve continuously in time. This cannot be described directly with standard architectures of artificial neural networks such as multi-layer feed-forward perceptrons. Therefore, in this paper, we will argue that neural networks modeling continuous time explicitly are needed for this purpose, because with them the synthesis and analysis of continuous and possibly periodic processes in time are possible (e. g. for robot behavior) besides computing discrete classification functions (e. g. logical boolean functions). We will relate possible neural network architectures with (hybrid) automata models that allow to express continuous processes.

Key words: neural networks; physical, technical, and cognitive processes; hybrid automata; continuous time modeling.

1 Introduction

During the last decades, the field of (artificial) *neural networks* has drawn more and more attention due to the progress in software engineering with artificial intelligence. Neural networks have been applied successfully e. g. to speech recognition, image analysis, and in order to construct software agents or autonomous robots. A basic model in the field is a multi-layer feed-forward perceptron. It can be automatically trained to solve complex classification and other tasks, e. g. by the well-known backpropagation algorithm (cf. [4, 16]). Implementing and/or learning discrete functions or behavior is in the focus of neural networks research.

Nevertheless, technical, physical, and also cognitive processes evolve in time continuously, especially if several agents are involved. In general, modeling multiagent systems means to cope with constraints that evolve according to the continuous dynamics of the environment. This is often simulated by the use of discrete time steps. In the literature, *hybrid automata* are considered for the description of systems by a mathematical model, where computational processes interact with physical processes. Their behavior consists of discrete state transitions plus continuous evolution [5]. Hybrid automata have been successfully applied especially to technical and embedded systems, e. g. for describing multi-robot behavior [2, 15]. However, a feasible procedure for learning hybrid automata does not seem to be available.

Therefore, we will at first introduce application scenarios that include complex cognitive, technical, or physical processes for the synthesis and analysis of continuous and possibly periodic systems of agent behavior (Sect. 2). After that, we briefly discuss some related works on neural networks and hybrid automata wrt. their applicability to timely continuous systems (Sect. 3). Then, we present an enhanced model of neural networks with continuous time, which we call *continuous-time neural network* (CNN) (Sect. 4), which can simulate the behavior of hybrid automata as a system that interprets periodic, continuous input and the response to that. It can also be used for periodicity detection, e. g. in speech or musical cognition. Finally, we will end up with conclusions (Sect. 5).

2 Scenarios of Agents in a Continuously Evolving Environment

Scenario 1 (deductive reasoning). *Classification tasks like e. g. image recognition or playing board games (see Fig. 1) require deductive reasoning and cognition. In this scenario, the environment is discrete (according to the classification in [16]), because there is only a limited number of distinct percepts and actions. In particular, it is not dynamic, i. e., the environment does not change over time, while the agent is deliberating.*

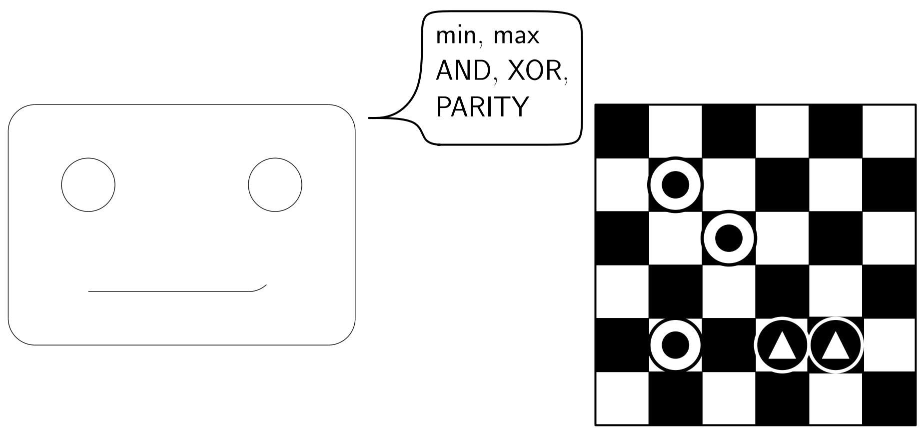


Fig. 1: Agent reasoning deductively.

Ordinary artificial neural networks allow to solve classification tasks and to express logical boolean functions for deductive reasoning directly, i. e. functions of the form $f : X \to Y$, where $X = (x_1, \ldots, x_n)$ represents the input values and $Y = (y_1, \ldots, y_m)$ the output values. Therefore, deductive reasoning can be adequately implemented by using them. Neural networks in general consist of an interconnected group of nodes, called units, which are programming constructs mimicking the properties of biological neurons. Standard neural networks such as multi-layer feed-forward perceptrons have a restricted architecture. There, we have only three or more layers of units: input, hidden, and output units, which are connected only in this order [4, 16]. It is well-known [4] that every continuous function that maps intervals of real numbers to some output interval of real numbers can be approximated arbitrarily closely by a multi-layer perceptron with just one hidden layer, if we have sigmoidal activation functions, i. e. bounded, nonlinear, and monotonously increasing functions, e. g. the logistic function or the hyperbolic tangent (tanh). Multi-layer networks use a variety of learning techniques, the most popular being backpropagation. In general, any declarative logical operation can be learned by such a network. However, many real cognitive or physical processes depend on time, as in the following scenario.

Scenario 2 (robot at a conveyor belt). *Let us consider a robot that has to perform a specific routine again and again, e. g. grabbing a brick from a conveyor belt (see Fig. 2). Fig. 3 shows the height h of the robot arm depending on the time t. For the ease of presentation, we abstract from releasing the box, moving the arm down and grabbing the next one here. In addition, we assume, that the agent knows the duration T of each episode.*

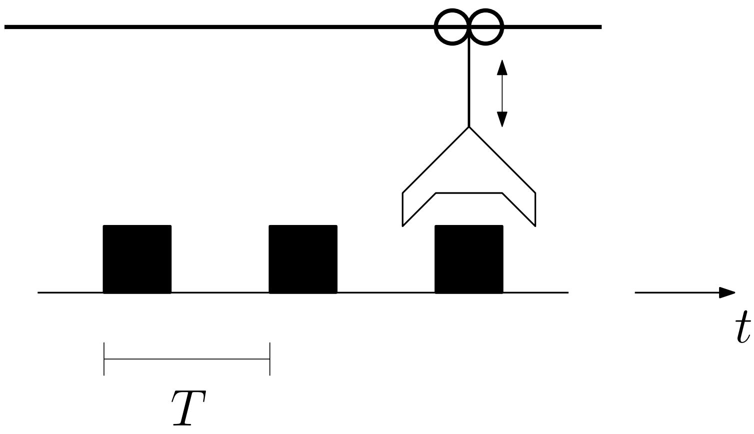


Fig. 2: An example robot arm, picking boxes on a conveyor belt.

This scenario requires the solution of several tasks. In particular, continuous behavior of the robot agent must be producible for grabbing the bricks continuously and periodically. Clearly, for synthesis and also for analysis of processes or behavior, modeling the time t explicitly is necessary, because we have to model mappings of the form $X(t) \mapsto Y(t)$. For Scenario 2, we assume that the robot has to move its arm up and down within a fixed time interval T. This leads to a sawtooth function, if we consider the dependency from time (see Fig. 3). Such behavior can be expressed easily by an automaton model, especially hybrid automata [5] (see Sect. 3). However, the procedure with hybrid automata mainly is a knowledge-based approach. They cannot be learned easily by examples as e. g. neural networks.

While clearly Scenario 1 can be specified directly with ordinary neural networks, Scenario 2 requires to model the time t somehow. This can be achieved by discretizing time, i. e. by considering input values at different discrete time points, $t, t-1, \ldots, t-T$ for some time horizon T. Then, we may use $x_i(t), x_i(t-1), \ldots, x_i(t-T)$ with $1 \leq i \leq n$

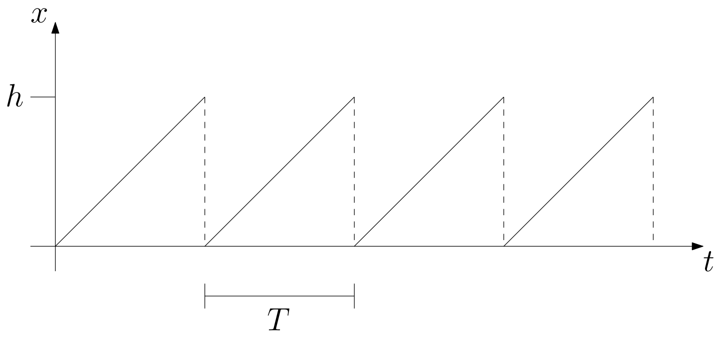


Fig. 3: The sawtooth function for the height of the robot arm, assuming that it can lower the arm in zero time.

as input values. But this procedure has several disadvantages: It increases the number of input units significantly, namely from only n to $(T+1)\cdot n$. In addition, it is not clear in this case, what granularity and past horizon of discrete time should be used.

Therefore, a presentation by (enhanced) neural networks seems to be a good idea, that makes use of the (continuous) time t as additional parameter, at least implicitly. In this context, oscillating periodic behavior must be producible, even if the input X remains static, i. e. constant. For instance, once switching on a robot, i. e. change one input unit from 0 to 1, the periodic behavior should hold on, until the input unit is switched off again (cf. [11]). Therefore, we will introduce units, whose input may be a fixed value, but whose output yields a sinusoid (see Sect. 4, Def. 2). By this, we can express periodic behavior in time by neural networks. Furthermore, we should be able to analyze behavior and to detect period lengths, which we formulate now:

Scenario 3 (behavior and periodicity analysis). *Before a robot is able to behave adequately in a dynamic environment, it has to analyze its environment, e. g. to find out the duration of an episode of the robot at the conveyor belt (Scenario 2, Fig. 2), i. e. the period length in time. This task also appears in speech and musical harmony recognition, as illustrated in Fig. 4.*

Since cognitive science may be defined as the study of the nature of intelligence and thus of intelligent behavior, drawing on multiple disciplines, including psychology, computer science, linguistics, and biology, we consider behavior and periodicity analysis here, because it is obviously an important aspect of intelligence. In particular, this holds for scenarios with several agents and/or agents in dynamically changing environments, because it is the basis for coordination and synchronization of (periodic) behavior of agents. For instance, finding the way through a dynamic environment with many obstacles and crossing traffic of a specific frequency, requires synchronization among agents, including periodicity analysis.

One possible key for determining overall period lengths is auto-correlation, i. e. the cross-correlation of a signal with itself. It can be mathematically defined by convolution (cf. [1], see also Sect. 3.3). However, we choose another formalization here: We simply assume that a unit of a CNN (cf. Def. 2) can delay its incoming signals for a specific

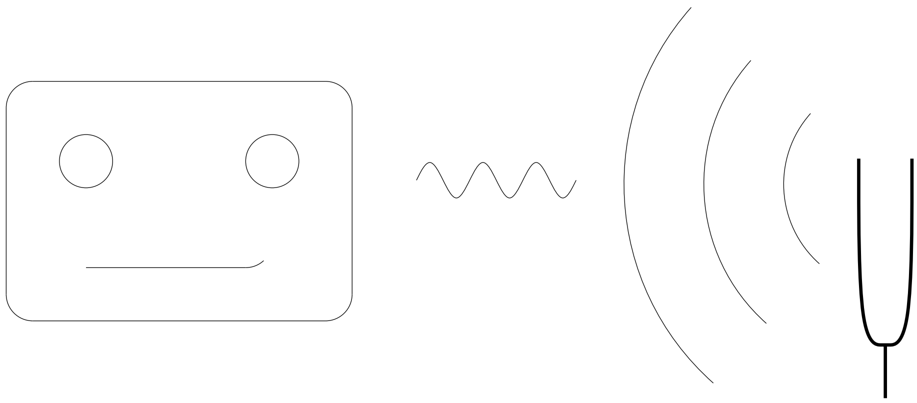


Fig. 4: Agent analyzing periodic episodes in the environment.

time delay δ. Then, a comparison of the original signal with the delayed one yields the appropriate result. Eventually, biological neural networks, e. g. the hearing system in the brain, seem to be able to delay signals [8]. Before we present the CNN model in more detail (Sect. 4), let us first discuss related works that are more or less suitable for modeling the scenarios introduced here.

3 Neural Networks, Hybrid Automata, and Continuous Time

The underlying idea that the original model of artificial neural networks tries to capture is that the response function of a neuron is a weighted sum of its inputs, filtered through a nonlinear, in most cases sigmoidal function

$$y = h(\sum_{i=1}^{n} w_i x_i)$$

where h is the activation function, e. g. the logistic function $(1/(1-\exp(-x)))$. Fig. 5 shows the general scheme of a unit of a neural network with the inputs $x_1, \ldots, x_n$ and one output y. Each incoming and also the outgoing edge is annotated with a weight w_0.

3.1 Fourier Neural Networks

An obvious paradigm to combine neural networks with periodic input are so-called *Fourier neural networks* [12, 17]. They allow a more realistic representation of the environment by considering input oscillation for implementing and/or learning discrete functions or behavior. From a neurophysiological point of view, they appear to be closer to reality, because they model the signals exchanged between neurons as oscillations, making the model to better agree with discoveries made in neurobiology. In [17], the output function of a neuron is defined as $f(X) = \int_D c(X)\,\varphi(X,Y)\,dY$, where $\varphi(X,Y)$ is some characteristics of the input X, weighted by the coefficients $c(X)$, i. e., we get a weighted integral (replacing the sum from above) of the inputs and their characteristics. However for the computation, a discretized model given by the equation

$f^d(x) = h\left(\sum_i c_i \Pi_{j=1}^n \cos(\omega_{ij} x_j + \varphi_{ij})\right)$ is used with the sigmoidal logistic function h from above in order to obtain output in the interval $[0;1]$.

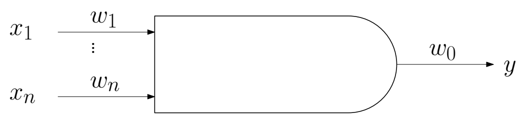


Fig. 5: A unit of a neural network (scheme).

In [12], Fourier neural networks with sinusoidal activation function $h(x) = c\sin(ax + b)$ are considered. Additional non-linear (sigmoidal) activation functions are not needed to express arbitrary functions in this case. In fact, the sine function has the characteristics of a sigmoid function in the interval $[-\pi;\pi]$. All logical operators with two inputs (Scenario 1) can be implemented in this framework (see Fig. 6) by only *one* single unit with sinusoidal activation function, in contrast to the standard neural networks with other, monotonously increasing activation functions. However, learning these neural networks is a difficult task, because sinusoidal activation functions are non-monotonous. In addition, continuous time is not modeled explicitly in this approach. fstolzenburg@hs-harz.de

function	# inputs	a	b	c	meaning
AND	2	$\frac{\pi}{4}$	$-\frac{\pi}{4}$	$\sqrt{2}$	logical conjunction
XOR	2	$\frac{\pi}{2}$	$-\frac{\pi}{2}$	1	exclusive or
ODD	n	$\frac{\pi}{2}$	$(n-1)\frac{\pi}{2}$	1	odd parity, returns 1 iff an odd number of inputs is 1

Fig. 6: Implementing logical functions for one Fourier neural network unit with activation function $c\sin(ax+b)$. The Boolean values *true* and *false* are represented by $+1$ and -1, respectively.

3.2 Continuous Neural Networks

[9] introduces neural networks with an uncountable number of hidden units. While such a network has the same number of parameters as an ordinary neural network, its internal structure suggests that it can represent some smooth functions more compactly. [9] presents another approach for neural networks with an uncountable number of units, where the weighted summation of input values is replaced by integration. Because of this, they are called continuous neural networks. However, continuous time and hence temporal processing is not modeled explicitly there, which is the primary goal of this paper.

In [10], neural networks are used in a nonlinear system identification algorithm for a class of nonlinear systems. The algorithm consists of two stages, namely preprocessing the system input and output and neural network parameter estimation. However, first and foremost, it is only applicable to the analysis of control systems with a special structure.

3.3 Finite Impulse Response Perceptrons

Temporal processing in neural networks means to deal with dynamic effects and to introduce time delays in the network structure [4]. Therefore, in the *finite-duration impulse response (FIR) model*, temporal processing is realized by a linear, time-invariant filter for the synapse i of a neuron j. Its impulse response $h_{ji}(t)$ depends on a unit impulse at time $t = 0$. Typically, each synapse in the FIR model is causal and has a finite memory, i. e. $h_{ji}(t) = 0$ for $t < 0$ or $t > \tau$, with the memory span τ for all synapses. The response of a synapse can be defined as the convolution (auto-correlation) of its impulse response with the input $x_i(t)$. Thus, we can express the output as $h_{ij}(t) * x_i(t) = \int_{-\infty}^{t} h_{ji}(u) x_i(t-u) du$. The net activation potential over all p synapses, with threshold θ_j, is given by $v_j(t) = \sum_{i=1}^{p} \int_0^{\tau} h_{ji}(u)\, x_i(t-u)\, du - \theta_j$, where the overall output is the sigmoidal nonlinear logistic activation function (see above). With this, an artificial neuron can represent temporal behavior. The FIR multi-layer perceptron, with its hidden and output neurons based on this FIR model, has been applied for adaptive control, dynamic system identification, and noise cancellation. Once trained, all synaptic weights are fixed. Then, the network can operate in real time.

Instead of the FIR model, where time is simulated by additional copies of a neuron for different times (cf. Sect. 2, Scenario 2), *real-time recurrent networks* (cf. [4]) are designed by using a common neural model, where the temporal processing is realized by the feedback of the network.

3.4 Hybrid Automata

Another model that allows to model discrete and dynamic changes of its environment and hence continuous time are *hybrid automata*, a combination of Moore and Mealy automata [5]. A hybrid automaton is a mathematical model for describing systems, where computational processes interact with physical processes. In contrast to simple finite state automata, well-known in computer science [3, 13], their behavior is stated not only by discrete state transitions, but also by continuous evolution. Hybrid automata consist of a finite set of states and transitions between them. Thus, continuous flows within states and discrete steps at the transitions are possible. If the state invariants do not hold any longer, a discrete state change takes place, where a jump condition indicates which transition shall be used. Then, a discrete step can be done, before the next state is reached. States are annotated with invariants and flow conditions, which may be differential equations. There, the continuous flow is applied to the variables within the state invariants. Thus, the behavior of the robot in Scenario 2 can be described as shown in Fig. 7. Hybrid automata, however, are not well-suited for mapping continuous input with periodic behavior. In addition, (hybrid) automata cannot be learned easily by examples as e. g. neural networks.

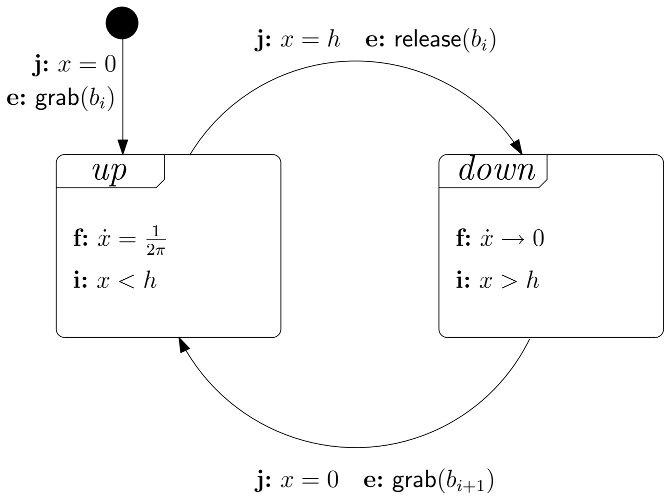


Fig. 7: Hybrid automaton for the robot arm (Scenario 2).

3.5 Central Pattern Generators

For Scenario 2, oscillating, periodic patterns must be generable. This can be achieved, if a single unit is able to oscillate spontaneously, as we will assume here (cf. Def. 2). Alternatively, recurrently connected units can trigger each other, inducing periodic patterns. Such a system is called *central pattern generator* (CPG). They can be defined as neural networks that can endogenously (i.e. without rhythmic sensory or central input) produce rhythmic patterned outputs [6] or as neural circuits that generate periodic motor commands for rhythmic movements such as locomotion [7]. CPGs have been shown to produce rhythmic outputs resembling normal rhythmic motor pattern production even in isolation from motor and sensory feedback from limbs and other muscle targets. To be classified as a rhythmic generator, a CPG requires: two or more processes that interact such that each process sequentially increases and decreases, and that, as a result of this interaction, the system repeatedly returns to its starting condition.

4 Towards Continuous-Time Neural Networks

We will now define *continuous-time neural networks* (CNN). With them, we are capable of modeling the three general scenarios, introduced in Sect. 2. At first glance, they are very similar to standard neural networks, because they also consist of an interconnected group of units. In fact, a CNN degenerates to an ordinary neural network, if the extended functionality is not used. We distinguish several types of units (see Def. 1 and 2).

Definition 1 (input and output units, on-neurons). *In a CNN, there may be one or more* input and output units. *Input units do not have any incoming edges, while output units do not have any outgoing edges. In the following, we restrict our attention to networks with only one output unit. The values of the input units $x_1(t), \ldots, x_n(t)$ and of the output unit $y(t)$ depend on the time t. There may also be so-called* on-neurons*, i. e.*

units without incoming edges, yielding a constant output c, independent from the actual time t.

In our model, as in standard neural networks, we assume that the input value of a unit j is a weighted sum of the incoming values, and we have a nonlinear activation function. But in addition, we have two further optional components in each unit (for integration over time and for enabling oscillation) that may be switched on or off. Furthermore, inputs may be delayed or not. This is summarized in the following definition, leading to a unit with up to four stages, called *sub-units* in the sequel:

Definition 2 (continuous neural network unit). *In general, a CNN unit computes its output value $y(t)$ from its input values $x_1(t),\dots,x_n(t)$, which may be the overall input values of the network or the output values of immediate predecessor units, in four steps. Each step yields the value $y_k(t)$ with $1 \leq k \leq 4$, where $y(t) = y_4(t)$. For $k \geq 2$, the respective sub-unit may be switched off, which means that $y_k(t) = y_{k-1}(t)$.*

1. ***summation:*** *The input value of the unit is the sum of the incoming values $x_i(t)$ with $1 \leq i \leq n$, each weighted with a factor w_i and possibly delayed by a time amount δ_i, which is 0 by default:*

$$y_1(t) = \sum_{i=1}^{n} w_i \cdot x_i(t - \delta_i)$$

2. ***integration:*** *In certain cases, the integrated activity, i. e. the average signal power, is useful. Therefore, we introduce an optional integration process, which is switched off by default.*

$$y_2(t) = \sqrt{\frac{1}{\tau}\int_{t-\tau}^{t} y_1(u)^2\,du}$$

Note that, for $\tau \to 0$, we have $y_2(t) = |y_1(t)|$, i. e., the unit is switched off for positive values. If it is switched on, we take $\tau \to \infty$ by default. Alternatively, the statistical variance of $y_1(t)$ could be used here.

3. ***activation:*** *In order to be able to express general, non-linear functions, we need a non-linear activation function (cf. [4]). Instead of the often used logistic function (cf. Sect. 3), we use the hyperbolic tangent here, because $\tanh(x) \approx x$ for small x and the range of the hyperbolic tangent is $[-1;+1]$, which corresponds to the range of sinusoidal periodic functions. We define:*

$$y_3(t) = \frac{\tanh(\alpha \cdot y_2(t))}{\alpha}$$

We make use of a factor α that retains these properties here. By default, $\alpha = 1$. For $\alpha \to 0$, the sub-unit is switched off.

4. ***oscillation:*** *The unit can start to oscillate with a fixed (angular) frequency ω:*

$$y_4(t) = y_3(t) \cdot \cos(\omega t)$$

This corresponds to amplitude modulation of the input signal. In principle, other types of modulation, e. g. frequency or phase modulation, would be possible, but this is not considered here. For $\omega = 0$, this sub-unit is switched off.

With this type of units, all scenarios, introduced in Sect. 2, can be implemented. If the integration and the oscillation sub-unit is switched off, the functionality of the unit is identical with that of standard neural network units (cf. Sect. 3 and [4, 16]). Hence, all logical boolean functions (Scenario 1) can be expressed easily, of course, in contrast to Fourier neural networks, generally with hidden units. Everything that can be expressed by an ordinary neural network can be expressed by a CNN, because the former one is a special case of a CNN.

Scenario 2 can be implemented with several oscillating units, i. e. $\omega_k \neq 0$, because it is known from the study of Fourier series, that arbitrary periodic functions can be written as the sum of simple waves represented by sines and cosines. For the sawtooth-like graph (Fig. 3), we have $f(x) = \frac{h}{2} - \frac{h}{\pi}\sum_{k=1}^{\infty}\frac{1}{k}\cdot\sin(\frac{2\pi}{T}kx)$. The latter sum may be approximated by the first n summands, which can be expressed by n oscillating CNN units (see Fig. 8).

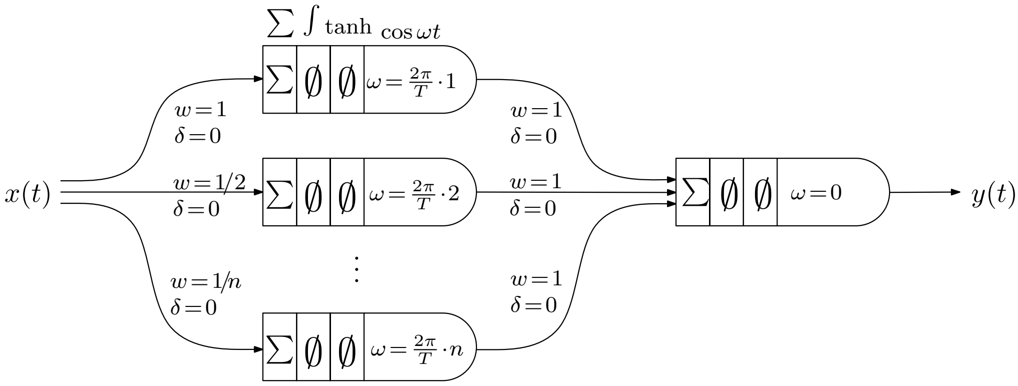


Fig. 8: Network with several oscillating units for Scenario 2. Sub-units, that are switched off, are marked with $\emptyset$.

In Scenario 3, we have to find out the period length T of a task automatically from a complex signal, e. g. the duration of an episode of the robot at the conveyor belt (Scenario 2, Fig. 2). For this, consider the function $x(t) = \cos(\omega_1 t) + \cos(\omega_2 t)$, whose overall period length depends on the ratio ω_2/ω_1. Let $\omega_1 = 2\pi$ and $\omega_2 = \sqrt{2}\omega_1$. The corresponding graph for $x(t)$ is shown in Fig. 9. In order to determine the overall period length, we must be able to find out the so-called missing fundamental frequency, i. e., we have to find a time duration T such that $x(t) - x(t-T)$ becomes zero. Applying the least squares method, this could be turned in finding the minima (almost zeros) of $1/T\int_0^T (x(u) - x(u-T))^2\,du$. Therefore, we overlap the original signal ($\delta = 0$, $w = 1$) with a phase-shifted and inverted copy of itself ($\delta = T$, $w = -1$), which yields an effect of *comb filtering* (cf. [8]).

Fig. 10 shows the graph for the square root of the latter integral in dependency from T, which can be achieved by switching on the integral sub-unit. It has minima near 5 and 12 (and also near 7 and 10) which alternatively can be derived by continued fraction development of the ratio ω_2/ω_1 [14, 18]. Thus, the corresponding CNN unit yields approximately constant output wrt. t, namely the values shown in the graph in Fig. 10, where small values near 0 indicate periodicity. This procedure allows us to express analysis of periodic behavior as desired.

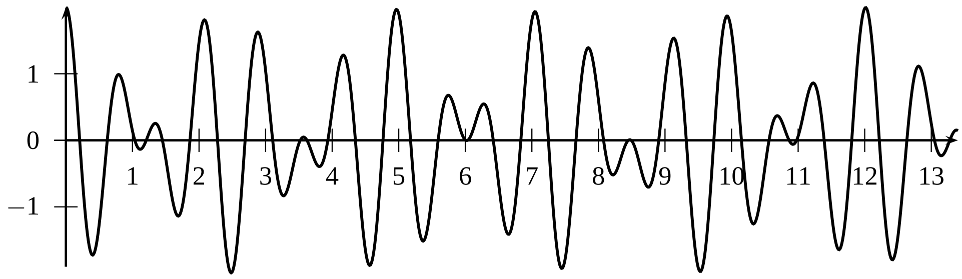


Fig. 9: Complex periodic signal $x(t) = \cos(\omega_1 t) + \cos(\omega_2 t)$ with $\omega_2/\omega_1 = \sqrt{2}$.

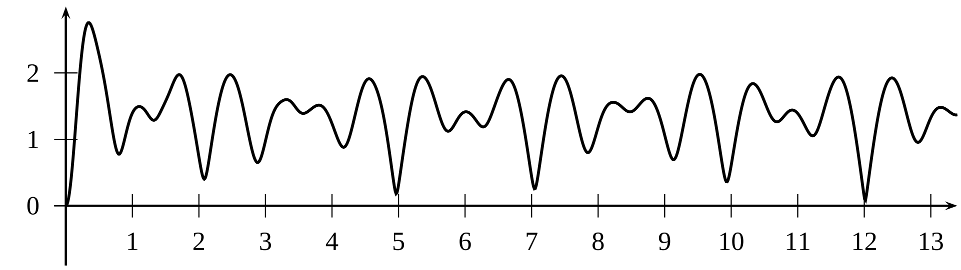


Fig. 10: Periodicity analysis for complex signal $x(t)$. The graph shows the output of the comb-filtered signal in dependency from the delay in time T, checking the period length of the overall signal, with main minima at $T = 5$ and $T = 12$. It is constant wrt. the current time t.

5 Conclusions

In this paper, we sketched ongoing work on neural networks with continuous time. These networks can support the modeling of behavior synthesis and analysis in robotics and for cognitive systems. For arbitrary continuous, periodic input, the robot or the agent in general has to react continuously and within a certain time interval. Hence, complex, physical and/or cognitive processes can be modeled adequately by a CNN. A CNN without recurrence and constant values for the angular frequencies ω_k in the oscillation sub-units and switched-off integration sub-units correspond to standard neural network units in principle. Thus, the classical backpropagation method can be employed for learning a CNN from examples, where a set of input and output values must be given for different time points t. Therefore, future work will implement this theory. We intend to do this on a concrete autonomous robot platform, namely a quadrocopter, i. e. flying robots with four propellers. Analysis of the network is also an important part and will be investigated in further detail, too.

References

[1] M. Ebeling. Neuronal periodicity detection as a basis for the perception of consonance: A mathematical model of tonal fusion. *Journal of the Acoustical Society of America*, 124(4):2320–2329, 2008.

[2] U. Furbach, J. Murray, F. Schmidsberger, and F. Stolzenburg. Hybrid multiagent systems with timed synchronization – specification and model checking. In M. Dastani, A. El Fallah Seghrouchni, A. Ricci, and M. Winikoff, editors, *Post-Proceedings of 5th International Workshop on Programming Multi-Agent Systems at 6th International Joint Conference on Autonomous Agents & Multi-Agent Systems*, LNAI 4908, pages 205–220, Honolulu, 2008. Springer, Berlin, Heidelberg, New York.

[3] A. Gill. *Introduction to the Theory of Finite-State Machines*. McGraw-Hill, New York, 1962.

[4] S. Haykin. *Neural Networks: A Comprehensive Fondation*. Macmillan College Publishing, New York, 2nd edition, 1994.

[5] T. Henzinger. The theory of hybrid automata. In *Proceedings of the 11th Annual Symposium on Logic in Computer Science*, pages 278–292, New Brunswick, NJ, 1996. IEEE Computer Society Press.

[6] S. L. Hooper. Central pattern generators. *Embryonic Encyclopedia of Life Sciences*, 1999.

[7] A. D. Kuo. The relative roles of feedforward and feedback in the control of rhythmic movements. *Motor Control*, 6:129–145, 2002.

[8] G. Langner. Temporal processing of periodic signals in the auditory system: Neuronal representation of pitch, timbre, and harmonicity. *Zeitschrift für Audiologie*, 46(1):8–21, 2007. In German.

[9] N. Le Roux and Y. Bengio. Continuous neural networks. In *Proceedings of 11th International Conference on Artificial Intelligence and Statistics*, San Juan, Puerto Rico, 2007.

[10] Y. Liu and J. J. Zhu. Continuous-time nonlinear system identification using neural network. In *Proceedings of American Control Conference*, pages 613–618, Seattle, USA, 2008.

[11] N. M. Mayer, J. Boedecker, K. Masui, M. Ogino, and M. Asada. HMDP: A new protocol for motion pattern generation towards behavior abstraction. In U. Visser, F. Ribeiro, T. Ohashi, and F. Dellaert, editors, *RoboCup 2007: Robot Soccer World Cup XI*, LNAI 5001, pages 184–195, Atlanta, GA, 2008. Springer, Berlin, Heidelberg, New York.

[12] L. Mingo, L. Aslanyan, J. Castellanos, M. Diaz, and V. Riazanov. Fourier neural networks: An approach with sinusoidal activation functions. *Information Theories and Applications*, 11:52–55, 2004.

[13] Object Management Group, Inc. *UML Version 2.1.2 (Infrastructure and Superstructure)*, November 2007.

[14] C. D. Olds. *Continued Fractions*, volume 9 of *New Mathematical Library*. L.W. Singer Company, 1963.

[15] F. Ruh and F. Stolzenburg. Translating cooperative strategies for robot behavior. In G. J. Nalepa and J. Baumeister, editors, *Proceedings of 4th Workshop on Knowledge Engineering and Software Engineering at 31st German Conference on Artificial Intelligence*, CEUR Workshop Proceedings 425, pages 85–96, Kaiserslautern, 2008.

[16] S. Russell and P. Norvig. *Artificial Intelligence – A Modern Approach*. Prentice Hall, Englewood Cliffs, NJ, 1995.

[17] A. Silvescu. Fourier neural networks. In *Proceedings of International Joint Conference on Neural Networks (IJCNN 1999)*, pages 488–491, Washington D. C., 1999.

[18] F. Stolzenburg. A periodicity-based theory for harmony perception and scales. In *Proceedings of 10th International Society for Music Information Retrieval Conference*, Kobe, Japan, 2009. To appear.

Bayesian Identification of Problem-Solving Strategies for Checking the ACT-R/Brain-Mapping Hypothesis

Claus Möbus and Jan Charles Lenk

Learning and Cognitive Systems, Department of Computing Science
Carl von Ossietzky Universität Oldenburg
26111 Oldenburg, Germany
{claus.moebus|jan.lenk}@uni-oldenburg.de,
WWW home page: http://www.lks.uni-oldenburg.de/

John R. Anderson proposed a correspondence between ACT-R modules and brain regions (brain mapping hypothesis). To check this conjecture we plan to compare model-generated blood-oxygen-level dependent (BOLD) signal curves with BOLD curves obtained from functional Magnetic Resonance Imaging (fMRI) scans. In contrast to Anderson's studies our subjects were not urged to follow a *single* strategy but construct their *personal* strategy within a constraint-based strategy space. So, the mapping hypothesis has to be checked strategy-specific. The identification of strategies was difficult because subjects were not able to identify their own in a retrospective manner. So we used Response-Time (RT) data in combination with a Bayesian Belief Net to identify *personal* problem solving strategies without using fMRI data for checking the mapping hypothesis.

Predicting Changes: A Cognitive Model for Dynamic Stocks and Flows

Felix Steffenhagen, Marco Ragni, Ivo Chichkov, Andreas Klein

University of Freiburg, Center of Cognitive Science
D-79098 Freiburg, Germany
{felix|ragni}@cognition.uni-freiburg.de

We briefly report about experimental investigations we conducted for the so-called dynamic stocks and flows task (DSF) and present a cognitive model to replicate human's behavior. The goal in the DSF task is to maintain a certain level of water in a tank under the influence environmental flows which depend on unknown dynamics. Our findings are complemented by an analysis of recent experimental data from the literature. The results are integrated in a cognitive model with which we are able to reproduce and predict human behavior for this task.

On Optimization of the Interface between Subsymbolic and Symbolic Representations and the Symbol Grounding Perspective

Oliver Kramer

Department of Computer Science
Technische Universität Dortmund
44227 Dortmund, Germany
oliver.kramer@tu-dortmund.de

From the point of view of an autonomous agent the world consists of high-dimensional dynamic sensorimotor data. Translated into symbols the data is easier to handle for cognitive processes. I propose to formulate the interface design between subsymbolic and symbolic representations as global optimization problem. The objective becomes to maximize the success of the overlying cognitive algorithm. From the formulation of the interface between subsymbolic and symbolic representations as optimization problem various consequences arise. The optimization process will improve the performance of interfaces and hence the success in solving cognitive tasks. Learning becomes a two-level optimization problem: interface learning and learning on the symbolic level.

The translation of high-dimensional subsymbolic data into symbols are tasks that are well known in data mining and machine learning under the terms classification, clustering and dimensionality reduction. Many symbol grounding related work exclusively concentrated on neural networks in the past, perhaps due to a historical affinity to connectionist models. To overcome the restriction this Section shows the relation between symbol grounding and machine learning: assigning unknown objects to known concepts is known as classification, grouping objects is known as clustering, finding low dimensional representations for high-dimensional data is denoted as dimension reduction. Having this optimization nature in mind, the creator of a cognitive system can invest more time into careful tuning and control of interface properties. Although the flexibility of most current dimensionality reduction and clustering methods is quite high, in the future the creators of artificial intelligent systems may spend more effort in the development of adaptive and evolvable interface algorithms, in particular in online-scenarios as conditions, e.g. the structure of high-dimensional data, may change significantly in time. A solely mathematical and algorithmic formulation only allows a narrow view on the optimization problem.

From the point of view of the symbol grounding problem the meaning of a symbol arises implicitly from this optimization formulation and fulfills the zero semantical commitment condition. The binding of the symbol grounding to the objective of the acting autonomous agent leads to the fulfillment of the zero semantical commitment condition as neither internal nor external knowledge, except the objective of the autonomous agent and its learning algorithms are explicitly integrated into the agent. Interface and optimization algorithm are computational and procedural resources.

Sensorimotor Self-Motivated Cognition

Martin V. Butz

University of Würzburg, Department of Psychology III
Röntgenring 11, 97070 Würzburg, Germany
butz@psychologie.uni-wuerzburg.de
http://www.coboslab.psychologie.uni-wuerzburg.de/

Abstract. Over the last years, sensorimotor codes, which are often termed schemata in their symbolic form, are becoming increasingly accepted as a fundamental structural code for biological cognitive systems. In their most embodied form, sensorimotor structures encode the correlations of behavioral activities with their immediate perceptual consequences. Behavior can thereby be a particular muscular contraction, more elaborate motor synergistic muscular activity, or even complex (trained) dynamic movements. Perceptual consequences may be very immediate proprioceptions, more elaborate sensory changes, or even complex perceptual dynamics. Besides their immediate behavioral relevance, sensorimotor codes glue sensory and behavioral codes together forming cognitive maps that enable the execution of complex goal-directed behavior. Together, sensorimotor bodyspaces are formed in which distances in space are not sensorially encoded but sensomotorically, that is, distances represent the motor effort necessary to transfer one sensory perception into another one. To create intrinsically motivated behavioral systems, motivations can be added to these modular bodyspace representations. We show that different types of motivations need to be distinguished for an effective design of interactively multi-motivated systems. Moreover, we show that such designs can be easily integrated into sensorimotor bodyspace representations. In conclusion, we propose that motivations may not only be necessary to induce goal-directed behavior, but they may also be a highly important component for shaping higher level cognitive modules.

Key words: Sensorimotor Codes, Cognitive Maps, Bodyspaces, Motivations, Homeostasis, Self-Motivated Reinforcement Learning, Layered Cognition

1 Sensorimotor Bodyspaces

From a high-level conscious view-point, we often perceive ourselves in the surrounding space from a somewhat allocentric, abstract perspective. During communication, we might localize ourselves in the environment as currently being, for example, in a certain room, a city, or country. If relevant for the conversation, however, we turn to more ego-centric information such as, for example, attending a certain event, facing a certain object or person, listening to a particular musical piece, or watching a certain program on TV. Thus, our self-perception can have many perspectives and is integrated into various, more-or-less egocentric, points of view.

What constitutes these perspectives? That is, in which forms of representation are such perspectives embedded? Various research directions suggest that the basis for our spatial perceptions are sensorimotor codes, which are necessarily purely egocentric. Wolff reviewed eye movement studies and concluded that a spatial representation has to have a dominant, sensorimotor component [1]. Studies in reaching movements led to the suggestion that multiple coupled forward-inverse models exist for motor control [2]. Experimental evidence is available that shows that dynamic sensorimotor encodings transfer bidirectionally between different tasks and different sensory modalities. For example, rotations can transfer bidirectionally between tracking to pointing [3]—which led the authors to the conclusion that the investigated adaptation mechanism lies in a common dynamic code that can transfer between categories. Similarly, dynamic hand movements can transfer between hands [4]. Thus, dynamic movements are partially encoded by a common code, as, for example, proposed in the theory of event coding (TEC) [5] and anticipatory, sensorimotor structures, or schemata, constitute the basis of this coding scheme [6, 7].

Interestingly, from a much more computationally oriented perspective, it was shown that correlations between sensory and motor codes may reveal the dimensionality in which interactions take place [8]. The authors show that the number of components represented in a correlation mapping allows the deduction of the dimensions of physical space. An overarching perspective of the body integrated into multimodal, highly modular sensorimotor bodyspaces can be found elsewhere [9].

As a consequence, sensorimotor bodyspaces do not encode the space purely sensory, but space is represented with various sensorimotor codes. This implies that distances in space are represented motor-dependently. And in fact, the conscious representation of spatial distances depends on the motor effort necessary to bridge the questioned distance [10]. It was even shown that tool-use can alter the distance perception as well—especially when tool-use is intended and an object is in reach with the tool but not without it [11].

In neuroscience studies with monkeys, it was even shown that single premotor cortical neurons with mirror-neuron properties distinguish between reachable and non-reachable locations in space [12]. Moreover, the structure of the monkey cortex was shown to be partitioned into various ethologically-relevant functions, besides limb topology and simple movement typological distinctions [13]. The different aspects are encoded with somewhat distinct regional, neural population codes. A single neuron in such a code may control a different facet of the encoded overall behavior and can be context-dependently modulated. Similar population codes are also found in the parietal cortex, where peripersonal body spaces are encoded [14], which surround the body and the reachable space with distributed neural codes. Thus, population codes are a fundamental encoding scheme of the brain. Each neuron in a population code represents a particular aspect of the code, such as a particular arm constellation in motor cortex or a particular hand-relative object location in parietal cortex.

Although most of the studies above investigate body-part-relative movements and representations (especially hand and arm but also eyes) rather than whole body movements, there are several indications that also spatial representations that have whole body movements as their motor-code use similar sensorimotor encoding strategies. In the hippocampus, information converges for the formation of episodic memory. During movements through space, place cells and head-direction cells were localized (among others) in the rat's hippocampus and were mimicked by associative population codes [15]. Various indications now suggest that the hippocampus is not only involved in the integration of allocentric and egocentric representations but it also plays an important role for the goal-oriented execution of behavior in space [16, 17].

In sum, sensorimotor highly interactive and dynamic bodyspace representations are omnipresent in the brain. They control simple reaching movements but also elaborate body movements and categorical movements, which are each encoded with modular, interactive sub-populations of neurons. Each sub-population covers a particular behavioral task or aspect of interaction with the environment, including manipulating behaviors as well as navigating behaviors. These representations however do not only serve as spatial representations and immediate behavioral control components, but also appear to constitute the basis for even higher forms of cognitive representations, leading eventually to complex social interactions, language, and abstract thought capabilities [18, 19].

2 Sensorimotor Models

Over the recent years, our research group has developed several sensorimotor models that are self-organized and developed for goal-directed behavioral control. Two models will be shortly reviewed here.

The sensorimotor redundancy resolving architecture SURE_REACH [20–22] is a psychologically and neuroscientifically plausible model of arm reaching behavior. It consists of two population codes that interactively represent and control the movement to hand locations or arm postures in the reachable space. An associative, inverse kinematics mapping correlates hand locations with redundant arm postures (one-to-many mapping) and a sensorimotor model self-associates arm postures motor-dependently. The latter essentially forms sensorimotor connections between behaviorally close postures, where each connection stores the motor vector that is necessary to reach the one posture from the other.

It was shown that this representation may be regarded as a neural implementation of the posture-based motion planning theory [23] with the additional capabilities of incorporating anticipated subsequent end-state priorities while reaching for a current target, the flexible adaptation of arm trajectories for, e.g., obstacle avoidance, and the inherent closed-loop control capability. Most recently, this architecture was also successfully applied for the control of a dynamic arm system in a realistic, physical 3D simulation environment [24].

While the inverse kinematics and sensorimotor mapping is learned in the original SURE_REACH implementation [20], the population codes were uniformly distributed in hand-location and arm-posture space. To overcome this shortcoming, we enhanced self-organizing neural network techniques to be able to connect perceptual spaces motor-dependently. The time-growing neural gas network (TGNG) grows a population code that covers a particular space, while the neural connectivity is motor-dependent [25]. It was shown that the resulting representation implicitly encodes motor-dependent distances in the explored space. So far, TGNG was applied to realize goal-directed robot movements in various maze-like environments. However, in principle the TGNG approach could also grow the population codes utilized in SURE_REACH.

Goal-directed behavior is realized in both systems by model-based reinforcement learning mechanisms, which is essentially discrete dynamic programming realized within the population encodings [26, 20, 25]. Given a particular external goal activation a_i^e of neuron i in the population code, the activity is propagated by

$$a_i \leftarrow \max \left\{ a_i^e ; \max_j \left[\gamma a_j \right] \right\}, \tag{1}$$

where a_i denotes the current activity of neuron i and index j iterates over all neurons j that are connected to neuron i via sensorimotor connections. Given the system state (such as the posture or location) is currently represented by a neuron i (usually a set of neurons represents the state of the system), then the behavior is determined by the motor activity that is stored in the sensorimotor connection that connects to the most activated neuron j, that is, $\texttt{arg}\ \max_j a_j$.

In sum, two sensorimotor population-encoded models exist and can be applied for the flexible, goal-directed control of arm-movements and embodied navigation tasks. In the following, we show that these encodings are highly suitable for the incorporation of motivation-based constraints and goal activations.

3 Self-Motivated Behavior

Until recently, the utility of the introduced population encodings was shown due to their (1) psychological and neuroscientific validity and (2) their capability to plan and control flexible, goal-directed behavior. However, for the design of an autonomous cognitive system, goals and constraints need to be self-activated. Thus, we now give an overview of how such self-motivated activities may be included in these systems. We essentially propose that the system should strive for inner homeostatic states, which may be encoded in a reservoir framework. These states may represent the internal needs of the particular system, such as hunger or thirst, as well as even more abstract homeostatic needs, such as the urge for safety, which can result in a scared system, or knowledge discovery, which can yields a curious system. As proposed elsewhere [27], we distinguish between these two types of motivations by terming the former *consummatory motivations* and the latter *property-based motivations*.

3.1 Homeostatic Reservoirs

In a similar vein to other recent publications [28, 29], we proposed to use reservoir encodings to reflect the actual needs of a system [27]: Each reservoir x can be represented by an internal state value $\sigma_x(t) \in [0, 1]$, which encodes how full the reservoir is at time t. Moreover, each reservoir may have an associated update function $\phi_x : [0, 1] \rightarrow [0, 1]$ and a weighting function $\omega_x : [0, 1] \rightarrow [0, 1]$. The update function specifies the change in the reservoir level over time given current interactions with the environment. The weighting function further controls the impact of the current reservoir state on behavior. Intuitively, this function encodes the importance of re-filling reservoir x given its current state. In addition, a constant priority weighting p_x for each drive describes the importance of this drive compared to the others. Thus, the overall importance can be computed as:

$$\iota_x(t) = \omega(\sigma_x(t))p_x. \tag{2}$$

The equation essentially reflects the importance of drive x and thus can be used to motivate current behavior. Given all current importance values, behavior can be controlled to still the currently most-important need satisfying other needs on the way given an appropriate opportunity. Thus, self-motivated behavior can be realized.

A fundamental distinction, however, can be drawn between motivations that can be stilled by a typical consummatory behavior, such as eating or drinking, or by obeying particular constraints, such as not entering certain regions. While the proposition of this general distinction was made elsewhere [27], terming it location-based and property-based motivations, here we further detail this distinction and embed it into a wider context. We consequently also refer to the location-based motivations more generally as *consummatory* motivations.

3.2 Consummatory Motivations

The impact of each motivations depends on its update function ϕ_x. Generally and without other context information, ϕ_x may be considered as continuously decreasing reflecting the bodily consumption of energy. However, in consummatory motivations an increase in the reservoir state occurs only upon a (successful) consummatory behavior while an increase in property-based motivations occurs while the encoded property is increasingly satisfied. Also blends between the two types are certainly possible.

With respect to sensorimotor population codes, consummatory motivations are satisfied upon the successful execution of a particular behavior in a particular context, such as eating. Thus, consummatory motivations can be associated with particular behavior patterns that are executed in a particular behaviorally-relevant context. These behavior patterns may be represented by particular neurons in a population code and may be activated when the importance $\iota_x(t)$ grows to a certain level compared to the other importance values $\iota_y(t)$ of other motivations y. The neural activity can then serve as a particular goal representation, which can lead to goal-directed behavioral patterns, regardless if using the SURE_REACH encoding for arm control, the TGNG encoding for navigation control, or other related sensorimotor-based spatial encodings.

Interestingly, Graziano has shown that ethologically relevant behaviors are encoded in partially regionally distinct population codes. Thus, consummatory motivations may be associated with particular ethologically relevant behavioral patterns and may even lead to the formation of such patterns in the first place.

3.3 Property-Based Motivations

In difference to consummatory motivations, property-based motivations rather strive for the maintenance or avoidance of particular situations. For example, we do not like to keep our arm in an uncomfortable position (such as an extreme twist) for an extended amount of time. This was shown in tasks in which a particular task had to be executed leading to a particular end-state. Results show that we chose, for example, to grip a stick in anticipation of the hand and arm end-state, which is optimized for posture comfort [30].

In the navigation domain, scared animals may avoid open areas striving for protection and thus preferring to move along walls or through tunnels. In this case a safety motivation may exist that leads to the modification of path planning given the current drive for safety. In population codes, both types may be encoded by a preference bias that is spread over the full population code. That is, while a consummatory motivation may activate a certain sub-population or even single neurons in a population code, property-based motivations pre-activate or pre-inhibit full population codes property-dependent.

Thus, property-based motivations will have a different effect on behavior than consummatory motivations.

3.4 Both Types Combined

When combining both types of motivations, it does not come as a surprise that they need to be handled differently. In fact, it was shown that property-based motivations need to modify the activity propagation of consummatory motivation-based activities through a population code in order to realize goal-directed behavior while satisfying property-based motivations.

As an example, it was shown that a simulated, “scared” robot may walk along walls in order to reach food locations by modifying the food-originating activity propagation by a wall closeness preference:

$$h_i \leftarrow \max\left\{h_i^e; \max_j \left[\gamma(h_j + (s_j - 1)\iota_f)\right]\right\}, \tag{3}$$

where h_i^e denotes the consummatory motivational activity in neuron i, h_i the propagated consummatory activity, s_i the property-based motivational activity,

ι_f the current importance of the property-based motivation, and where j iterates over all neighboring neurons of neuron i (cf. [27]). Note that $(s_j - 1)\iota_f$ is always a negative value, which essentially denotes the cost of moving towards node j from node i.

In general, property motivation dependent goal-pursuance can be realized by constraining consummatory activity propagation by the current property-based motivational activities. The result is a system that acts goal-directedly striving for consummatory behaviors while considering property-based motivations concurrently. For TGNG, it was shown that the combination results in a system that exhibits latent learning capabilities, exploits behavioral opportunities, and yields emergent behavioral patterns due to the concurrent combination of different motivations respecting their current priorities [31].

Since the SURE_REACH architecture essentially encodes sensorimotor spaces similar to the TGNG approach for navigational spaces, similar motivational combinations can be used also for SURE_REACH. So far, only a simple priority-based drive has been included in SURE_REACH in order to avoid extreme arm postures [32], but more elaborate combinations are in preparation.

4 Summary and Conclusions

Summing up, we proposed that population codes can be very suitably combined with motivational drives in order to realize systems that exhibit self-motivated, goal-oriented behavior. Such systems may thus not need to be explicitly programmed to execute particular behaviors or reach particular goals, but rather only need to be informed about which internal variables need to be kept in sufficient homeostasis. Learning and adaptation of the population codes and their associations with particular motivations then lead to the pursuance of goal-directed behavior.

The changes in the internal variables can additionally distinguish between consummatory motivations and property-based motivations. Thus, the distinct propagation of both types of motivation and the influence of property-based motivations on the consummatory activity propagations can be realized without the need for pre-programming. This motivation concept can be added to both, the SURE_REACH system for flexible, end-posture oriented arm control [20] as well as to the TGNG model for robot navigation [25].

In general, any system that utilizes sensorimotor codes for effective behavior control may be combined with the homeostatic motivation concept. For example, a sensorimotor representation was recently used to optimize self-localization based on the principle of information gain [33]. The utilized information gain principle may be coupled with the proposed curiosity-based motivation. In this way, the system may become even more knowledge-gain directed, as long as the curiosity drive is stronger than the current consummatory motivations. Similarly, more schema-based approaches such as anticipatory learning classifier system architectures [34, 35] may be combined with such reservoir-based motivational systems, as originally already envisioned by John H. Holland in his first learning classifier system implementation—the “cognitive system” CS1 [36].

While the thoughts pursued herein were so-far only tested on rather static, location-based sensorimotor population codes, it becomes apparent that many behaviors are not simple transitions in space or posture, but unfold in an extended fashion over time. As mentioned above, a recent neuroscience review of Graziano [13] suggests that the motor cortex concurrently encodes ethologically-relevant behaviors besides posture-based and directional codes of particular limbs or motor synergistic combinations of limbs.

Particularly the formation of ethologically relevant behavioral codes may be controlled not directly by genetic encodings, but rather by motivational encodings coupled with an appropriate bodily morphology and basic reflexes. Thus, motivations may serve as a fundamental brain structuring principle that may lead to the formation of those increasingly abstract sensorimotor representations that are relevant for the organism. By the encoding of relevancy (via motivations) rather than the full behavior, much more flexibility may be maintained in the evolution of the respective genetic encoding and in learning to satisfy the respective motivational drives during development. Future research will show to what extend this proposition holds for highly complex, adaptive, social, cognitive systems, such as us humans.

References

1. Wolff, P.: Wahrnehmungslernen durch Blickbewegungen. In Neumann, O., ed.: Perspektiven der Kognitionspsychologie. Springer-Verlag (1985) 63–111
2. Wolpert, D.M., Goodbody, S.J., Husain, M.: Maintaining internal representations: the role of the human superior parietal lobe. Nature Neuroscience **1** (1998) 529 – 533
3. Abeele, S., Bock, O.: Transfer of sensorimotor adaptation between different movement categories. Experimental Brain Research **148**(1) (2003) 128–132
4. Thut, G., Cook, N.D., Regard, M., Leenders, K.L., Halsband, U., Landis, T.: Intermanual transfer of proximal and distal motor engrams in humans. Experimental Brain Research **108** (1996) 321–327
5. Hommel, B., Müsseler, J., Aschersleben, G., Prinz, W.: The theory of event coding (TEC): A framework for perception and action planning. Behavioral and Brain Sciences **24** (2001) 849–878
6. Hoffmann, J.: Vorhersage und Erkenntnis: Die Funktion von Antizipationen in der menschlichen Verhaltenssteuerung und Wahrnehmung. [Anticipation and cognition: The function of anticipations in human behavioral control and perception.]. Hogrefe, Göttingen, Germany (1993)
7. Hoffmann, J.: Anticipatory behavioral control. In Butz, M.V., Sigaud, O., Gérard, P., eds.: Anticipatory Behavior in Adaptive Learning Systems: Foundations, Theories, and Systems. Springer-Verlag, Berlin Heidelberg (2003) 44–65
8. Philipona, D., O’Regan, J.K., Nadal, J.P.: Is there something out there? Inferring space from sensorimotor dependencies. Neural Computation **15** (2003) 2029–2049
9. Butz, M.V.: Sensomotorische Raumreprsentationen. Informatik-Spektrum **31** (2008) 237–240

10. Witt, J.K., Proffitt, D.R.: Action-specific influences on distance perception: a role for motor simulation. J Exp Psychol Hum Percept Perform **34**(6) (2008) 1479–1492
11. Witt, J.K., Proffitt, D.R., Epstein, W.: Tool use affects perceived distance, but only when you intend to use it. Journal of Experimental Psychology: Human Perception and Performance **31** (2005) 880888
12. Caggiano, V., Fogassi, L., Rizzolatti, G., Thier, P., Casile, A.: Mirror neurons differentially encode the peripersonal and extrapersonal space of monkeys. Science **324** (2009) 403–406
13. Graziano, M.S.A.: The organization of behavioral repertoire in motor cortex. Annual Review of Neuroscience **29** (2006) 105–134
14. Holmes, N.P., Spence, C.: The body schema and multisensory representation(s) of peripersonal space. Cognitive Processing **5** (2004) 94–105
15. Arleo, A., Gerstner, W.: Spatial cognition and neuro-mimetic navigation: A model of hippocampal place cell activity. Biological Cybernetics **83** (2000) 287–299
16. Fleischer, J.G.: Neural correlates of anticipation in cerebellum, basal ganglia, and hippocampus. In Butz, M.V., Sigaud, O., Pezzulo, G., Baldassarre, G., eds.: Anticipatory Behavior in Adaptive Learning Systems: From Brains to Individual and Social Behavior. Springer-Verlag (2007) 19–34
17. Hok, V., Lenck-Santini, P., Roux, S., Save, E., Muller, R.U., Poucet, B.: Goal-related activity in hippocampal place cells. The Journal of Neuroscience **27** (2007) 472 482
18. Butz, M.V.: How and why the brain lays the foundations for a conscious self. Constructivist Foundations **4**(1) (2008) 1–42
19. Gallese, V.: Motor abstraction: a neuroscientific account of how action goals and intentions are mapped and understood. Psychological Research **73**(4) (2009) 486 – 498
20. Butz, M.V., Herbort, O., Hoffmann, J.: Exploiting redundancy for flexible behavior: Unsupervised learning in a modular sensorimotor control architecture. Psychological Review **114** (2007) 1015–1046
21. Herbort, O., Butz, M.V.: Encoding complete body models enables task dependent optimal behavior. Proceedings of International Joint Conference on Neural Networks, Orlando, Florida, USA, August 12-17, 2007 (2007) 1424–1429
22. Herbort, O., Ognibene, D., Butz, M.V., Baldassarre, G.: Learning to select targets within targets in reaching tasks. 6th IEEE International Conference on Development and Learning **ICDL 2007** (2007) 7 – 12
23. Rosenbaum, D.A., Meulenbroek, R.G.J., Vaughan, J., Jansen, C.: Posture-based motion planning: Applications to grasping. Psychological Review **108** (2001) 709–734
24. Pedersen, G., Butz, M.V., Herbort, O.: Integrating dynamics into a human behavior model for highly flexible autonomous manipulator control. IEEE Transactions of Systems, Man and Cybernetics, Part B (submitted)
25. Butz, M.V., Reif, K., Herbort, O.: Bridging the gap: Learning sensorimotor-linked population codes for planning and motor control. International Conference on Cognitive Systems **CogSys 2008** (2008)
26. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction. MIT Press, Cambridge, MA (1998)
27. Shirinov, E., Butz, M.V.: Distinction between types of motivations: Emergent behavior with a neural, model-based reinforcement learning system. 2009 IEEE Symposium Series on Artificial Life (ALIFE 2009) Proceedings (2009) 69–76

28. Hsiao, K.Y., Roy, D.: A habit system for an interactive robot. In Castelfranchi, C., Balkenius, C., Butz, M.V., Ortony, A., eds.: From Reactive to Anticipatory Cognitive Embodied Systems: Papers from the AAAI Fall Symposium, Menlo Park, CA, AAAI Press (2005) 83–90
29. Konidaris, G., Barto, A.: An adaptive robot motivational system. From Animals to Animats **9** (2006) 346–356
30. Weigelt, M., Kunde, W., Prinz, W.: End-state comfort in bimanual object manipulation. Experimental Psychology **53** (2006) 143–148
31. Shirinov, E., Reif, K., Butz, M.V.: Selforganizing sensorimotor maps plus homeostatic motivations yield animal-like behavior. Adaptive Behavior (submitted)
32. Butz, M.V., Pedersen, G.K.M.: The scared robot: Motivations in a simulated robot arm. 32nd Annual Conference on Artificial Intelligence **KI 2009** (2009) 460467
33. Zetzsche, C., Wolter, J., Schill, K.: Sensorimotor representation and knowledge-based reasoning for spatial exploration and localisation. Cognitive Processing **9** (2008) 283–297
34. Butz, M.V., Goldberg, D.E.: Generalized state values in an anticipatory learning classifier system. In Butz, M.V., Sigaud, O., Gérard, P., eds.: Anticipatory Behavior in Adaptive Learning Systems: Foundations, Theories, and Systems. Springer-Verlag, Berlin Heidelberg (2003) 282–301
35. Gérard, P., Sigaud, O.: YACS: Combining dynamic programming with generalization in classifier systems. In Lanzi, P.L., Stolzmann, W., Wilson, S.W., eds.: Advances in learning classifier systems: Third international workshop, IWLCS 2000 (LNAI 1996). Springer-Verlag, Berlin Heidelberg (2001) 52–69
36. Holland, J.H., Reitman, J.S.: Cognitive systems based on adaptive algorithms. In Waterman, D.A., Hayes-Roth, F., eds.: Pattern directed inference systems. Academic Press, New York (1978) 313–329

Conflict Resolution while Coping with Critical Situations

Régis Newo[1], Klaus-Dieter Althoff[1], and Werner Greve[2]

[1] University of Hildesheim, Institute of Computer Sciences,
Laboratory of Intelligent Information Systems
Email: newo|althoff@iis.uni-hildesheim.de
[2] University of Hildesheim, Institute of Psychology
Email: wgreve@uni-hildesheim.de

Abstract. When facing critical situations, for example the loss of a job or the breakup of a partner, humans reactions are not predictable. In this paper we present apart of our conceptual design of a system, which simulates human's behavior in critical situations. We concentrate in this paper on conflict management strategies for our simulation Our system is based on the multi-agent systems technology and use planing algorithms. We show how we intend to resolve conflicts while coping with critical situations and present the current state of our work.

1 Introduction

How does someone react when he faces a critical situation in his life? In our everyday life, we consistently face situations which pose more or less immense challenges. Examples can be the breakup with a partner, the loss of a job, an illness or even the death of a relative. As different as those challenges can be, the reactions of the persons who are facing the same kind of challenges can be very different as well. The problem consists in finding out how someone reacts when he/she faces up a given challenge. The problem being a psychological one, there have been many research groups in psychology working in that direction, beginning in the early 1980s. They developed psychological models and paradigms in order to represent and analyze people's behaviors.

In this paper, we present our approach for the simulation of human's behavior in critical situations. From a psychological point of view, our approach is based on the theory on coping strategies developed by Brandtstädter and Greve [6]. On the other hand, from a computer sciences point of view, our approach relies on the use of multi-agent systems and Case-Based Reasoning (CBR) [1] as the main knowledge representation inference. CBR is based uses past experiences to solve new problems. We use it because human's way to act is mostly based on past experiences (first or second hand).

Past approaches of modeling psychic processes have remained on a macroscopical level, as it were, simply connecting functional devices such as "central executive", or "motivational center", etc. Switching to an agent-based approach, it will become possible to include dynamical interactions within such functional clusters (e.g., interactions between various goals a person holds or between heterogeneous emotional states - "mixed emotions" - within a person). This will certainly come much closer to what we actually are than roughly computer-metaphor inspired simulations. On the other hand, agent-based simulational models within the psychological realm have exclusively focused on interactions between persons (e.g., attitude change; [15]). Hence, the combination of (a) simulational approaches of intrapsychic processes (such as coping responses) and (b) agent-based technologies seems to be a highly promising constellation to further advance both the applicational options of simuational models and the theoretical integrity and clarity of psychological theories in the coping realm.

For this purpose we developed the SIMOCOSTS (SImulation MOdel for COping STrategy Selection) model. In the SIMOCOSTS project we are actually aiming at a threefold goal, namely (1) developing a research software tool for supporting psychologists, who are working on cognitive modeling and learning as roughly described above, in their research work, (2) realizing what we call "collaborative multi-expert-systems" (CoMES; [2]), and (3) instantiating the SEASALT software architecture [4] we developed in our research lab as a first step towards realizing CoMES.

Our approach for the simulation in this paper is based on the fact that each person is goal-driven. That means that the actions made by the person are intended to be a part of the achievement of a certain goal. The simulated person has many goals and each goal wants to be fulfilled independently of each other and plans have to be computed (for each goal) in order achieve the goals. A critical situation is thus a situation in which a goal can not be fulfilled. These goals all interact in a sort of market place. We use the terminology "market place" because we want to accentuate the competitiveness of the goals. This might lead to conflicts. Conflicts occur when different plans (which were computed for different goals) contain actions which are contradictory. We thus also have elaborate a conflict management methodology for our approach. This is the main focus of this paper.

We will first present in the next section some psychological background. After presenting some related work, we will explain how we are actually implementing our system by elaborating on the underlying concepts and the used algorithm for conflict resolution. We will present the current state and an outlook on the implementation of our system.

2 Psychological Background

Psychological coping research, during the past three decades, has largely rested on correlational questionnaire studies. Unfortunately, the causal connections between the various factors included in the available theoretical models can hardly be tested with these data. On the other hand, valid experimental studies can hardly be done in this highly sensible area, both for ethical and practical reasons. As a consequence, theoretical models have remain underdeveloped and seldom directly tested. Notwithstanding a bulk of empirical studies in this field, we still do not know the interplay of different facets and layers of the "psychic system" in its response to a threatening or burdensome experience or constellation. From a theoretical point of view, however, this interplay of intrapsychic factors is crucial for our understand of coping processes and, thus, for successful intervention. Moreover, the possible intersections to developmental theories (i.e., processing developmental challenges and tasks) is another underinvestigated issue.

At this juncture, simulational methods offer a highly useful way to sharpen theoretical assumptions (claims) and to test theoretical hypotheses on possible interactional processes of several psychic subsystems. In order to create a formalized model, an empirically corroborated theory is needed in the first place. In our work, we start from the two-process model of developmental regulation [5, 7, 10]. The starting point here is the consideration that stressful events, threats to identity, and developmental losses can be understood as problem situations with an underlying discrepancy between an is and a should be perspective of personal development, that is a regulatory deficit. However, in this approach, the differentiation between fundamental reaction modes is drawn along the boundaries of personal (i.e., behavior which is consciously and intentionally planned and governed by the person as the acting unit) and subpersonal (i.e., intraindividual processes such as information processing or emotional regulation which cannot be controlled or even initiated, often not even consciously be recognized by the person) perspectives against the background of an action-theoretical perspective of human development [11]. The model basically differentiates between two modes of coping with problems, designated as *assimilative* and *accommodative* processes [7]; these can be supported by a third mode of dealing with threats: *defensive* processes [6].

Assimilative Strategies: Intentional Self-Development. In the assimilative reaction mode, individuals try to change their life situation or their own behavior in the sense of a better alignment between their normative expectations and goals in relation to themselves [5]. For example, we can do sports to improve diminishing physical condition, or change our eating behavior to make our figure closer to our ideal in this respect. Characteristic for this mode is that personal standards and goals underlying the situational or developmental appraisal are maintained. Coping attempts in this mode are usually carried out intentionally, consciously, and controlled, and can thus appropriately be called coping strategies.

Accommodative Processes: Development as Adaptation. The attempt to remove or prevent developmental losses by means of active problem-solving can fail or be bound to difficulties and costs that are too high. Often in life fundamental revisions in life- and developmental blueprints become necessary beyond simply compensatory measures. Serious threats occur that cannot be actively removed and need to be resolved through reactive preference readjustments. In response to these burdens, the alternative option consists of revising standards and goals to the given action possibilities: This is the accommodative mode. Typical examples of accommodative reactions are the relinquishing and devaluation of blocked goals, processes of regulating standards, but also processes that lead to a more readily acceptable reinterpretation of the given situation. According to Brandtstädter's view, neither of the modes has primacy. For a given situation it is not only open which of the modes is "appropriate" or even "successful"; and it is also an empirical question with which modus the person will initially react in a stress situation; from a dynamic perspective it might often even be that just the combination of both forms is effective.

Defensively Dealing with Problems: Escape or Detour? From a coping point of view, however, it seems to make sense to add a third reaction mode to the developmental model that several of the above-mentioned models included: Individuals can apparently also completely ignore a problem, denying its meaning or even its existence. In this case they change neither the problem nor themselves: Neither personal goals, preferences, standards, nor aspects of the self-image get adjusted nor does the problem get solved actively. This defensive mode operates, as it were, entirely behind the back, as the mechanisms as well as the effects of these processes principally remain hidden from the individual.

3 Related Work

As already discussed in [17] and [16], there already exists many agent-based simulation approaches (like EOS [13] and Sugarscape [8]) that deal with human behavior. However we can not use them for our simulation because they do not deal with coping. Furthermore, cognitive architecture have also been developed (like ACT-R [3] and EPIC [12]). Yet they do not deal with critical situations. In [17] and [16], we showed a process based architecture for SIMOCOSTS. Our work is based on it. In this section we will introduce some techniques used in our simulation.

As said earlier, the simulated person has several goals and each of them needs to compute a plan on how they will be achieved. The type of planning we will use for our simulation is derived from logic-based planning (see [14]). The reason is that the simulated person is mainly represented with predicates, which give us the current state (physical and psychological) of the person. In the area of (agent) planning, some work has also already be done. We can for example see in [18] that there exists several kinds of planning algorithms. The first type of planning algorithms are the so-called linear algorithms. The particularity of linear algorithm is that the generated plan consists of actions which are chosen by only considering the preconditions and postconditions of the actions. Essentially 2 types of linear planning algorithms exists

- Progression (also called set based planning): the actions are chosen while transforming the initial state into the goal state. STRIPS [9] is a prominent example.
- Regression: the actions are chosen while transforming the goal state into the initial state.

Other methods include plan based planning and graph based planning (see [18]). The main difference between linear algorithms and the other planning methods is that dependencies between the actions are not considered in linear algorithms.

As for agent planning while dealing with conflicts, some work was already done the area also. One good example is done by Timm in [19]. Timm actually differentiate between 2 types of conflicts management methods (i.e. internal and external). The internal one considers that an agent might have several goals and trying to accomplish them might create conflicts. He developed a method (called cobac) in order to solve those conflicts. Yet we can not use it, because the agents in our simulation just have one goal they want achieve. Each simulated person has many goals and each goal is implemented as an agent. We thus do not have internal conflicts in our simulation.
His external method deals with the conflicts that might occur between several agents. His proposed algorithm (called oac) mainly uses communication between the agents in order to resolve the conflicts. That means, the agents automatically try to solve the conflicts by communicating with each other and proposing solutions. Because of the fact that we consider that the simulated person acts in a critical situation, it is not realistic to suppose that the person would take his time and elaborate a perfect solution to his problems or conflicts. The person would rather try to find quick solutions for his most important goals. The second method of Timm is thus not appropriate for our simulation either. We will explain in the next how we intend to deal with conflicts in our simulation.

4 Simulation

In this section we will explain how we intend to implement the simulation with a focus on conflict management.

In Figure 1, we can see the course of the simulation. At the beginning we have an initial (generated) situation. When facing a new situation, each agent tries to find out if the situation is critical for him. This will be done by comparing the goal state of the agent with the current state (inclusive the new situation) of the person.

Let us take a student, called Mr. X, as our example. Mr. X has three goals

1. have a stable financial situation

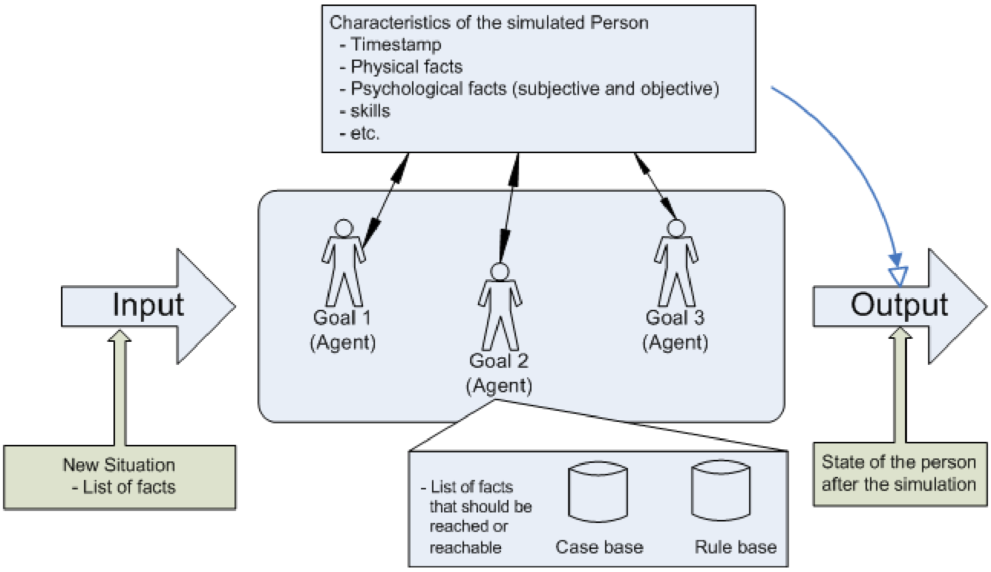


Fig. 1. Illustration of the simulation

2. finish his master's degree in Business Informatics
3. create a family

We can further suppose that none of the listed goals has already been reached. Nevertheless, plans have have already been computed in order to achieve these goals. That means that the goals are reachable and the actual situation is therefore not critical. Knowing for example that three out of four basic courses in computer sciences (let say CS1 till CS4) are needed (among other courses) in order to finish his master's degree, a valid plan for that goal might contain the actions which prompt Mr.X to take the courses CS1, CS2 and CS3.
A new situation with the information "CS3 failed" would be a critical situation for Mr. X because the second goal is not reachable anymore (with the actual plan).

We saw in earlier that the market place, in which the goals interact, is the most important part of the simulated person (and thus of the simulation). We will now explain how the intend to implement it.
The market place contains many competitive goals. As it is the case for humans, the goals are prioritized. The market place is implemented as a multiagent system with each goal being implemented by an agent. In fact these agents will be implemented following the Belief-Desire-Intention (BDI) principle (see [20]). The different properties of the person (which are the same for each agent) represent the situation. The simulated person also have many actions (e.g. take the course CS3), which can be used by any agent. In each situation, each agent tries to find out if it is critical for him. If it is the case, the agent tries to find a solution by computing a (linear) plan. The computed plan consists of several actions which

might or should affect the situation. The situation is thus updated after each action. The actions contain all facts, which will be modified in the situation after it has been applied. They also contain a time stamp which indicates the time needed before an action is completed.

A conflict occurs when the plan computed by one agent affects another agent. That is, when applying an action, the situation might change so that it becomes critical for another agent. In our example, a new plan for the second goal might include taking the course CS4 in the next term (which will be used instead of CS3). Yet if we suppose that the plan of the first goal includes having a job (in the next term) at which the person has to be at the same time on which CS4 takes place, we would have a conflict.
We developed a methodology to resolve conflicts in our simulation which is based on the prioritization of the goals. Our methodology is based on the fact that a human will first try to achieve his most important goals before achieving the others. The algorithm can be seen below.

if any conflict exists **then**
  $ConflictedAgents \leftarrow \{A_1, \ldots, A_n\}$
  while $ConflictedAgents \neq \emptyset$ **do**
    Recompute the plan for the agent A_i with the highest priority while considering the situations generated by previous plans.
    Save the all different situations from the beginning of the plan to the end.
    remove A_i from $ConflictedAgents$
  end while
end if

When applying it to the example, this simply means that we would solve the conflict by first trying to recompute the plan of the most important goal, which states that the person wants to have a stable financial situation. We will suppose that the initial plan for this goal do not change, which means that the person will keep his job. Then a new plan for the next goal in the set of conflicted agents (master's degree in business Informatics) should be computed. The agent might know from its knowledge base that the course CS3 takes place each year. That would lead to a plan stating that Mr. X should take that course again in the following year. In this case the conflict would be solved.

There are a few things that should be noticed for our conflict management methodology. First, recomputing plans does not always lead to valid one. In this case, we will have to reconsider the intentions (i.e. the goals). That means, either the goal itself or its priority will be changed (leading to an accommodative process).
Second, for a better conflict management, the new plans of each agent should be computed while taking the modifications of the situation by higher prioritized agents into account.
The output of our simulation is the state of person (i.e. the situation) after the

execution of all plans and also the plans as justifications. An important point to our simulation is that all computed plans are not executed by the person. They rather represent what the person "thinks", he should do. The person would in fact just execute the final solution (i.e. the output of the simulation)

5 Conclusion and Outlook

We presented in this paper how we intend to implement our approach SIMOCOSTS for the simulation human's behavior in critical situations. Our approach is based on the theory on coping strategies developed by Brandtstädter and Greve. Our simulation is based on the multi-agent system technology. We explained the underlying algorithm and showed by means of an example how it works. In this paper we focused on the conflict management of our agent-based system. The methodology used for this purpose derivate from human's behavior in critical situations.

Currently, we are still implementing the system. We can already represent the psychological and physical state of a person, which is an important step for the representation of the situations. We are now implementing the course of of the presented algorithm. Nevertheless a first prototype will be available in a few months. The system, once implemented, will be handed to psychologists who will conduct experiments with it and will also fill it with the required knowledge for the experiments. We will just have a few examples at our disposal which will be used for testing while implementing the systems.
We also, with this system, fulfill our aim of extending our CoMES [2] environment with another distributed knowledge-based system. Another goal is to provide a more generic architecture for the simulation, such that it can be applied in other domains like economy. It would then be possible to simulate different scenarios in a stock market for example.

References

1. Agnar Aamodt and Enric Plaza. Case-based reasoning : Foundational issues, methodological variations, and system approaches. *AI Communications*, 1(7), March 1994.
2. Klaus-Dieter Althoff, Kerstin Bach, Jan-Oliver Deutsch, Alexandre Hanft, Jens Mänz, Thomas Müller, Régis Newo, Meike Reichle, Martin Schaaf, and Karl-Heinz Weis. Collaborative Multi-Expert-Systems – Realizing Knowlegde-Product-Lines with Case Factories and Distributed Learning Systems. In J. Baumeister and D. Seipel, editors, *Proc. of the 3rd Workshop on Knowledge Engineering and Software Engineering (KESE 2007), Osnabrück, Germany*, Berlin, Heidelberg, Paris, 2007. Springer Verlag.
3. J. R. Anderson, M. D. Byrne, S. Douglas, C. Lebiere, and Y. Qin. An integrated theory of the mind. *Psychological Review*, 111(4):1036–1050, 2004.

4. Kerstin Bach, Meike Reichle, and Klaus-Dieter Althoff. A Domain Independant System Architecture for Sharing Experience. In Alexander Hinneburg, editor, *Proceedings of the LWA 2007: Lernen - Wissen - Adaptivität*, pages 296–303, September 2007.
5. J. Brandtstädter. Action perspectives on human development. In R.M. Lerner, editor, *Theoretical models of human development (Handbook of child psychology)*, volume 1, pages 516–568. Wiley, New York, 6th edition, 2006.
6. J. Brandtstädter and W. Greve. The aging self: Stabilizing and protective processes. *Developmental Review*, 14:52–80, 1994.
7. J. Brandtstädter and K. Rothermund. The life course dynamics of goal pursuit and goal adjustment: A two-process framework. *Developmental Review*, 22:117–150, 2002.
8. J.M. Epstein and R. Axtell. *Growing Artificial Societies: Social Science from the Bottom Up.* Brookings Institution Press, Washington D.C., 1996.
9. R. E. Fikes and N. J. Nilsson. STRIPS: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.
10. W. Greve. Maintanining personality: The active and adaptive self as core of individuality and personhood. In W. Greve, K. Rothermund, and D. Wentura, editors, *The adaptive self. Personal continuity and intentional self-development*, pages 49–70. Hogrefe & Huber, Cambridge, MA, 2005.
11. W. Greve and D. Wentura. Personal and Subpersonal Regulation of Human Development: Beyond Complementary Categories. *Human Development*, 50:201–207, 2007.
12. David E. Kieras. EPIC Architecture Principles of Operation, 2004.
13. Franziska Klügl. *Aktivitätsbasierte Verhaltensmodellierung und ihre Unterstützung bei Multiagentensimulationen.* PhD thesis, Universität Würzburg, 2000.
14. Steven M. LaValle. *Planning Algorithms.* Cambridge University Press, 2006.
15. H.-J. Mosler. *Computersimulation sozialpsychologischer Theorien.* Weinheim: Beltz, 2000.
16. Thomas Müller. Simulation kognitiver Prozesse mit Multiagentensystemen. Master's thesis, Universität Hildesheim, 2006.
17. Régis Newo, Thomas Müller, Klaus-Dieter Althoff, and Werner Greve. Learning to Cope with Critical Situations - A Simulation Model of Cognitive Processes using Multiagent Systems. In Alexander Hinneburg, editor, *Proceedings of the LWA 2007: Lernen - Wissen - Adaptivität*, pages 159–164, September 2007.
18. Josef Schneeberger. Planen. In Oldenburg, editor, *Handbuch der Künstlichen Intgelligenz*, chapter 13, pages 491–515. G. Görz and C.-R. Rollinger and J. Schneeberger, 4 edition, 2003.
19. Ingo J. Timm. *Dynamisches Konfliktmanagement als Verhaltenssteuerung Intelligenter Agenten.* PhD thesis, Universität Bremen - Technologie-Zentrum Informatik, 2004.
20. Michael Wooldridge. *An Introduction to MultiAgent Systems.* John Wiley & Sons, June 2002.

CASIMIR – A Computational Architecture for Modeling Human Spatial Information Processing

Thomas Barkowsky

SFB/TR 8 Spatial Cognition, Enrique-Schmidt-Str. 5
28359 Bremen, Germany
barkowsky@sfbtr8.uni-bremen.de

Abstract

The computational cognitive architecture CASIMIR aims at modeling a wide range of phenomena, representations, and processes involved in human spatial reasoning and problem solving. While performing on a spatial problem, pieces of information stored in semantically organized long-term memory structures are retrieved and aggregated for being used to build up spatio-analogical working memory representations.

These working memory representations are specifically adapted to the task being performed. That is, from a structural point of view, they are highly economic in that they only require storage capacity and processing power in the order required by the amount and complexity of the spatial knowledge involved. Depending on the variety and the types of spatial knowledge dealt with, working memory representations gradually may be extended in a qualitative way thus forming flexible representation structures that may range from basic forms of spatial representations (e.g. reflecting spatial ordering information) up to fully fledged mental images involving a full range of visual features.

As a future perspective, CASIMIR will be extended to interact with external pictorial representation thus providing the option to apply the cognitive architecture in interactive intelligent assistance scenarios, for instance in spatial reasoning or planning tasks.

The 32nd Annual Conference on
Artificial Intelligence (KI-2009)
September 15-18, 2009, Paderborn, Germany

Workshop on

Distributed Computing in Ambient Environments (DiComAe)

http://ki2009.uni-paderborn.de/index.php?id=54

André Brinkmann, Heinz-Josef Eikerling &
M. Zaheer Aziz (Eds.)

KI 2009

AI And Automation

Workshop Organizers and Co-Chairs

André Brinkmann	University of Paderborn, Germany Andre.Brinkmann@hni.uni-paderborn.de
Heinz-Josef Eikerling	Siemens AG, Germany Heinz-Josef.Eikerling@c-lab.de
M. Zaheer Aziz	University of Paderborn, Germany aziz@get.uni-paderborn.de

Publication Chair

Sajjad Mohsin	COMSATS Institute of Information Technology

Program Committee

Adedayo Adetoye	University of Reading, UK
Arshad Ali	National University of Science and Technology, Pakistan
Roger Leslie Anderton Cottrell	Stanford University, USA
Atta Badii	University of Reading, UK
Farokh B. Bastani	University of Texas at Dallas, USA
Koen Bertels	Delft University of Technology, Netherlands
Markus Eisenhauer	Fraunhofer Institute for applied Information
Arshad Ali	Technology FIT, Germany
Klaus Marius Hansen	University of Arhus, Denmark
Mario Hoffmann	Fraunhofer-Institute for Secure Information
Arshad Ali	Technology SIT, Germany
Animesh Pathak	INRIA Paris-Rocquencourt, France
Pablo Antoln Rafael	Telefnica I+D (TID), Spain
Omer F. Rana	SCS, Cardiff University, Wales
Daniel Thiemert	University of Reading, UK
Weishan Zhang	University of Aarhus, Denmark

Preface

André Brinkmann[1], Heinz-Josef Eikerling[2] and M. Zaheer Aziz[3]

[1] Paderborn Center for Parallel Computing
[2] Siemens AG SIS C-LAB
[3] GET Lab, Paderborn University
brinkman@uni-paderborn.de, heijo@c-lab.de, aziz@get.upb.de

Ambient computing as a discipline of distributed computing is assumed to be a key enabler for the "Internet of Things and Services". Key tasks consist of the seamless integration of physical objects and virtual objects like software services by means of advanced semantic technologies, establishing secured data transmission between the involved peers taking into account contextual information, and featuring self-management and reconfiguration capabilities. The availability of appropriate technologies and solutions will be a necessity for many future applications, e.g. in health care and consumer electronics domains.

The DiComAe (Distributed Computing in Ambient Environments) workshop was organized to provide an open forum for scientists and engineers from academia and industry to exchange and discuss their experiences, new ideas, research results, and products particularly dealing with semantics and knowledge processing for developing ambient applications. The contributions in these proceedings document the diversity of the subject.

The challenges imposed by ambient computing are examined by the Hydra Project Consortium, which is funded by the European Community IST Directorate FP6 Program. Some of the contributions capture results and experiences gained when specifying the Hydra middleware, which explicitly intends to define a configurable set of components (Hydra managers) plus according tools to more quickly and reliably assemble ambient computing applications.

André Brinkmann,
Heinz-Josef Eikerling,
M. Zaheer Aziz

Architecting Self-Management in the Hydra Middleware

Mads Ingstrup[1], Weishan Zhang[1], Klaus M. Hansen[1,2], and João Fernandes[1]

[1] Aarhus University
[2] University of Iceland
[1]{ingstrup,zhangws,jfmf}@cs.au.dk
[2]kmh@hi.is

The Hydra self-management architecture is based on Kramer and Magee's three-layer architecture (3L) for self-managing systems. However the 3L architecture is conceptual and thus at a very high level of abstraction. In this paper we describe in detail the architectural design of the self-* features in the Hydra middleware, and thus contribute significant details on how the 3L architecture can be realized in practise. We describe how each layer has been realized and our design for how the layers interact, as well as argue the particular design choices we have made in order to move from a conceptual to a real 3L architecture.

Hydra Middleware for Developing Pervasive Systems: A Case Study in the e-Health Domain

Marco Jahn, Ferry Pramudianto, Ahmad-Amr Al-Akkad

Fraunhofer Institute for Applied Information Technology FIT, Schloss Birlinghoven, 53754 Sankt Augustin, Germany
{marco.jahn, ferry.pramudianto, ahmad-amr.al-akkad}@fit.fraunhofer.de

Abstract. In this paper we present from a developer's point of view lessons learned from building an ambient e-health system using Hydra middleware. Hydra provides a middleware framework that facilitates developers to build efficiently scalable, embedded systems while offering web service interfaces for controlling any type of physical device. It comprises a set of sophisticated components taking care of security, device and service discovery using an overlay P2P communication network. We describe in detail how we applied Hydra middleware for an e-health scenario aimed at supporting the routine, medical care of patients at home. Such a scenario illustrates the complexity of a pervasive environment that Hydra aims to solve. We elaborate the process of applying Hydra for building a pervasive environment, the problems we faced, and what we learned for future research.

Keywords: Pervasive Computing, Middleware, Networked Embedded Systems, SOA

1 Introduction

The seamless integration of technical devices becoming an integral part of our everyday life reflects many visions of pervasive computing [1]. Mobile computing has already prevailed as a common technology helping people achieving their routine activities. Going one step further, pervasive environments proactively support the user by providing ambient interaction with his surroundings and at the same time being minimally intrusive. For instance, in the area of home automation such a system could help people saving energy according to the their behavior and context information. In an e-health scenario a pervasive environment could monitor a patient's condition and constantly send these data to his doctor and notify an emergency service in a critical situation. Developing such pervasive environments covers a broad range of different technological challenges that developers have to consider such as [2]:

- Devices must be able to dynamically enter and leave the environment and at the same time they need to be aware of this frequent changes.

- Devices should be able to exchange information regardless of their communication technology, such as Bluetooth, RFID, Wi-Fi, Zigbee[1] etc.
- The communication flow among devices has to be secured to guarantee privacy and protect against misuse of information.

These points show that a pervasive environment implies a high degree of flexibility. Accordingly, the software architecture has to support this need. The service oriented architecture (SOA) paradigm represents a promising approach to face these challenges. It allows the integration of distributed software systems, which is highly desirable when trying to interconnect heterogeneous devices and systems. Services in a SOA are described by a platform-independent specification that abstracts from the complexity of the underlying systems, are loosely coupled, and in particular reusable [3].

The Hydra middleware framework strives to provide a SOA that enables developers to build scalable, embedded systems without having to deal with the complexity that pervasive environments require. In particular, Hydra makes benefit of Web Services as middleware technology to support the discovery, description and access based on XML and web protocols over the Internet.

In this paper we present our experiences gained from a developer's point of view, building an ambient e-health system, which is an appropriate test bed for Hydra since such a system requires the developers to deal with various sensors and actuators, and their communication protocols. Moreover, e-health is becoming an important factor improving our quality of life.

We show what the benefits and pitfalls are, when setting up such an environment based on the Hydra middleware. Our work shows, how well a middleware framework like Hydra applies to the requirements evolving from pervasive environment scenarios.

We first explain the motivation of our work by describing the problem and presenting an envisioned scenario. To outfit the reader with the current state of research we first give an overview of the main components of the Hydra middleware and present examples of related work. Next, we describe our testing environment and the experiences gained. This paper concludes with an outlook on the upcoming work.

2 Problem

To efficiently build pervasive environments that interconnect a broad range of devices and establish communication among these devices, developers need to be equipped with tools and frameworks that simplify the development process by abstracting from the technological heterogeneity. The Hydra middleware framework with its modular service oriented architecture claims to provide a solution to these problems. Developers have to carefully elaborate the requirements of their envisioned environment and make a decision whether developing everything from scratch or utilizing a framework that promises to meet their requirements. Making such a

[1] http://www.zigbee.org/ (last visited on 15/07/09)

decision is never easy and especially when the developers are faced many choices without supporting facts about the tools or frameworks.

3 Scenario

Otto - a former professional e-sports gamer - has become old and now is a patient who is living at home and taken care of by a nursing service. Due to a heart disease, his health values have to be monitored by his family doctor so he can react to any aggravation of his patient's health. Otto sometimes is a little bit lazy, so his doctor sends him a message, that he should measure his temperature and blood glucose level and weigh himself. Otto heeds his doctor's advice but unfortunately he broke his blood glucose meter, so he just takes the temperature and weighs himself on his Wii Balance Board that he used to play a lot on, back in his better days. The resulting values are sent to his doctor's Tablet PC automatically. The doctor recognizes that the weight is ok, but the glucose level is missing and the temperature rose significantly since the last measurement. He decides it would be best, to have Otto checked through in the hospital. Therefore he sends this task to the nursing service with Otto's health information and a notification to Otto's phone that a nurse will come to pick him up. Otto agrees and generates a security token from his phone and sends it to the nursing service. The nurse receives her token and saves it to her nursing smart card while Otto saves his token on his Playstation 3 (he was a distinguished online soccer expert, but now uses it to control his front door). As the nurse arrives, she utilizes her smart card to open the door; the Playstation 3 (PS3) matches both tokens and she can enter Otto's house. Luckily the nurse brought a glucose meter that seamlessly integrates into Otto's home environment. She measures Otto's glucose level (again the result is sent to the doctor automatically) and takes Otto to the hospital. Since Otto's doctor knows best about his progress of disease he calls the hospital and discusses further treatment modalities.

4 The Hydra Middleware

To provide the reader with the basic concepts of Hydra we briefly introduce important terms and functionalities of the framework:

Hydra Devices Hydra separates between resource restricted devices that are not able to host Hydra middleware (non Hydra-enabled devices) and more powerful devices that are (Hydra-enabled devices). Restricted devices are connected to the Hydra network via a proxy mechanism. Such proxies are deployed on Hydra-enabled devices that are then called gateways. In a Hydra network, every device, whether it is Hydra-enabled or represented through a proxy, is called a Hydra Device.

Networking In Hydra the main component responsible for communication among devices is the Network Manager. It creates an overlay P2P network that all Hydra Devices are connected to. Since Hydra implements a service oriented architecture using web services, the Network Manager employs SOAP Tunneling [4] as transport mechanism for web service calls. This allows Hydra Devices to communicate through firewalls or NAT.

Device Discovery As stated before, the detection of new devices is a key requirement to a pervasive environment. In our example, the nurse's glucose meter has to seamlessly integrate into Otto's environment. To this end the device has to be a Hydra Device that can be discovered by Otto's controlling application.

Security When sending private data over a computer network, security and privacy are always major issues. When Otto sends his health values to his doctor it is essential, that these sensitive data cannot be accessed by anybody but the dedicated recipient. Hydra supports security on different abstraction levels.

Context Awareness In a pervasive environment, devices and applications ought to be aware of the user and his surroundings, being able to react to changes in context. Hydra provides dedicated components that implement extendable rule based configurations for collecting, processing and combining context information.

Device and Application Development The Hydra architecture design allows for a clear division of work regarding device- and application development. A device developer is responsible for making Hydra Devices out of any kinds of physical devices. The application developer then should be able to easily integrate such devices into an existing Hydra network or build up a new one.

5 Related Work

Several middleware for networked embedded devices exist, for example SOCRADES[2], an effort to integrate embedded devices into SOA so that the information on the device level can be included easily to the business processes [5, 6]. It provides many features such as eventing, management, service composition and discovery. Unfortunately, they did not discuss about the possibility of semantic selection which is desirable when we deal with a large number of devices in an ad-hoc and dynamic environment. Another related work is AMIGO, an EU commission project aiming at development of an open, standardized, interoperable middleware and intelligent user services for the networked home environment [7].

[2] www.socrades.eu (last visited on 15/07/09)

6 Test Bed and Lessons Learned

6.1 Architecture

Figure 1 visualizes the runtime architecture of our e-health environment. It shows that we integrated several non Hydra-enabled devices (depicted as images of the real devices) and four Hydra-enabled devices. Each non Hydra-enabled device is connected via a proxy on a Hydra-enabled device. To an application developer this is completely transparent; to him each device in this network is a Hydra Device, whether it is the Balance Board or the PS3. The complete setup is as follows (leaving out the glucose meter):

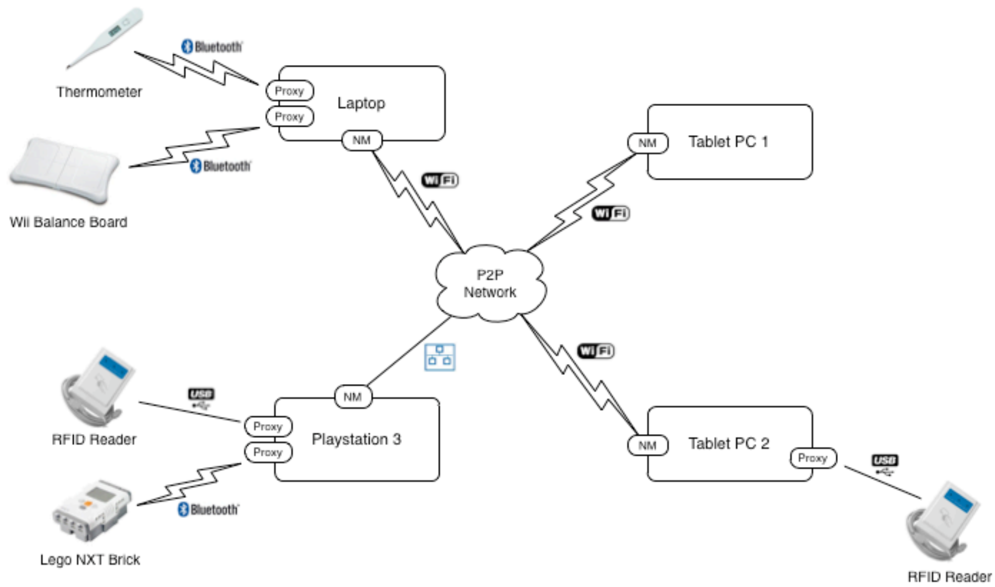


Fig. 1. Runtime architecture of the e-health environment

The laptop acts as proxy for the thermometer and the Balance Board. Both devices are connected via Bluetooth. The laptop itself is connected to the Hydra network via WiFi. The PS3 communicates with the Hydra network via LAN. Additionally it hosts proxies for the RFID reader (in the scenario it controls door access) and a Lego NXT brick that opens and closes the door of Otto's home. Tablet PC 1 (the doctor's PC) does not host any other devices, it may provide own services and invoke calls to other Hydra Devices. Tablet PC 2 is the nurse's PC that connects another RFID reader to the network. Recalling the scenario, the nurse needs to save her security token on a RFID card.
This visualization of the runtime architecture shows again that Hydra developers mainly deal with two tasks: Integrating non Hydra-enabled devices and connecting Hydra-enabled devices to a network. We try to put the Hydra middleware framework to a test by using a wide range of devices with different hardware specifications. In

the following we present our experiences gained from setting up the described environment. Following our approach of reporting from a device developer's perspective we mainly describe the process of integrating the two different types of devices: As examples of non Hydra-enabled devices we choose the Wii Balance Board and a RFID Reader. To show our experiences with a Hydra-enabled device, we report on the integration of Playstation 3.

6.2 Integrating Resource Restricted Devices

One major concept for making a non Hydra-enabled device a Hydra Device is the proxy. Such proxies offer web service interfaces for accessing the functionalities of the respective device as well as a set of basic functions that represent the behavior of a Hydra Device. Moreover, a proxy offers services running on top of UPnP protocol, allowing automatic device discovery.
For developing proxies, Hydra offers tool support suited for different platforms. One can either use Limbo [8], a Java-based web service compiler, or a set of discovery tools using .NET.
Limbo generates web service code for different target platforms (currently J2SE and J2ME) and different protocols allowing SOAP communication over TCP, UDP or Bluetooth. When using Limbo to integrate a device, developers have to specify the device's service interface and provide some Meta-information about the device. Limbo can then generate web service code that serves as basis for the device proxy. These services include Hydra standard services, device specific services, energy efficiency services and UPnP services to make the device discoverable.
When the device developer decides to build a Hydra Device using .NET, he extends the Hydra discovery tool by integrating the corresponding proxy into the discovery tool. This tool then instantiates the proxy when it discovers the respective device. The .NET library defines two abstractions for a device proxy: Device Device Manager and Device Service Manager. A Device Device Manager handles the creation of services that can be consumed from a Hydra network. A Device Service Manager is in charge of the communication between proxy and the physical device.

Integrating Devices with Limbo The Wii Balance Board itself is not capable of hosting parts of the Hydra middleware. Thus, we had to build a proxy to make it available as Hydra Device. For our scenario we decided to let Limbo generate the web service code for UPnP discovery, Hydra standard services, and device specific services. The device specific web service represents the functionality offered by the Wii Balance Board, consumes incoming calls from the Hydra network and delegates them to the logic that handles low level Bluetooth communication between proxy and Balance Board.
Before letting Limbo generate any code, we had to do two things: Specify the device service's interface and provide some basic Meta-information about the device itself. For the service interface Limbo supports WSDL, the device information has to be provided in OWL, both well-known standards. The Balance Board's interface was kept quite simple, providing only one method for accessing the weight value. Meta-information included things like manufacturer, model name, device description etc.

Outfitted with this information we took Limbo to generate the three services that are required to integrate a device: The device specific service (depicted in the WSDL), the Hydra standard service and the UPnP service. The next step was to register these services at the Network Manager running on the proxy machine, to ensure that device-to-device communication is completely under the control of Hydra. At that point the Balance Board proxy is visible and accessible inside the Hydra network. What remained to be done was implementing the low-level communication between proxy and Balance Board. We used the BlueCove[3] Library to provide Java-access to the Balance Board via Bluetooth.

Integrating Devices with .NET Just like the Balance Board, the RFID readers are non Hydra-enabled devices and have to be integrated via proxies. The .NET tools comprise a set of libraries (including the aforementioned Device Device- and Device Service Manager) providing reusable functionality for device developers. To enable UPnP-based device discovery we used the Intel UPnP Tool[4] to generate UPnP service code (C#) and then adopted that code inside the Device Device Manager. Doing so, the proxy can announce itself representing the corresponding physical device in the Hydra network. Next, we wrote a service contract for the RFID reader as well as for the Tablet PC with the methods that we would like to offer. We inherited the class library `BasicHydraDevice`, and sent our service contract to this parent class, which then generates a web service stack. In the Device Service Manager we use an external library that handles the communication with the RFID reader and our Tablet PC UPnP device.

Comparing both approaches, we feel that none of them has major advantages over the other. Both provide good support for automated generation of UPnP services, which is essential to make devices discoverable. Also implementing device specific services is well supported by both approaches.

Nevertheless, there are some unique features to each of the tools. Limbo is able to generate web service code for several platforms including J2ME. This allows developers to integrate mobile phones supporting J2ME directly, without having to build proxies. In that case all the Limbo-generated code will run on the device itself. As described in the Balance Board report, when integrating a device with Limbo, developers have to deal with the underlying low-level communication themselves, which might not be a too big drawback because established libraries for many protocols exist. Nonetheless, the Hydra .NET libraries offer a bit more comfort, providing implementations of various device protocols, e.g. Bluetooth, Zigbee, and RFID that can be used out of the box. The decision which approach to choose might mostly be influenced by the developers' programming language preference. Of course a dedicated Java developer might go for the Limbo approach while a .NET developer might probably make a different decision.

[3] http://www.bluecove.org/ (last visited on 15/07/09)

[4] http://software.intel.com/en-us/articles/intel-software-for-upnp-technology-technology - overview/ (last visited on 26/06/09)

6.3 Building the Hydra Network

Since the focus of our work lies on device development, our application logic is still on a rather basic level. Nevertheless, we have all our devices connected to the Hydra network; communication is completely realized over Hydra mechanisms. This means, that all web service calls are routed through the corresponding Network Managers that take care of SOAP Tunneling.
As depicted in Figure 1, each Hydra-enabled device hosts a Network Manager. The installation of these components on the PCs worked very well; not least because these managers are deployed in an OSGi[5] runtime supporting the need for a modular and pure service oriented environment. To test Hydra under harder conditions we took the Playstation 3 as an example of a non-standard device.

Integrating Playstation 3 As the PS 3 is powerful enough to host a Network Manager it does not need a proxy to become a Hydra Device. As a first step we set up Equinox as OSGi runtime environment, which did not cause any problems. At first, we needed adapted versions of the Network Manager and related bundles for the Power PC architecture and Java 1.5 SDK. And second, this was the problem we couldn't fix for this installation: the constant interchanging of data with Hydra network consumed too much working memory. The working memory of the PS3 is 256MB, but the actual one was just 128MB. We solved these problems by installing a simplified Yellow Dog[6] Linux dedicated for the PS3 called Helios[7]. All in all, the PS3 with its Power PC architecture and relatively low amount of memory was not so easy to integrate. The problems we faced show that there is still some work to be done, to make Hydra available on as many devices as possible.

7 Conclusions and Future Work

Building a demonstrator for a pervasive environment in the e-health domain proved to be a good use case for evaluating the Hydra middleware framework. As this paper shows we are still at the beginning of building such an environment. Thus, our focus lied on testing Hydra against the fundamental requirements of such an environment, namely device integration and basic networking capabilities. We report from a device developer's perspective and show what it takes to make various kinds of devices usable for a Hydra application developer. The obtained results show that Hydra provides a good foundation for integrating heterogeneous devices. The tested tools and software components significantly ease the tasks of a device developer for pervasive environments. In the near future we will integrate the G1 mobile phone as a mobile device that is capable of hosting the essential parts of the Hydra middleware.
In this paper we evaluate Hydra mainly from a device developer's perspective. The next step is to move the focus to application development. Once we have a set of

[5] http://www.osgi.org/Main/HomePage (last visited on 26/06/09)
[6] http://us.fixstars.com/products/ydl/ (last visited on 26/06/09)
[7] http://www.helios.de/news/news07/N_04_07.phtml (last visited on 26/06/09)

devices and are able to easily integrate new ones, we can evaluate the features that Hydra offers for application developers. One major task will of course be to integrate Hydra security components, since secure communication is one key requirement in a pervasive e-health environment.

Acknowledgements. The work presented in this paper is supported by the European research project Hydra IP (IST-2005-034891).

References

1. Satyanarayanan, M.: Pervasive Computing: Vision and Challenges. IEEE Personal Communications, August (2001) 10-17
2. Eikerling, H.-J., Gräfe, G., Röhr, F., & Schneider: Ambient Healthcare Systems - Using the Hydra Embedded Middleware for Implementing an Ambient Disease Management System. In: Luís Azevedo & Ana Rita Londral, ed., HEALTHINF, INSTICC Press, pp. 82-89 (2009)
3. Sanders, D. T., Hamilton, J. A., and MacDonald, R. A. 2008. Supporting a service-oriented-architecture. In Proceedings of the 2008 Spring Simulation Multiconference (Ottawa, Canada, April 14 - 17, 2008). Spring Simulation Multiconference. The Society for Computer Simulation International, San Diego, CA, 325-334.
4. SOAP Tunnel through a P2P Network of Physical Devices
   Francisco Milagro, Pablo Antolín, Peeter Kool, Peter Rosengren, Matts Ahlsén
5. Karnouskos, S.; Baecker, O.; de Souza, L.M.S.; Spiess, P., "Integration of SOA-ready networked embedded devices in enterprise systems via a cross-layered web service infrastructure," Emerging Technologies and Factory Automation, 2007. ETFA. IEEE Conference on , vol., no., pp.293-300, 25-28 Sept. 2007.
6. Kirkham, T.; Savio, D.; Smit, H.; Harrison, R.; Monfared, R.P.; Phaithoonbuathong, P., "SOA middleware and automation: Services, applications and architectures," Industrial Informatics, 2008. INDIN 2008. 6th IEEE International Conference on , vol., no., pp.1419-1424, 13-16 July 2008
7. Ressel, C.; Ziegler, J.; Naroska, E., "An approach towards personalized user interfaces for ambient intelligent home environments," Intelligent Environments, 2006. IE 06. 2nd IET International Conference on , vol.1, no., pp.247-255, 5-6 July 2006.
8. Klaus M. Hansen, Weishan Zhang, and Goncalo Soares. Limbo: an ontology based web service compiler for networked embedded devices. In Proceedings of the International Conference on Software Reuse, 2008.

Hydra Semantic Security Resolution Framework to support Ambient Intelligent Environments

Atta Badii[1], Adedayo O. Adetoye[1], Daniel Thiemert[1], Mario Hoffmann[2]

[1] IMSS, University of Reading, UK
[2] Fraunhofer SIT, Darmstadt, Germany
[1]{atta.badii, a.o.adetoye, d.thiemert}@reading.ac.uk
[2]mario.hoffmann@sit.fraunhofer.de

The ambient lifestyle is characterised by users' interactions with one another as well as with service provisioning platforms in their immediate environment no matter what kind of mobile and wireless sensor, service and network technologies are behind it. The ambient user explicitly expects intelligently personalised and seamless cooperation with those technologies depending on his or her current situation and context. Furthermore, this user implicitly requires transparent, but inherently secure communications and trustworthy transactions. This paper introduces a framework architecture which supports interoperable security that meets the needs of the mobile user and the service providers in the ambient intelligent environment via semantic security resolution. In this environment intelligent resolutions of security profiles and mechanisms are made for the user with high assurances of security guarantees which meets the user's needs as specified by their security, privacy and trust policies. The framework architecture for security presented in this paper is the security implementation cornerstone of the EU project Hydra, a middleware platform for networked embedded heterogeneous physical devices, which provides high-level security, trust, and privacy guarantees via end-to-end semantic security resolution and enforcement techniques. In this paper, we present the details of the framework architecture for semantic security resolution in ambient environments. The reader is referred to the concrete implementation of this framework architecture in the Hydra middleware development platform, providing a practical platform for developers to realise secure ambient scenarios.

Security Mechanisms for an Ambient Environment Middleware

Tobias Wahl[1] , Julian Schütte[1], and Peter Kostelnik[2]

[1] Fraunhofer Institute SIT, Germany
[2] Technical University Ko2ice, Slovak Republic
[1]tobias.wahl@sit.fraunhofer.de, [1]julian.schuette@sit.fraunhofer.de
[2]peter.kostelnik@tuke.sk

Ambient environments foster a lifestyle of permanent, yet unconscious interaction with unobtrusive technology surrounding the user. One of the main challenges in this field is to avoid ambient environmentsturning into a tool for perfect surveillance and control. The EU funded project Hydra aims at developing a middleware for ambient environments bringing forward mechanisms for security, privacy and trust. This paper introduces the security approaches, following the principles virtualization, loosely coupled services and semantics.

Agent Toolkits for Ad Hoc Grids

Tariq Abdullah and Koen Bertels

Computer Engineering Laboratory, EEMCS,
Delft University of Technology, The Netherlands
{m.t.abdullah, k.l.m.bertels}@tudelft.nl

Autonomic computing systems can manage themselves by self-configuration, self-healing, self-optimization, and self-protection. Agents are the promising candidates for making the autonomic systems a reality due to their characteristics. Different research projects used software agents for developing autonomic computing applications. Agent development toolkits address different aspects of software agents. In this paper, we present a categorization of agent development toolkits from different aspects of autonomic computing paradigm. Secondly, some recommendations for selecting the appropriate toolkit while developing an autonomic ad hoc grids will be given. Finally, results of micro economic based resource discovery in a local ad hoc grid are presented.

A Distributed Computing Architecture for Artificial Visual Attention on Mobile Robots

M. Zaheer Aziz, Dirk Fischer, and Bärbel Mertsching

No Institute Given

A Distributed Computing Architecture for Artificial Visual Attention on Mobile Robots

M. Zaheer Aziz, Dirk Fischer, and Bärbel Mertsching

GET LAB, Universität Paderborn, 33098 Paderborn, Germany,
<last name>@upb.de,
WWW home page: http://getwww.uni-paderborn.de

Abstract. Mobile robot systems require real time output from many computationally expensive procedures. On the other hand they have a restricted capacity of carrying computing resources. This paper proposes a tightly coupled framework for distributed processing of the visual attention mechanism that selects salient locations in a scene as per role model of natural vision. The proposed architecture is conceptually evaluated for its feasibility and a critical analysis is done for selection of hardware to enable execution of the proposed model on mobile platforms.

1 Introduction

Mobile vision systems such as rescue robots have a major challenge of performing many complex operations in real time. The tasks to be performed are not only related to image processing and target searching but area exploration, path planning, collision avoidance, and self localization as well. In context of machine-vision, processing of data from visual and other perception related sensors is managed using a phenomenon called visual attention in our research in order to restrict time consuming operations only to some selected locations. Visual attention is a mechanism of selecting relevant or important areas of an available scene found in vision systems of primates. The selection process is done using some quickly computed features such as color contrast [1], motion[2], size [3], and orientation [4] etc. Under bottom-up attention the selected locations emerge based upon saliency due to contrast between the objects due to different feature values. On the other hand during top-down attention, e. g. searching for a target, the focus of attention (FOA) is selected depending upon similarity of features found in the scene with those of the search target.

Computation of the feature channels involved in visual attention are computationally expensive especially the processes of depth from stereo and motion analysis. Running these tasks on single processor architecture cannot yield results in real time with the current state of technology. Therefore there is need for distributing these procedures into multiple processing units to facilitate achieving higher overall efficiency. A special issue in case of mobile robotics is the limitation on the amount of computational resources to reduce pay load and energy consumption. The overhead of process and data distribution has to be also minimized in order to obtain maximum benefit of distributed computing and achieve results in real time.

Keeping in view the said requirements, this paper proposes a tightly coupled distributed architecture for the vision subsystem of a mobile robot as a prospective solution to the problem. In this discussion we concentrate on the visual attention module that selects locations in space for focusing the robotic camera head during a search operation, e.g. finding survivors in a rescue mission, or exploring an environment for collecting data according to human interest. In perspective of a complete system, this is a work in progress in which some components are complete while others are being developed. The architectural design and implementation proposal for applying distributed computing in mobile robotics scenario is the major contribution of this paper.

2 Related Literature

Design of a distributed processing architecture for visual attention systems has not been done before. Although natural visual attention works with massively parallel processes in the brain but the contemporary computational models such, as [5], [6], and [7], suggest solutions with iterative pixel processing. Our previous work on visual attention modeling (see for example [8], [9], [10], [11]) has also taken single processor infrastructure into account. In this paper we extend our system towards parallel processing architecture.

Work on distributed image processing may be considered as relevant literature to the proposed system. We mention some examples here to provide an overview of the state-of-the-art. An infrastructure for programming and executing image processing algorithms on parallel processing units is described in [12]. They report a significant speedup with usage of up to four CPUs. The work in [13] proposes an object classification and search system distributed over four servers. They evaluate the acceleration gained after distributed processing under different load balancing methods. In [14] results of experiments on large image data have been demonstrated with edge detection task. Different partitioning and scheduling strategies are investigated there.

Among the recent related literature is [15] that enables secure remote data access using a grid infrastructure without copying data from its original location. They evaluate the effect of grid overhead on distribution of image data on the network. The evaluation approach given in [16] has similarity with the one used in this paper as they divide their time line into different phases of processing while observing the processor activities. The work presented in [17] proposes a job redistribution approach according to node availability using image processing applications as case study.

3 Proposed Architecture

According the classical Flynn's taxonomy [18] the proposed system can be counted as a multiple instructions single data (MISD) mechanism. On the other hand in architectural classification point of view [19], the proposed model is tightly coupled distributed system. Figure 1 demonstrates the overall infrastructure of the proposed distributed attention system. The system takes input from a stereo camera head that is able to rotate in two degrees of freedom (pan and tilt). The hardware control server (HCS) is responsible for initializations of electromechanical components, acquiring data from sensors, and executing movement commands such as rotation of camera head.

For the purpose of a reasonable load balancing we propose to distribute single-frame image processing, stereo processing, and motion analysis into three parallel processors. The fourth processor selects the output from these three and produces the final output. As our area of application requires at least three fairly computationally heavy tasks running simultaneously therefore we opt for task-wise distribution rather than distributing portions of the input data as usually done in existing distributed image processing methods.

The image processing server (IPS) performs the procedures that require only one frame of camera input whereas the stereo processing server (SPS) is responsible to process the parallely acquired two frames from the stereo camera pair. IPS uses either the left or the right camera frame as input and extracts uniformly colored regions along with the information about neighbors of each segmented region. Saliency of a region is computed depending upon its contrast with its neighborhood in term of color, orientation, eccentricity, symmetry, and size. Details of internal processes of this module can be seen in [20].

The SPS finds corresponding pixels between the two stereo frames and constructs a disparity map. Then this map is segmented for regions possessing similar amount of disparities. Later these regions are analyzed to mark those areas that have high contrasting values. For a robot, an object that has reached too near to the system requires immediate attention, hence high depth-based saliency is given to low depth. A working prototype of this module was presented in [21]. The motion processing server (MPS) waits until two frames are captured from a single camera (left or right). After comparing the two frames it builds motion vectors and constructs regions that demonstrate similar type and speed of motion. Then these regions are analyzed to assign high saliency to those regions that behave different from others. Current status of this module can be seen in [22].

The saliency lists constructed by the IPS, SPS, and MPS are combined into a master saliency map by the attention processing server (APS). As the above mentioned three processes are of different time span they produce results at different instances. The APS activates at arrival of output from IPS and starts integrating the results of SPS as soon as they are generated. During the time when APS waits for saliency values provided by the MPS it applies the multiplicative factor of spatial inhibition of return (IOR) on the already attended areas. Later, the regions arriving from MPS also get affected by the IOR when they get merged into the master map. Then a region with highest amount of resultant saliency is selected for focusing the attention and the angles to bring this FOA into the center of view frame are conveyed to the HCS.

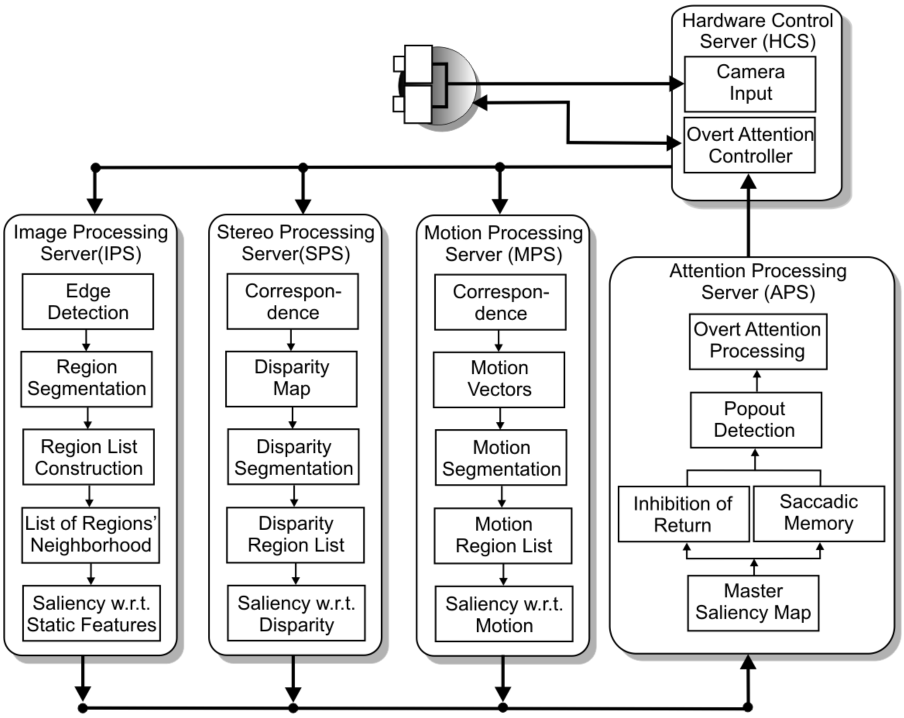


Fig. 1. Proposed architecture for distributed processing of artificial visual attention

4 Parallel Processing Resources

In the proposed distributed architecture the IPS, SPS, and MPS are extensively heavy processes for computation whereas the other two are comparatively light weight. In order to have a tightly coupled distributed system for executing the algorithms parallely we prefer to use multiple processors on a single board for minimizing pay load and energy consumption on the mobile robot. At the same time we would like to remain in a cost effective solution in order to keep mass production of the proposed system (such as a standard rescue robot) economically feasible. The choices suiting the said requirements from the available four-processor systems include AMD Phenom, Intel Core 2 Quad, Intel-Xeon, and Intel Core i7.

The five modules of the proposed system will be executed as individual threads because they share the same address space and can quickly exchange data between them. These threads will naturally share the CPUs with other processes. If the processes keep on jumping between CPUs then the *Cache Trashing* effect will start arising. To avoid time wastage in such issues we have to assign a thread permanently to a fixed CPU. Therefore our choice of operating systems converges to Linux (above version 2.5) as it allows *Hard CPU Affinity*, i. e., binding of a process to a particular CPU according to user's choice. We still have chances of cache trashing if other system processes jump around the available CPUs. This problem can be solved by assigning computationally light modules of APS and HCS along with all system processes to a single CPU while the other heavy ones to the other three. The system processes can be assigned to a processor by assigning the `init` process (process ID = 1) to the required unit. As all child processes remain on the same CPU therefore the operating system processes will remain on the assigned processor.

5 Current Status and Results

The system under discussion is a large scale project in which some components are fully functional while other are under development. Figures 2 (b) to (f) demonstrate the output of the IPS module in which bottom-up saliency maps with respect to color contrast, eccentricity, orientation, symmetry, and size are computed using a sample image containing obvious contrasts due to these features shown in figures 2(a). Results of APS module using the output of IPS for the first five fixations of attention are shown in figures 2(g). Work on SPS and MPS is in progress. Figure 3 (c) shows the motion vectors computed by the current status of MPS for the moving person and vehicles in a sequence for which two frames are given in figures 3 (a) and (b). The depth map produced by the SPS using a sample stereo pair, shown in figures 3 (d) and (e), is given in figure 3 (f). Algorithms to detect saliency with respect to depth and motion are under research.

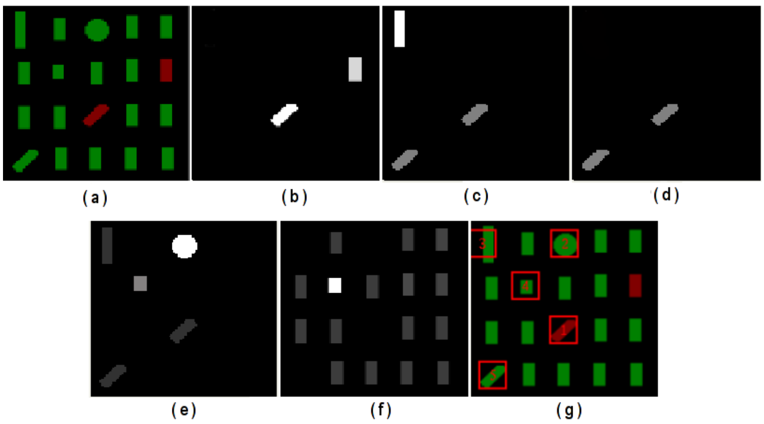


Fig. 2. Results of the proposed system on monocular static images: (a) A sample image with obvious contrast with respect to different features. (b) Regions with high bottom-up saliency with respect to color contrast computed by IPS. (c) Regions with saliency due to eccentricity feature. (d) Orientation wise salient regions. (e) Saliency with respect to symmetry. (f) Size wise saliency. (g) First five foci of attention performed by the APS.

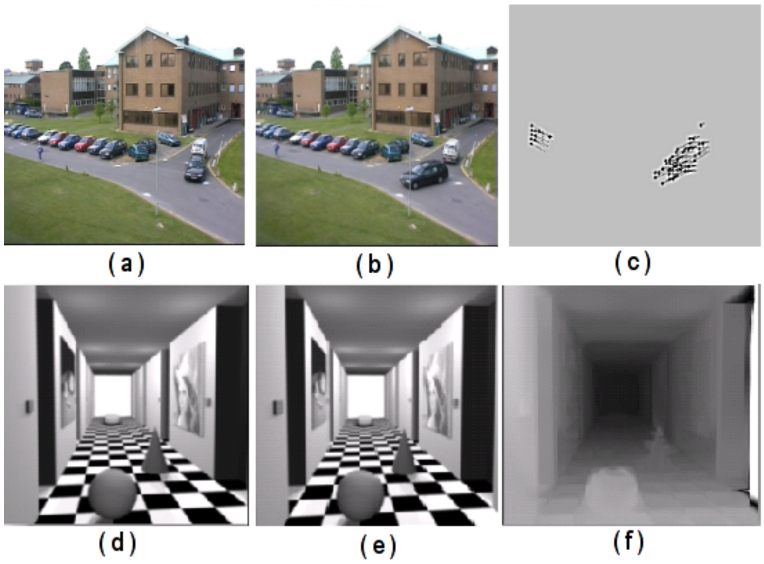


Fig. 3. Results of current status of MPS and SPS: (a) and (b) Samples from sequence frames containing motion of a person and two vehicles. (c) Motion vectors computed by the MPS. (d) and (e) Right and left frames of a stereo image pair. (f) Depth map computed by SPS with near objects shaded with brighter intensity.

6 Evaluation of Distributed System

In this section we provide a conceptual analysis of the anticipated time gain from the implementation of the proposed system as a distributed architecture. A cycle of the system's actions will usually start with movement of the sensor (camera) head to fixate on, or search for, some object. Let the time required for this sensor movement be T_{sm}. The image acquisition time may be represented as T_{ia} and the distribution and communication time may be called T_{dc}. Image acquisition from the stereo cameras is done using parallel hardware hence after the first occurrence of T_{ia} and T_{dc} the input for IPS is available. The SPS may wait for another period of T_{dc} as the second image needs to be transmitted. For motion processing, the MPS needs to wait for another period of $T_{ia} + T_{dc}$ as it needs two consecutive frames to analyze movements in the scene. Let IPS takes a time T_{ip} to complete its procedures while SPS and MPS take T_{sp} and T_{mp} respectively. Each of these processes produces a list of regions to be provided to APS. Let us assume the time required to transmit a list to be T'_{dc}. As the three processes start at different times therefore APS needs to wait for IPS before starting its execution and then it gathers the output from SPS and MPS one after the other. Finally the APS provides a focus of attention to the HCS in communication time T''_{dc}. HCS moves the sensor and then the cycle starts again. The lists of regions produced by IPS, SPS, and MPS have much smaller size as compared to the original images and the data concerning the focus of attention is only a single region therefore $T''_{dc} < T'_{dc} < T_{dc}$.

Figure 4 demonstrates the model of parallel activation of the five processors using the above mentioned time periods as building blocks. It may be noted that the sizes of time slots are only symbolic to represent which period is longer. Obviously T_{ip}, T_{sp}, and T_{mp} are significantly longer than the other time slots.

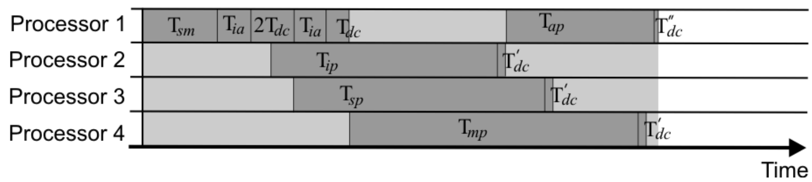


Fig. 4. Conceptual analysis of processor occupancy in the proposed distributed model. Light grey represents idle time available for other tasks.

Using the time modeling in figure 4, we can derive the expression for the total time T taken by the system with the proposed distributed setup as follows

$$T = T_{sm} + T_{ia} + max(T_{ip}, T_{sp}) + T_{ap} + 3T_{dc} + 3T'_{dc} + T''_{dc}$$

On the other hand if the same processes were to be executed on a single processor mechanism the total time T' would be

$$T' = T_{sm} + 2T_{ia} + T_{ip} + T_{sp} + T_{mp} + T_{ap}$$

Now the expression for the time gain T_g will simply be the difference of the two above times

$$T_g = T_{ia} + min(T_{ip}, T_{sp}) + T_{mp} - 3(T_{dc} + T'_{dc}) - T''_{dc}$$

Hence it may be concluded that if we have a fairly small consumption of time for data distribution and communication then the proposed distributed architecture can yield a significant gain in processing time. The gain can be approximately equal to the time needed for two of the heavy procedures.

References

1. Law, M.B., Pratt, J., Abrams, R.A.: Color-based inhibition of return. Perception & Psychophysics (1995) 402–408
2. Itakura, S.: Attention to repeated events in human infants (homo sapiens): effects of joint visual attention versus stimulus change. Animal Cognition (2001) 281–284
3. Wolfe, J.M., Horowitz, T.S.: What attributes guide the deployment of visual attention and how do they do it? Nature Reviews, Neuroscience **5** (2004) 1–7
4. Neri, P.: Attentional effects on sensory tuning for single-feature detection and double-feature conjunction. Vision Research (2004) 3053–3064
5. Itti, L., Koch, C.: A saliency based search mechanism for overt and covert shifts of visual attention. Vision Research (2000) 1489–1506

6. Zaharescu, A., Rothenstein, A., Tsotsos, J.: Towards a biologically plausible active visual search model. In: WAPCV 2004, LNCS 3368, Springer (2005) 133–147
7. Chen, H.Y., Leou, J.J.: A new visual attention model using texture and object features. In: 8th International Conference on Computer and Information Technology Workshops, IEEE (2008) 374–378
8. Backer, G., Mertsching, B., Bollmann, M.: Data- and model-driven gaze control for an active-vision system. Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 1415–1429
9. Aziz, M.Z., Mertsching, B.: Color saliency and inhibition using static and dynamic scenes in region based visual attention. Attention in Cognitive Systems, LNAI 4840 (2007) 234–250
10. Aziz, M.Z., Mertsching, B.: An attentional approach for perceptual grouping of spatially distributed patterns. In: DAGM 2007, LNCS 4713, Heidelberg - Germany, Springer (2007) 345–354
11. Aziz, M.Z., Mertsching, B.: Visual search in static and dynamic scenes using fine-grain top-down visual attention. In: ICVS 08, LNCS 5008, Santorini - Greece, Springer (2008) 3–12
12. Matsuo, H., Nakada, K., Iwata, A.: A distributed image processing environment VIOS III and it's performance evaluation. In: International Conference on Pattern Recognition, Brisbane - Australia (1998) 1538–1542
13. Krüger, T., Wickel, J., Kraiss, K.F.: Parallel and distributed computing for an adaptive visual object retrieval system. In: PDIVM 2003, Nice - France (2003)
14. Li, X.L., Veeravalli, B., Ko, C.C.: Distributed image processing on a network of workstations. International Journal of Computers and Applications (2003) 136–145
15. Galizia, A., Viti, F., D'Agostino, D., Merelli, I., Milanesi, L., Clematis, A.: Experimenting grid protocols to improve privacy preservation in efficient distributed image processing. Parallel Computing: Architectures, Algorithms, and Applications **38** (2008) 139–146
16. Thompson, C., Brooks, C., Greer, J.: Distributed image processing for automated lecture capture post-production. In: IEEE International Symposium on Multimedia, IEEE (2008) 698–701
17. Caton, S.J., Rana, O.F., Batchelor, B.G.: Distributed image processing over an adaptive campus grid. Concurrency and Computation: Practice and Experience **21** (2009) 321–336
18. Flynn, M.: Some computer organizations and their effectiveness. IEEE Transactions on Computers **21** (1972) 948–960
19. Mullender, S.J., Tanenbaum, A.S.: The design of a capability-based distributed operating system. Computer Journal (1986) 105–121
20. Aziz, M.Z., Mertsching, B.: Fast and robust generation of feature maps for region-based visual attention. Transactions on Image Processing **17** (2008) 633–644
21. Aziz, M.Z., Stemmer, R., Mertsching, B.: Region-based depth feature map for visual attention in autonomous mobile systems. In: AMS 2005, Stuttgart - Germany, Informatic Actuell, Springer (2005) 89–95
22. Shafik, M.S., Mertsching, B.: Enhanced Motion Parameters Estimation for an Active Vision System . Pattern Recognition and Image Analysis **18**(3) (2008) 370 – 375

Audio-visual Data Processing for Ambient Communication

Joerg Schmalenstroeer[1], Volker Leutnant[2], Reinhold Haeb-Umbach[3]

University of Paderborn, Department of Communications Engineering, Germany
[1]schmalen@nt.uni-paderborn.de,[2]leutnant@nt.uni-paderborn.de,
[3]haeb@nt.uni-paderborn.de

Abstract. In this paper we present our system for acoustic scene analysis and ambient communication. The acoustic scene analysis delivers information about the user's location which is utilized in ambient communication such that audio-visual data are captured and rendered by the most appropriate I/O-device, which allows the user to move freely from one room to another during a teleconversation. The system employs a steerable camera, controlled jointly by acoustic speaker localization and face detection. The ambient communication system is implemented on top of a context management system which maintains context information provided by context sources and consumed by applications.

1 Introduction

Ambient communication as a future trend of ambient telephony [1] formulates the vision of a user-oriented, service based infrastructure for audio and video communication [2]. Thus, it follows the paradigm of Ambient Intelligence (AmI) that claims the key elements of an "intelligent" system to be embedded, context-aware, personalized, adaptive and anticipatory [3]. Hence it overcomes the limitations of a hardware-oriented telephony application or device by hiding the hardware from the user in the walls and at the same time retaining and extending its original functionality.

The aforementioned elements of AmI can only be realized if a sufficient amount of reliable context information is available. This constraint asks for two tasks to be solved by an intelligent system. First of all sensors, devices and applications have to be integrated in a network, utilizing a common middleware for communication and interaction. Second, the inherent knowledge of the information sources has to be prepared such that machines can understand and process it. A widely accepted approach for this task is the use of an ontology, which in principle constitutes a joint knowledge base by definition of terms and their relationships.

Regarding ambient communication scenarios the acoustic signals recorded by the microphones are interesting context sources as they provide information about users and events. Obviously localization and identification by audio signals assumes that the user is speaking, however during a communication this should be fulfilled. Our acoustic scene analysis localizes and identifies active speakers and thus generates information about: “Who speaks, When and Where?”. This context information is used by a camera to focus the active speaker and it is also provided to other applications.

In the next section we briefly describe the usage scenario and the used hardware. Section 3 gives an overview about the system building blocks for audio and video processing. After presenting the middleware in section 4, section 5 explains our system for ambient communication, and we finish with some conclusions.

2 Usage scenario

We envisage a networked home environment as the typical environment where ambient telephony is to be used. It is characterised by a multiplicity of hardware components stemming from the domains consumer electronics, household appliances, personal computing and telecommunication, all more or less connected via networks. Especially, if we focus on the audio-visual equipment, we find a large variety of hardware configuration to be installed in the home. This may range from single or no microphones per room to rooms equipped with distributed microphone arrays, loudspeakers and cameras.

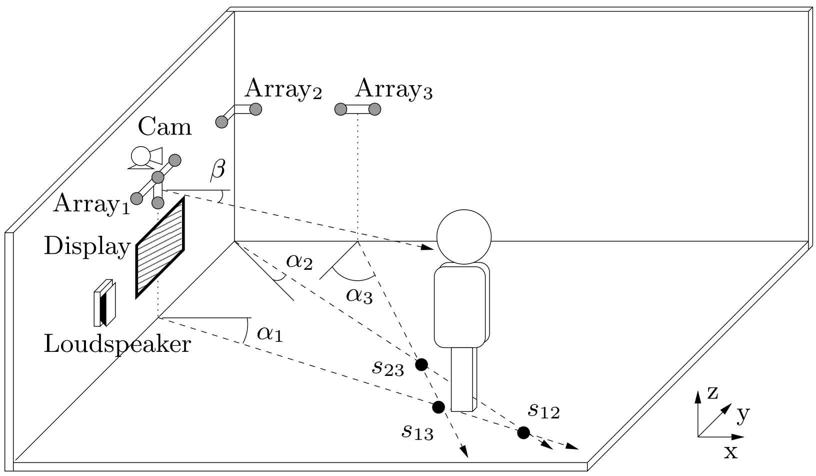


Fig. 1. Ambient communication setup

Our further explanations are based on a room with a high level of equipment as depicted in Fig. 1. For audio signal processing three microphone arrays, namely one T-shaped and two linear arrays, are used. The T-shaped one is mounted at the wall between a display and a pan-tilt-zoom camera. It is assumed that the user looks in the direction of the camera, and thus in the direction of the array, while having an audio-visual communication with a distant person. Together with the two other arrays the speaker can be located which is intern used to focus the camera on the user.

3 System overview

The system for audio-visual data processing is divided in two parts working in parallel, which are synchronized and connected via a shared memory (SHM) approach, see Fig. 2. In the video subsystem the webcam stream is processed on a frame-by-frame basis, where the frame rate may vary because of changing network quality. The audio subsystem works at a constant sampling rate of 16 kHz and a block length of 10 ms. Information gathered by one of the subsystems is stored in the shared memory and used by the other until it is overwritten.

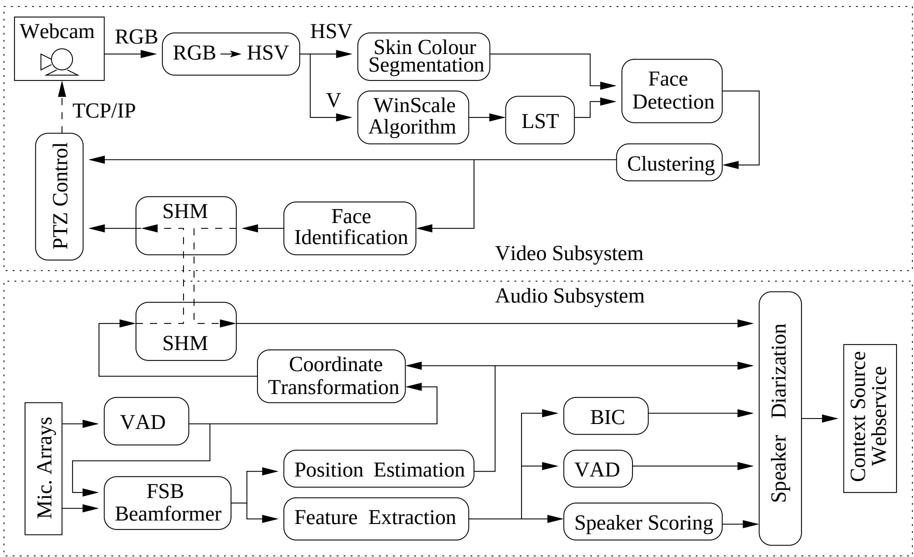


Fig. 2. Speaker localization and camera control

3.1 Video subsystem

The frames of the video stream are converted from RGB to HSV to ease the skin color segmentation and in parallel retrieve the grey scale version (V component) of the frames. We perform a histogram look-up to find the regions of skin color, which simultaneously reduces the computational demand and the false alarm rates of the face detector by constraining the areas of the frame to be examined for faces. We employ a face detector that is optimized to find faces at a size of 19×19 pixels. Thus we have to scale down the original frame to subframes with different resolutions to find faces at larger sizes. Here we employ the WinScale algorithm from [5]. Each subframe is processed with a local structure transformation (LST) as proposed in [6] and subsequently scanned for faces by a 4-stage detection cascade as suggested by Viola and Jones in [7].

The face detection method tends to detect a face multiple times in marginally varying positions and sizes, thus a Leader-Follower clustering is employed to merge the results. According to the information from the face detection we cut out the parts of the greyscale picture at the face positions and scale them to a size of 60×60 pixels. Further the well-known Fisherfaces approach [8] is applied to identify the persons. In a first step we use a principal component analysis (PCA) matrix that was determined on training data to reduce the feature vector size from 3600 to 200. In the second step we further reduce the dimension to the number of trained users minus one, by applying a LDA matrix from a linear discriminant analysis (LDA) that was also estimated on the training data. A single Gaussian is estimated for each user to model him in a probabilistically manner. Consecutive observations of faces in the same look direction are tracked by interpreting the posterior likelihoods of the last timestep as a priori likelihoods of the current timestep. The current posteriors are stored in the shared memory.

3.2 Audio subsystem

The audio subsystem uses the spatially distributed microphones for localization and identification of speakers (cf. Fig. 2). First of all we use a beamformer for speech enhancement to reduce the detrimental effects of reverberation and noise. We employed a filter-sum beamformer (FSB) [9] which performs a principal component analysis on each microphone array signal and thus blindly adapts to the strongest sound source. The correlation of the FSB filter coefficients enables an estimation of the Direction-of-Arrival (DoA) for each array and jointly a localization of the user, if multiple distributed arrays are available. In our setup the DoA information of each array is transformed in corresponding azimuth angles $[\alpha_1, \alpha_2, \alpha_3]$ while the T-shaped array is also able to provide a tilt angle estimate β (cf. Fig. 1). Next we calculate the intersection points $[s_{13}, s_{23}, s_{12}]$ of the direction estimates and retrieve the speaker position estimate as their centroid.

Speaker identification requires a segmentation of the audio stream in homogeneous parts. Since the applications in mind asks for online data processing with short latencies, multi-stage batch procedures or iterative methods as normally proposed for speaker diarization [10] are not applicable. Our approach uses a Hidden Markov Model (HMM) where each state corresponds to a certain user. A partial traceback is implemented to enable joint speaker segmentation, identification and localization at low latency [11]. In contrast to other methods, e.g. [12], we estimate a time variant transition matrix from speaker change hypotheses. Information about possible speaker changes are retrieved from the variance of the speaker localization and the variance of the Bayesian Information Criterion (BIC) [13].

We use the ETSI advanced feature extraction front-end [14] on the beamformer output signal to to obtain a 39-dimensional feature vector. The vector is extended to 42 dimensions by adding a voicedness feature and its first and second order derivatives. The speaker scoring calculates the likelihoods from the feature vectors, based on the Gaussian mixture models (GMM) of the users. Further we interpret the posteriors of the face identification as a priori knowledge for the speaker diarization. It follows that the product of the GMM likelihoods and the posteriors of the face identification are the state observation probabilities of the HMM. The partial traceback of the speaker diarization module estimates the single best state sequence given the acoustical and visual observations and then hands over the information to the context source. This context source can be used by any application or device via its webservice interface.

3.3 Camera control

We employed a pan-tilt-zoom camera for visual communication and also for identifying users. The camera orientation and depth view is controlled by incorporating visual as well as acoustical position information. Localized users that are not within the camera view are automatically focused so that the currently active speaker gets into the camera view after just a short delay.

4 Middleware and context management

The middleware represents the backbone of a networked home environment. Its ability to provide context information and to integrate different services is of utmost importance to realize a perceived level of "intelligence". Our system builds upon the open source middleware that was developed during the Amigo project [15]. It uses webservice technologies from the semantic web and comes along with basic services for context management, aggregation and distribution [16]. In Fig. 3 the architecture of the Amigo context management system is depicted. The context broker (CB) is the central unit for registering and searching context sources, whereas context sources are defined as any element that delivers a kind of information that may be interesting for the system.

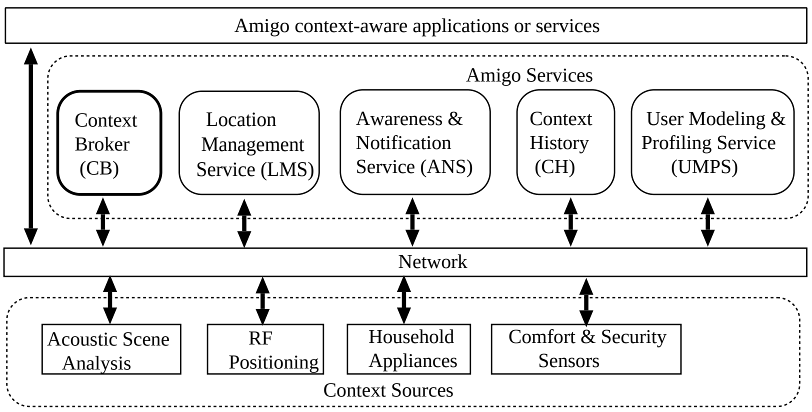


Fig. 3. Amigo context management system

Applications or services search via the context broker for suitable context sources and contact them by their standardized webservice interfaces. Either a direct request for information is forwarded to the context source or the application registers at the context source for notifications in case of new context information. In both cases a SPARQL query is formulated [17] and the answer is given in RDF/XML description format [18].

A key context information is the user's location, which is handled by the Amigo middleware within the location management service (LMS). This service continuously searches for context sources providing location information, e.g. the acoustic speaker diarization or a RFID positioning system. All context information is aggregated by the LMS and delivered as new contextual information to other applications.

5 Webservice audio interface

The webservice based audio interface connects the audio processing part for communication with the context-aware applications using the Amigo middleware. We coined this assembly of building blocks *Seamless Audio Interface* (SAInt) to outline one of the key features of the system. SAInt realizes a follow-me functionality for audio communications such that the user can freely change rooms while the communication follows him seamlessly.

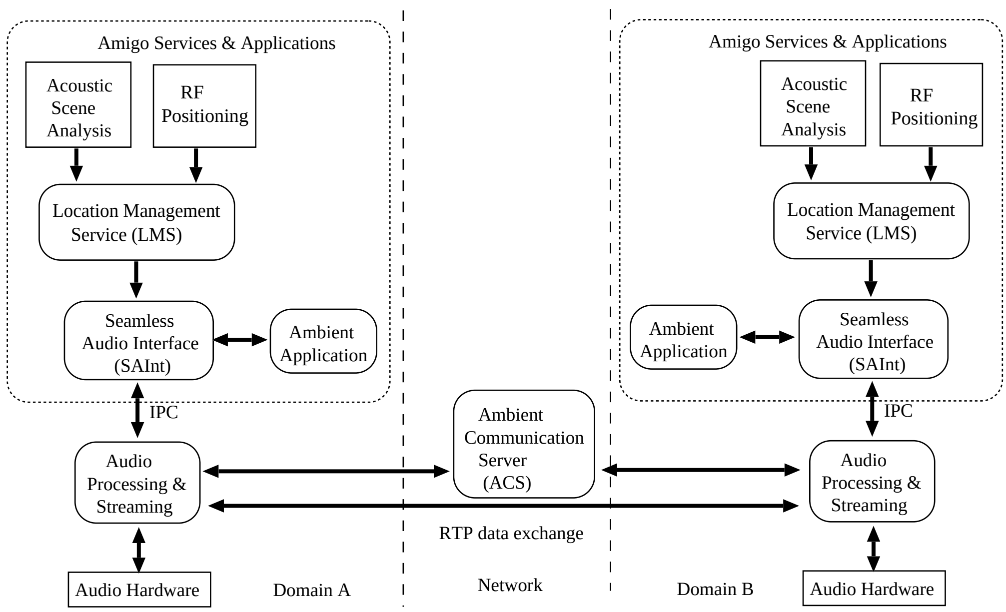


Fig. 4. SAInT: Seamless audio interface

In Fig. 4 the building blocks of SAInt are depicted. The audio processing and streaming block receives the acoustical signals from the sound capturing hardware and first of all performs an echo cancellation and noise suppression for signal enhancement. The streaming itself is initiated, controlled and terminated by the applications or by contextual information. Therefore the signal processing block is asynchronously connected via an interprocess communication (IPC) with the SAInt middleware service.

SAInt obtains information about user locations directly from the LMS and offers a webservice interface for applications. In parallel SAInt acts as a context source, publishing information about the rooms equipped with audio hardware, about ongoing connections and about users available for communication. Thus applications can get an overview about the hardware and the users in range of it by registering to all SAInt services in the connected home.

An application asks for an audio or audio-visual connection by instructing the SAInt service to connect two persons. SAInt uses the LMS to look up the location of the persons and sets up the connection. If a person moves from one room to another, the change of context information triggers a redirect of the audio streaming, while the application using SAInt does not have to take care about it. Thus a communication is internally bound to a user and follows him on his way through the house.

The audio streams are compressed with an 16 kHz Speex wideband audio coder and the video data is compressed with the Theora coder, both including a packet-loss concealment. We use the real-time transport protocol (RTP) for interchanging the audio and video data between two SAInt instances. External connections to other houses are initiated via a central server that is called *ambient communication server* (ACS). It enables firewall and network address translation (NAT) traversal as well as session initialization and handovers.

6 Discussion

In this paper we have presented our system for ambient communication and acoustic scene analysis. Both tasks are closely related, as they are based on the same acoustical signals. We have shown how context information about the user's location is obtained from analyzing the data captured by microphone arrays and a steerable camera. This location information is intern utilized to control the camera and to select the most appropriate I/O-device while the user is moving freely in the home doing a teleconversation with a remote partner. We have further described an open middleware which connects context sources to context consumers and which thus enables services to take context-aware decisions.

References

1. A. Härmä, "Ambient Telephony: scenarios and research challenges", Proceedings Interspeech 2007, Antwerpen, 2007
2. S. Borkowski, T. Flury, A. Gerodolle, G. Privat, "Ambient Communication and Context-Aware Presence Management", Communications in Computer and Information Science, Vol. 11, pp. 391-396, Springer LNCS, Germany, Berlin, 2008

3. S. Marzano, E. Aarts, “The New Everyday - Views on Ambient Intelligence”, Koninklijke Philips Electronics N.V., 010 Publishers, The Netherlands, Rotterdam, 2004
4. N. Georgantas, S. B. Mokhtar, Y. Bromberg, V. Issarny, J. Kalaoja, J. Kantarovich, A. Gerodolle, R. Mevissen, “The Amigo Service Architecture for the Open Networked Home Environment”, Proceedings 5th Working IEEE/IFIP Conference on Software Architecture (WICSA), 2005
5. C. Kim, S. Seong, J. Lee, L. Kim, “WinScale: An Image-Scaling Algorithm Using an Area Pixel Model”, IEEE Transactions on Curcuits and Systems for Video Technology, Vol. 13, No.6, pp. 549-553, 2003
6. B. Froeba, C. Kueblbeck, “Face tracking by Means of Continuous Detection”, Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition, USA, Washington, DC., 2004
7. P. Viola, M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features”, Proceedings IEEE Conference on Computer Vision and Pattern Recognition, Hawaii, Kauai, 2001
8. P. Belhumeur, J. Hespanha, D. Kriegman “Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No. 7, pp. 711-720, Jul. 1997
9. E. Warsitz, R. Haeb-Umbach “Acoustic Filter-and-Sum Beamforming by Adaptive Principal Component Analysis”, Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing, Philadelphia, USA, 2005
10. S. Tranter, D. Reynolds , “An overview of Automatic Speaker Diarization Systems”, IEEE Transactions on Audio, Speech and Language Processing, Vol. 14, No. 5, pp. 1557-1565, Sep. 2006
11. J. Schmalenstroeer, R. Haeb-Umbach “Joint Speaker Segmentation, Localization and Identification for Streaming Audio”, Proceedings Interspeech 2007, Belgium, Antwerp, 2007
12. S. Meignier et al., “Step-by-Step and Integrated Approaches in Broadcast News Speaker Diarization”, Computer Speech Language, Vol. 20, Issues 2-3, pp. 303-330, Sept. 2005
13. M. Nishida, T. Kawahara, “Speaker Model Selection Based on the Bayesian Information Criterion Applied to Unsupervised Speaker Indexing”, IEEE Transactions on Speech and Audio Processing, Vol. 13, No. 4, pp. 583-592, July 2005
14. ETSI ES 202 050 V1.1.3 , ”ETSI Standard Speech Processing, Transmission and Quality Aspects (STQ); Distributed speech recognition; Advanced front-end feature extraction algorithm; Compression algorithms”, Nov. 2003
15. Amigo Project website: http://www.amigo-project.org, 2007
16. F. Ramparany, R. Poortinga, M. Stikic, J. Schmalenstroeer, T. Prante, “An open Context Information Management Infrastructure - the IST-Amigo Project”, Proceedings Conference on Intelligent Environments, Ulm, Germany, 2007
17. SPARQL Protocol and RDF Query Language: http://www.w3.org/TR/rdf-sparql-query/
18. Resource Description Format (RDF) Specifications: http://www.w3.org/RDF/

OS service optimization in a heterogeneous distributed System on Chip (SoC)

Sufyan Samara, Fahad Bin Tariq

Design of Distributed Embedded
Heinz Nixdorf Institute
University of Paderborn
Paderborn, Germany
Email: sufyan@mail.uni-paderborn.de

Abstract. In order for an Operating System (OS) service to be distributable across a network of heterogeneous SoCs, certain properties have to be fulfilled. Such properties include supporting many hardware architectures, smart resource optimization, and the ability to change its configuration to suit the non-constant SoC demands and requirements. In this paper, some of the problems that an OS service faces in a distributed heterogeneous system of SoCs are shown. An OS service design model which enables solving such problems is discussed, and a linear running time algorithm which works with this model is given.

Keywords: OS service, SoC, heterogeneous, distributed, resource optimization, load balancing.

1 Introduction

Heterogeneous devices are composed of different hardware architectures. Heterogeneity is introduced to provide for application specific requirements in embedded systems. It is not uncommon to find such devices communicating with each other forming a distributed heterogeneous network. With the variety of applications and the increased complexity of the embedded systems, Operating Systems (OS) become a necessity. An OS provides a layer of transparency to applications and hides the complexity found in SoC systems. Due to the fact that SoCs are most likely to be a resource limited embedded system, an OS is considered an overhead. Hence, the OS should take this into account and try to minimize its resource usage to free as many resources as possible for other tasks/applications. Moreover, if a new application demands resources that were allocated an OS service, the OS should have the possibility to reconfigure to adapt to the variations and use as little application demanded resources as possible. Heterogeneous systems differ in speed, power consumption, and application load. The OS needs to be aware of these and execute its services in a manner that provides as much resources to the application as possible. Additionally, not all of the OS services are needed in every SoC, nor do we need a specific OS service all the time. This provides the possibility to store the OS services on one or more repositories and dispatch the service when it is requested by the SoC.

A single SoC in the distributed system may contain two or more computational elements, e.g. Xilinx Virtex II pro (a Field Programmable Gate Array (FPGA) which contains a General Purpose Processor (GPP) inside it). This calls for a new service design to exploit all the capabilities offered by the system.
A different design model is needed to allow an OS to execute and exploit the entire computational element on a SoC. Nevertheless, we need a design which provides support to a network of communicating heterogeneous SoCs without much overhead complexity. We also need the OS service to utilize such systems with some flexibility and adaptability characteristics. In this paper we will introduce an OS service design model and an architecture which is suitable for distributed heterogeneous SoCs. We will also introduce a runtime algorithm which provides the support needed for such systems. The paper contains the following sections: section 2 shows some related work to this topic. In section 3, an overview of the problem and the proposed solution is given. The service design model and the system architecture are discussed in section 4. The algorithm used is explained in section 5, and finally the future work is given in section 6.

2 Related Work.

Designing OSs for embedded systems has been acknowledged by many previous works [2, 3, 4, 7]. The idea of an operating system running on multiple computational elements at the same time is not new. Work done in [5, 6] shows OS services implemented on both the GPP and FPGA. Both implementations exist at the same time on the system and context switching is done at runtime. The algorithm used is based on Binary Integer Programming (BIP). This strategy is not suitable for distributed systems as it requires resource reservation for a complete OS service on both computation elements. In other words, every OS service exists in pairs (an FPGA and a GPP implementation) on every SoC in the network at all times. On the other hand, many algorithms have been developed and used for scheduling and partitioning applications/tasks to be executed on software and hardware; however most of these algorithms are static in nature such as in [9]. Solutions to fixed problems are provided without any regard to dynamic changes (runtime). For example, resources allocated to an application may be required by newly arriving higher priority tasks. In this paper we present an OS service model partitioned to be executed and reconfigured on many computational elements for runtime resource optimization and load balancing.

3 Problem and the Solution Overview.

Heterogeneous embedded SoCs are limited in resources hence an OS may be seen as an overhead. Nevertheless, an OS is needed to support a wide range of applications. Providing that not all the services are needed all the time, we can store the OS services in a repository and dispatch the service when needed. In a heterogeneous distributed system, as well as at the level of a single SoC, we need to utilize the variety of computational elements. The difference in speed and power consumption introduced by the heterogeneity must also be exploited. Moreover, even for a single computation element, different coding styles may lead to different resource usage. All of this leads to the conclusion of providing multiple implementations of each OS service at a repository. A service implementation is an instance version of a service which is suitable to be executed on a specific computation element. A SoC with three computation elements (e.g. FPGA, intel GPP, Motorola GPP) requires at least three implementations for each service. This gives us the possibility of executing a service on any computational element across the distributed system. At this point, we handle the heterogeneity of the system, but still need to provide for the dynamic resource utilization that may arise at run-time in a single SoC. This is handled by OS service partitioning. For every service, we partition all of its implementations into small blocks, which we call small execution segment (SES) as can be seen in figure 1. All the different SES implementations of a single service (same row) have the same behaviour, i.e. the same input to all SESs in a row will produce the same output. The partitioning into SESs results in the partitioning of a service execution on a SoC enabling adaptation and optimization for speed, power, or load balancing.

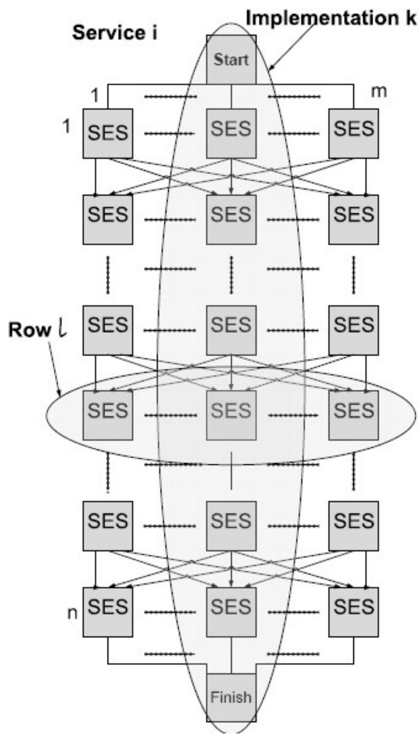


Figure 1: Partitioned implementations for a service. Any SES across one row has the same behaviour as the other SESs in the same row. Column ‘k’ is the complete service implementation for one computation element.

Every SES contains execution information (constraints) about itself. This includes the worst case execution time, the power estimation, and the amount of area allocation.

These constraints vary depending on the architecture implementation and even for the same implementation depending on coding style.
These constraints are used in the algorithm (section 5) to find and optimize a configuration which fits into the SoC resource constraints.

4 Architecture

The distributed system under consideration is a network of SoCs connected to one or more OS Service Repositories (OSR), shown in figure 2. Having a single OSR or a network of OSRs, should be of no concern to a SoC. Each SoC connects to the OSR which is closest to it. When a SoC requires a service or a part of a service, it sends a request to the OSR connected to it. In case the OSR does not have the requested information, it communicates with the other OSRs to provide for the SoC request.
As mentioned earlier, a SoC is assumed to contain one or more processors (Multi-Processor SoC i.e. MPSoC), and/or reconfigurable elements (e.g. FPGA). To support the execution of SESs on the SoCs, each SoC is assumed to have the architecture shown in figure 3.

Figure 2: Distributed SoCs connected to OSR(s)

Traditionally, an application runs on top of an operating system (OS), using its services to fulfill its goal. Besides providing these services, the operating system acts as an indirection to the hardware by managing the available resources to meet the application requirements. In our work, since the OS services have been partitioned into small blocks (SESs), another layer is required below the OS to support these blocks. The required support, such as inter-SES data transfer and synchronization, stems from the decision of partitioning OS services. One important aim of this layer is to make the distribution of the partitioned OS services appear transparent to themselves and to the applications using them.

The middleware shown in figure 3 behaves as the necessary layer required by the partitioned OS services (SESs). The middleware is a component framework that treats the SESs as components with defined interfaces. A component based approach allows us to use pre-defined components with well defined patterns of interaction, which is due to being in conformance with a component model [8]. Treating SESs as components is achieved by encapsulating them in component containers, which provide the middleware the required interface to manage the life cycle of an SES on a SoC. Simultaneously; the SES provides an interface to transfer data to the subsequent SES in the execution path. The usage as a middleware for partitioned OS services is a specialization of the generic component framework for runtime adaptation of real-time applications. In this work, the framework treats the OS services as an application that adapts during runtime.

To achieve a degree of determinism, the middleware employs a time-triggered policy in various aspects of the SESs such as when dealing with their communication, managing newly incoming SESs, reconfiguring inter-SES connections etc. This is realized by using a Time Division Multiple Access (TDMA) strategy to access the resources offered by the middleware. Three phases namely the communication phase, the reconfiguration phase and the computation phase, repeat in a cyclic manner with strict timed boundaries signaling a switch from one phase to the other. Actions within these phases are carried out in sub-phases that are again driven by time. Details of these phases are beyond the scope of this paper.

The middleware runs on top of a layer that provides basic level support which are necessary to the middleware in performing its function of managing the SESs. The MicroRM (Micro Resource Manager) shown in figure 3, plays the role of such a layer in our architecture. It provides support to the middleware such as hardware abstraction, basic memory management which is needed during memory allocation, network communication needed to communicate with OSRs to import new SESs etc.

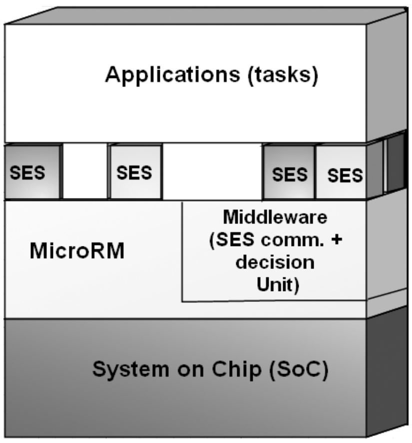


Figure 3: Basic system layers in a SoC

The OSR stores the partitioned OS service implementations (SESs) and manages the SoCs requests. It also contains the algorithm which decides which SESs combinations are most suitable for which SoC, as shown in the following section. The algorithm decides at runtime based on the information received in the requests which are provided by the middleware on the requesting SoC. In case of OSR network, the SESs are initially distributed across the OSRs. After some runs, it may appear that it is better to relocate some SESs to other OSR. Enabling such rearrangements depends on the communication time and the requests frequency.

5 Algorithms

The aim of the algorithm, residing on an OSR, is to configure the service to be executed on the requesting SoC with regard to the SoC's architecture, the availability of its resources, and the communication overhead. The algorithm assumes the presence of some meta-information provided by each SES such as its worst case execution time, and its estimated power consumption. It also assumes that the request sent by the SoC contains information about its current resource usage, architecture, and any specific property (e.g. remaining power, application deadlines etc.).
The OSR also contains information about all SoCs with a similar architecture and their current resource status. This information may be needed if the requesting SoC does not have enough resources to execute any of the service configurations due to a lack of resources. The OSR finds the configuration based on three steps; the first step involves finding the service implementations which match the requesting SoCs architecture. The second step involves finding the configuration(s) which fit into the SoC's resources and constraints. In the case where the SoC does not have enough resources, we look into the third step of finding a close SoC to the requesting SoC in terms of communication time to migrate the data to execute the service. These steps can be seen in algorithm 1 and 2.
To keep the OSR(s) up-to-date with the SoCs resource usage, information is communicated to the OSRs when ever resource usage is changed at an SoC. Moreover, each SoC provides the OSR information about its neighbors. This allows the OSR(s) to build a table of alternatives in case step three is needed. This information is provided at runtime and is kept up-to-date in case of any changes.

```
Algorithm AddSoCAndCreateAtlernativesTable
 Foreach SoC join the Distributed system
   -Add the SoC to the List of SoCs
   -Create and assign an information table To that SoC
      which contains all the information about the SoC
      resources, its computation elements, and
      pointers to its SoCs neighbor
 end
```

Algorithm 1

```
Algorithm FindServiceConfigurationForSoC
   -Choose the service implementations applicable
        for the requesting SoC architecture
   -Calculate all the configurations that can run on
        the requesting SoC.
   -IF no configuration found
      -Get the requesting SoC neighbors
      -Search the neighbor SoCs for one with enough
          resources and minimum communication time.
      -IF found
        -Recalculate and migrate to execute on that SoC
       ELSE
        -ERROR
    ELSE
      -Optimize a configuration according to the
         requested special requirements and send it to
         execute on that SoC

end.
```

Algorithm 2

Finding the applicable service configuration in algorithm 2 can be done using the dynamic programming paradigm; as we have done in [1]. The procedure involves two steps; creating information tables and doing a controlled switching based on the information in these tables. There are two tables for every resource (e.g. power) we want to optimize. The first table stores information from the "Start" SES along with all the decisions made up until the "Finish" SES. The second table stores information about the paths we took in calculating the information in the first table.
Next, as mentioned above, the algorithm exchange and switch over the created tables to find all the configurations that fit into the requesting SoC's demands and resources. This is done quickly because many of these operations will be excluded due to constraints violation. Moreover, these operations are controlled by minimum and maximum possible resource usage. This helps identifying if the configuration found is the only possible configuration or if all configurations can fit before performing any operation. Algorithm 2 uses the results of the previous step to optimize and balance the load on the requesting SoC. The running time for Algorithm 1 and 2 are in the order of O(n), where "n" is the number of SESs of an OS service implementation.

5.1 Self-healing

Another benefit for the SESs partitioning is the self-healing. This is possible because of the information available about the SESs requirements and the system resources. Using this information enables error detection and system-recovery operations. For example, let an SoC receive a service configuration best suited to its current resource status. During service execution, the resource status changes so that the SESs yet to execute will not be able to operate without violating constraints. This is easy to detect looking at the needed resources each remaining SES requires. In such cases and because of SESs partitioning, the data obtained up to this point can be saved and a new request sent to the OSR for recovery solution. The OSR calculates and sends another configuration. The new configuration continues executing from the point the last configuration stopped by using the saved data. Again, if such configuration cannot be found, the possibility of migrating to another SoC is considered.

Another scenario would be an internal code error in a SES. The errors detected at this point are errors which make the SES exceed its worst case resource usage estimation, e.g. WCET. This can also be resolved by requesting a new configuration which does not contain the faulty SES. In case of the same SES error in the future, the OSR may mark the SES as faulty and a notification can be sent to a human administrator.

Hardware errors such as a damaged computational element may also be resolved by switching all or the remaining SESs to execute on another operational computational element.

Figure 4 shows an illustration of a scenario where SoC1 requests a service from OSR. The OSR calculates a suitable configuration of the requested service based on the resources available on SoC1 and sends it to SoC1. After SoC1 begins executing, the middleware predict a possible future failure or constraints violation because of a current change in SoC1 recourses. The middleware sends a recovery request to the OSR from the point where the failure is predicted to happen. In the mean time SoC1 continues executing. The OSR calculates another partial configuration based on the new SoC1 resources SoC1. In case of no suitable configuration being found, the OSR sends a suitable partial configuration to SoC1's immediate neighbors (SoC2, SoC3) and acknowledges SoC1 with either a new partial configuration or the SoCs containing the partial recovery configurations. SoC1 saves the data calculated up till the SES with the predicted failure and sends the data to either the new configuration or the SoC with minimum communication time; SoC2. SoC2 calculates the rest and sends the result to SoC1. In case of a predicted failure in SoC2, data are sent to SoC3.

The algorithms used for the self-healing process in SoC and OSR are shown in Algorithm 3 and 4 respectively.

```
Algorithm ServiceSelfHealing_SoC
   -IF FAULT occur
        -Save the data obtained up till and before the
            SES containing the predicted error.
        -Send a request to OSR with the current
           resource status and the last executed SES
           info.
        -Wait for replay
           -IF new configuration received
               -Continue executing from the point the
                   old configuration stopped.
            ELSIF ACK_EXT_SOCS %%execute on other SoCs
               -Send the data to the SoC with minimum
                   communications
               -Wait for the result.
            ELSE  ERROR        %%No solution found
```

Algorithm 3

```
Algorithm ServiceSelfHealing_OSR
      -IF a request received
          -calculate a configuration from the SES where
              the error is predicted to occur.
          -IF configuration found
             -Send the new configuration to the SoC
           ELSE
             -Find the immediate neighbor SoCs.
             -Optimize the configuration for new SoCs
             -Send new configurations to the SoCs.
             -Send the SoCs information to the
                 requesting SoC.
```

Algorithm 4

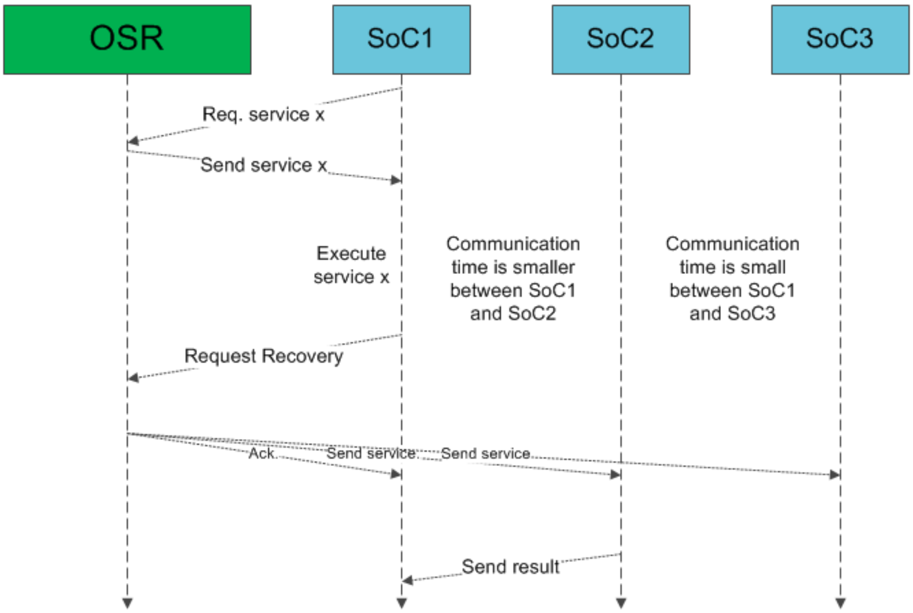


Figure 4: An illustration of how faults or predicted constraints violations are handled.

6 Conclusion and Future work

A new design methodology for OS services is presented. The introduced enables a partitioned OS service into SESs to run across a distributable network of heterogeneous SoCs. This opens many interesting problems which need to be pursued. One problem is to find an automated way to partition an OS service into a collection of SESs in different implementations. In the future we aim to distribute all the SESs across the distributed system without the need of OSR(s). This also introduces many challenges such as coordination, localization, load balancing and finding the best execution route. One possible direction is to employ solutions from the domain of biologically inspired algorithms and organic computing.

References

1. S.Samara, F. Bin Tariq, T. Kerstan, K. Stahl: "Applications adaptable execution path for operating system services on a distributed reconfigurable system on chip", In: Proc. ICESS09, pages. 461-466, IEEE 2009.

2. R. H. Campbell, N. Islam, D. Raila, and P. Madany.: "Designing and Implementing Choices: An Object-Oriented System in C++" Commun. ACM,36(9):117-126, 1993.

3. C. Ditze. "DReaMS-Concepts of a Distributed Real-Time Management System." In Proc. IFIP/IFAC Workshop on Real-Time Programming, 1995.

4. M. F. Kaashoek, D. R. Engler, G. R. Ganger, H. M. Priceno, R. Hunt, D. Mazieres, T. Pinckney, R.Grimm, J. Jannotty, and K. Mackenzie. "Application Performance and Flexibility on Exokernel Systems" In Proc. Fifteenth ACM symposium on Operating systems principles (SOSP), pages 52-65, 1997.

5. M. Goetz; "'Dynamic hardware-software codesign of a reconfigurable real-time operating syste."' *In: Proc. of ReConFig04, Mexican Society of Computer Science*, SMCC 2004

6. M. Goetz, A. Rettberg, and C. E. Pereira; "'A Run-Time Partitioning Algorithm for RTOS on Reconfigurable Hardware"', pages. 469-478 *IFIP International Federation for Information Processing* 2005

7. J. Lee, III, V.J.M, K. Ingstrm, A. Daleby, T. Klevin, and L. Lindh; "'A comparison of the rtu hardware rtos with a hardware/software rtos."', proc. Asia and South Pacific Design Automation Conference, 2003

8. Heineman G. T. and Councill W. T., Component-Based Software Engineering: Putting the Pieces Together (ACM Press, 2001). ISBN 0-201-70485-4.

9. Wu Jigang, T. Srikanthan; "'Low-complex dynamic programming algorithm for hardware/software partitioning"', pages. 41-46 , 2006 *Information Processing Letters 98*, ELSEVIER, 2005

Context-aware self-management for dynamic service platforms

Wolfgang Thronicke

Siemens AG, SIS C-LAB, Fürstenallee 11,
33102 Paderborn, Germany
wolfgang.thronicke@siemens.com

This paper describes the ongoing work in the research project OSAMI (Open Source Ambient Intelligence Commons) which targets an OSGi based service platform for multi-domain use. As such systems are usually employed in specific environments-home or off-road-traditional system management as for stationary "enterprise" applications is not an option. With dynamic services and applications which rely on multi-sensory input and processing the introduction to context-based techniques as a base concept for autonomous management is being examined and realized.

A Robust Fingerprint Matching Algorithm with Special Consideration of Missing Points

Neeraj Sharma, Jeong Il Kwon, and Choi Byoung Tae

Electronic Telecommunication and Research Institute
Daejeon, Republic of Korea
{neerajsharma,jik,btchoi}@etri.re.kr

This paper presents a distance histogram comparison method for an efficient fingerprint matching. Here, we introduce a randomized algorithm which exploits pairwise distances between the pairs of minutiae, as a basic feature for match. The method takes two steps for completion i.e. local matching and global matching. This method is especially useful for the places where input fingerprint patterns often miss a lot of points from a particular region/area of finger. The algorithm was examined on various randomly generated point patterns and real fingerprints, whose results are provided in the paper. These results resemble the accuracy and utility of method in practical usage.

Opportunities and Risks of Ubiquitous Computing and Ambient Intelligence in View of Modern Business Processes in the Insurance Industry

Sebastian P. Schroeder, Volker Gruhn

LPZ e-Business, Klostergasse 3, 04109 Leipzig, Germany
mail@sebastian-schroeder.eu, gruhn@ebus.informatik.uni-leipzig.de

Abstract. In the insurance industry there is great commercial potential for modern technologies that can be taken advantage of to develop innovative products and services, thanks to which business models as well as central business processes like premium calculation and risk management are clearly modified. The confidentiality of sensitive data, the authenticity and integrity of all involved entities as well as the reliability of essential infrastructure components constitute in this context critical success factors. In order to meet the demands many most variegated challenges must be met whose structuring and analysis are the focus of the research project described here.

Keywords: Mobile and ubiquitous computing, ambient intelligence, insurance industry, core business processes, data quality, information security

1 Changes due to mobilisation and technology embedding

The future technology revolution will be characterised by the embedding of computers into everyday life as well as the interactions of these embedded devices with their environment.

A central challenge in the insurance business consists of the fact that policyholders are in general better informed about their individual risk profile than the company insuring them. This asymmetric information distribution entails moral risk and adverse selection. Since in extreme cases this can entail market failure a trade-off favoured by modern technologies is highly desirable from the vantage point of the insurance business.

In the insurance business, for instance, the opportunities include the implementation of product designs not previously realisable due to a lack of databases or a generally improved understanding of risks, new distribution and marketing aspects, better customer loyalty and satisfaction, virtual advisors, better protection against insurance fraud as well as more just and individual risk adjustment by means of more or improved rate-setting criteria.

Even if with overly individualised rate adjustment the core competency of insurance companies theoretically goes to absurd lengths in the collective the actual opportunity consists of identifying bad risks and strengthening the collective by appropriate measures.

An image transition, brought about by more professional counselling and innovative benefits, constitutes alongside of optimising of information gathering, customer bonding and analysis processes a competitive factor that should not be underestimated. Mobile and embedded technologies can additionally make more dynamic organisational structures possible and in that way can have positive effects on profitability.

2 Implementation scenario “intelligent property insurance”

In connection with a scenario analysis initially the effects of mobilisation and technology embedding were discussed subsequent to which the concrete examples of pay-as-you-drive, additional motor vehicle benefits, intelligent property insurance and personal insurance were taken up. In doing so, among other things the subjects of innovative health services and assistance services, context-sensitive risk warnings as well as claim management were discussed.

Intelligent property insurance could be identified as a business field that was potentially achievable in the near future without the danger of allowing for reference to a specific person on the basis of the data. The subject was considered in detail with the aid of the case study “Using Intelligent Technology in Hazardous Materials Shipping,” where reference to other property insurance products was possible.

Hazardous materials and transport means are very much diversified. In addition and depending on the case in question, the most variegated technologies may come to be used. The costs for the basic technology and for transmission of data increase in such cases just as the possibilities of using them do as well.

In general, by using modern technologies in shipping a continuous and transparent tracking is made possible, both during transport as well as in case of the logistically necessitated interim storage. Furthermore, the conditions of the goods shipped and the means of transport can be recorded and evaluated with sensors.

As a result in the long run better data quality emerges. In addition, enhanced verifiability of correct handling and better control of the conditions become possible. Furthermore, dangerous shipping routes or crucial environmental conditions could be warned of and compliance with instructions could be checked. Due to the influence on active and pro-active claim management the possibility of proving relevant circumstances was improved. In the ideal case, accident frequency and accident ramifications can be significantly reduced and costs can be saved, more advantageous insurance products offered or general competitive advantages can be generated.

2.1 Risks of using mobile and embedded technologies

In terms of the social and organisational context the greatest damage potential primarily stems in regard to the integrity and confidentiality of internal data from employees provided with the wrong authorisations or who were deliberately deceived.

The risk increases even further if one considers that attackers with specialised professional knowledge act in the role of an employee or can be deliberately placed in the relevant position in the company.

Manipulations of the data in the external dimension may very well mean high individual damages but will only very rarely result in high total damages. In regard to technological risks, on the other hand, large-scale manipulation of the hardware may occur out in the field, something that can cause considerable financial damage and ultimately risk evaluation that is starkly impaired. External hardware and software can generally be manipulated in several different ways.

Additional risks stem from any eventual administrative interfaces as well as from essential infrastructural components like telecommunications and localisation services that can be deliberately manipulated or whose identity can be forged. Where RFID technology is also used then it should be mentioned that the integrity of RFID systems is likewise not all that trivial.

2.2 Appropriate countermeasures

Among the most important countermeasures in regard to high efficiency and effectiveness, in the internal dimension most especially data validation as well as strong authentication and stringent authorisation mechanisms must be mentioned. The internal focus should be on information security since here many new requirements arise that result from the use of modern technologies.

In the external dimension mainly manipulation-proof hardware and software as well as encryption procedures must be mentioned. Furthermore, mechanisms for cost savings and intelligent energy management, automated alarm functions and mechanisms for recording critical events should be deployed. In the external dimension functional security constitutes a crucial success factor.

Appropriate measures must be chosen that ensure a high degree of protection against manipulation so that the expense to be incurred exceeds the resulting benefit due to cost savings. The risk still remains that criminal elements specialise in manipulation of such equipment or modules and in addition can with financial resources filch sensitive information from insurance companies or technology producers.

By means of hefty contractual penalties and the failure of insurance cover this risk should be largely acceptable, at least in industrial property insurance. The same applies to the deliberate insulation of mobile communications or sensor units or destruction of the relevant modules.

2.3 Opportunities and remaining risks

By using the technologies described in connection with the case study counselling, premium calculation, risk models, risk checking, demonstrability and claim management can be optimised. All optimisations are in particular due to enhanced data quality and a greater degree of automation. In that way, the market position in relation to competitors can be improved and profitability significantly raised.

Remaining technology risks can be largely transformed. The unintentional leakage of internal data, incorrect conclusions due to new data as well as the selection of unreliable partners must be minimised by means of individual organisational measures.

Benefits are incurred not only at the level of the insurance company but, for instance, among policyholders. They benefit from better counselling, better demonstrability and better claim management and can in addition achieve savings on premiums in the case of good risk profiles. One can also imagine benefits at the level of the national economy due to fewer accidents.

Customers with poor risk characteristics will by contrast not be able to achieve any savings on premiums. The task of the insurance company will therefore be to communicate the advantages of good risk characteristics in a targeted manner and to create genuine incentives.

3 Summary

The vision of ubiquitous computing stands and falls with the confidence people have in technology as well as to suppliers or operators of the infrastructure, context-related information bases and modern services.

Companies that become active on the market with innovative products early on or accumulate experience with modern technologies in pilot projects must in cases of doubt admittedly expect high research and development costs but by doing so are pitching their company at a future market offering enormous growth and success opportunities.

Depending on how much mobile and embedded technologies continue to become established it can even come to pass that, due to late realignment of strategy, market displacement occurs both due to all competitors as well as, in particular, to new ones.

Not to be underestimated in this context is the fact that companies from other fields are enabled, by embedding modern technologies in their habitat, to perform insurance services without having to rely on the core competency of risk compensation in the collective and without having to contribute to financing historically developed organisational structures.

The developments in the automotive industry, for instance, and not just in regards to benefit related rate setting and assistance services, are in this respect only the beginning. Opportunities and risks of modern technologies for the insurance business should therefore on the one hand be considered individually and on the other hand not only discussed and evaluated inside the financial services sector.

Case Adaptation through Case Based Reasoning and Neural Networks for Predictive Prefetching

Sohail Sarwar[1], Hironao Takahashi[2], Zia-ul-Qayyum[1], Owais Ahmed Malik[1], Bilal Ali[1], H. Farooq Ahmed[1,2]

[1] School of Electrical Engineering and Computer Science
National University of Sciences and Technology, Islamabad, Pakistan
[2] DTS Inc, Kyoudou Building 9F, 2-18-7 Higashiueno,
Taitou-ku, Tokyo 110-0015 JAPAN
{sohail.sarwar, zia.qayyum, owais.ahmed, bilal.ali, farooq.ahmed}@seecs.edu.pk, hiro@dts-1.com

Predictive Prefetching being an endemic concern in information systems is desirous of higher level of predictive accuracy. So we have congregated Case Based Reasoning (CBR) and Neural Networks (NN) for improvising the degree of predictive accuracy in speculative prefetching. The most important aspects in this hybrid of CBR and NN is that of case retrieval and adaptation where former yields related solutions to current problem and former cedes the solution if we don't find an exact solution for current problem. We have specifically focused on adaptation and proved that Neural Networks have better predictive performance for solution than CBR while performing case adaptation.

Multi-core Technology: An overview

Syed Yasir Imtiaz[1], Abdul Hameed[2] and Nasro Min-Allah[2]

[1] Low Power Mobile Architecture and OS Lab,
Center for Advanced Studies in Telecommunications (CAST),
COMSATS Institute of Information Technology, Islamabad Pakistan.
[2] Department of Computer Science,
COMSATS Institute of Information Technology, Islamabad Pakistan.
{yasir_imtiaz, abdul_hameed, nasar}@comsats.edu.pk

Abstract. Over the past few decades, uni-processor performance has been increased exponentially and improvement in the design and architecture has been witnessed greatly. Since computing is becoming ubiquitous, more and more complicated applications are being developed today. Due to the emergence of such applications, the need for a faster system is essential. To handle the performance issues, the concept of multi-core is introduced recently, where multiple processing cores are built unto a single die. Multi-core processors present many performance benefits such as on core cache, reduced power consumption and smaller size. However increasing the number of cores on a single chip also presents some challenges such as shared memory bandwidth, cache coherence, cache contention and the scheduling problems for the multi-core processors. To the best of our knowledge, the multi-core architecture is not well portrayed in literature and no architectural comparison has been made so far. This paper reviews architectural designs and comparison of Intel and AMD processors to enable researcher for proposing novel solutions accordingly. The potential issues with multi-core systems are also highlighted.

Keywords: Multi-core processors, multiprocessing, cache coherence, memory bandwidth, scheduling.

1 Introduction

Multi-core technology is usually the term used to describe two or more CPUs or cores working together on the same chip where a single physical processor contains the core logic of two or more processors. These processors are packaged side-by-side on a same die in a single integrated circuit (IC) but the mounting of the chip is same as in the traditional CPU. In comparison to multiprocessing, multi-core is efficient and supports development of different types of software at a standardized platform. The multi-core processor technology was conceptualized around the idea of being able to make parallel computing possible because it increases the performance, speed and efficiency of computers. Having multiple cores on a single chip minimizes the power and heat consumption of the system while still being able to greatly boost overall system performance. The amount of performance gained by the use of a multi-core processor depends on the problem being solved, the algorithms used and their implementation in software (Amdahl's law) [17]. A dual-core processor with two cores at 2GHz may perform near to a single core of 4GHz but multiple cores provide increased computational capability on a single chip without requiring a complex architecture. As a result, simple multi-core processors have better performance than complex single-core processors [1].

Dual-core processors provide two complete execution cores, each with an independent interface to the front side bus (FSB) and because of separate caches for each core, the operating system has enough resources to handle different tasks in parallel, thus providing remarkable improvement in performance. Cores in a multi-core device may share a single coherent cache at the highest on-device cache level e.g. L2 for the Intel Core 2 or may have separate caches e.g. current AMD dual-core processors like AMD Athlon 64X2. Since more computing workloads could be done on parallel therefore manufacturers such as Intel and AMD are trying to focus more on computing and processing performance without increasing clock speed [3], [4].

2 Multi-core Architecture

Like any other technology, multi-core architectures of different manufacturers vary from each other. The number of cores, memory configuration and sharing of cache may differ. The design shown in Fig 1 is not specific to any particular multi-core design, rather it is a generic design of multi-core architecture and is not specific to any vendor. Although manufacturers' designs differ, still there are some integral parts that are included in every multi-core architecture.

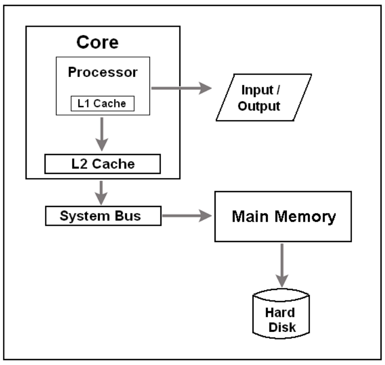


Fig. 1 Simple core architecture [18]

The processors work by reading a stream of instructions and determine if the data is in memory and the operations to do with it. The conventional processors used to take one instruction from memory and execute it completely before starting the next one. But the current processors take blocks of hundreds or thousands of instructions into cache before executing them. They can execute four or more instructions in parallel at a time [3].

Level 1 or L1 cache resides closest to the processor. It is a very fast memory to store data frequently used by the processor. Level 2 or L2 cache, though resides outside the processor, and is just slower than L1 cache but still much faster than main memory. Compared to L1 and L2 caches, main memory is very large in size. Most existing systems have memory of 1GB to 16GB while comparing cache is approximately 32KB (L1) and 2MB (L2) [8]. When data is not located in cache or main memory the system retrieves it from the hard disk which is the slowest memory system. A single communication bus called system bus is used for communication between cores and the main memory.

3 Current Multi-core Architectures and their Comparison

Intel and AMD are the most popular microprocessor manufacturers. Intel produces many kinds of multi-core processors like the Pentium D used in desktop, Core 2 Duo used in both laptop and desktop environments, and the Xeon processor used in servers [3]. AMD has the Althon lineup for desktops, Turion for laptops, and Opteron for servers/workstations [4]. Although Core 2 Duo and Athlon 64 X2 run on the same platforms, their architectures differ significantly. Therefore, we have tried to analyze and compare their architectural designs and how these two architectures try to solve the current multi-core architectural challenges.

3.1 Intel's Core Architecture

In Intel's core architecture, each core operates entirely independently of the other. They share their Level 2 cache and because of this, Intel has optimized operations for the simultaneous use of both cores. This means that when both cores are operating on the same area of memory, just one copy of the data is needed in the cache. This increases efficiency and allocates more cache for other processes. The cache also dynamically allocates its space that is to be used by each core, so that if one core is idle then more cache is available for the other core [8].

Intel's core is a pipelined architecture where instructions move through a number of internal stages between entering the processor and being completed as shown in Fig 2 After an instruction exits a stage, another one can enter the pipeline. A core has around fourteen stages in its pipeline but there are a number of complications, such as early completion and out of order execution, which makes it hard to analyze exactly how many stages are in the pipeline [2], [6].

The Intel's Core 2 duo has advantage of faster core-to-core communication and dynamic cache sharing between cores. Due to faster core-to-core communication, Intel is considered to be the best for the multi-threaded applications that have high amount of data sharing or communication. But in situations where two cores have different demands for cache then it is a real challenge for them to manage cache resources efficiently [1], [3].

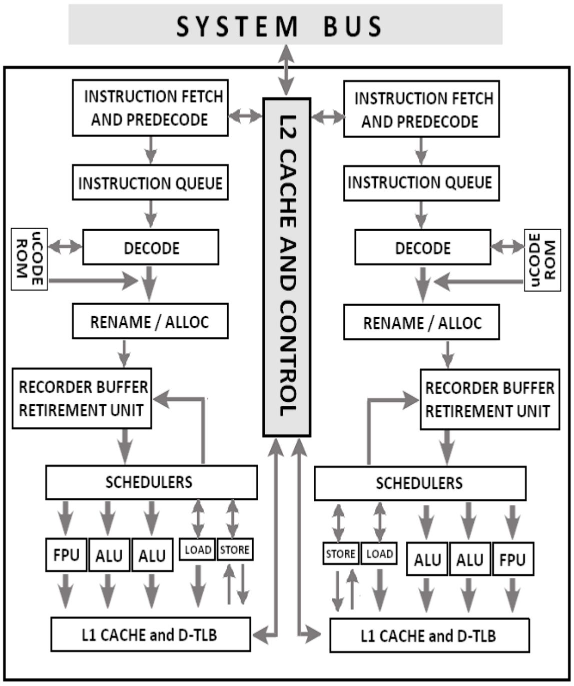


Fig. 2 Intel's Core architecture [3]

3.2 AMD's Core Architecture

In 2003, AMD launched the world's first 32-bit and 64-bit processor compatible with the x86 architecture. The AMD processors eliminate the bottlenecks caused by the front-side bus by directly connecting the processor, the memory controller and the I/O to the central processor unit to enable improved overall system performance and efficiency. We are going to describe AMD Athlon 64X2 processor which is designed specifically for multi-core architectures. AMD athlon has private L2 Caches as shown in Fig 3. Both of these L2 caches share a system request interface, which connects with an on-die memory controller and a HyperTransport. The system request interface serves as an interconnection between the two cores and does not require an external bus. The HyperTransport removes system bottlenecks by reducing the number of buses required. It provides more bandwidth than the current PCI technology. Recent AMD processors are based on K8 architecture. Some examples are AMD Opteron, Athlon, Turion and Sempron [4], [5].

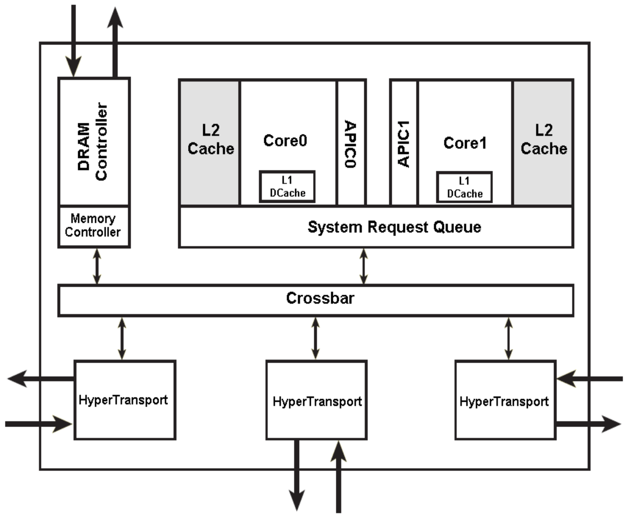


Fig. 3 AMD Multi-core architecture [4]

3.3 Comparison of Architectures

Fig 4 shows block diagrams for the Core 2 Duo and Athlon 64 X2, respectively. The figure is self explanatory; both architectures are homogenous dual-core processors. Intel's Core 2 Duo contains a shared memory model with private L1 caches and a shared L2 cache. When a L1 cache miss occur, both the L2 cache and the second core's L1 cache are traversed in parallel before sending a request to main memory. In contrast, the Athlon follows a distributed memory model with discrete L2 caches. Instead of a bus, the L2 cache shares a system request interface [7], [8].

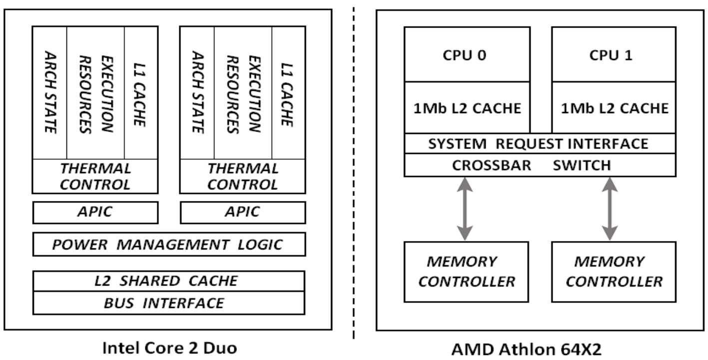


Fig. 4 Comparison of Intel and AMD architectures [7]

4 Multi-core: Promises and Challenges

Besides performance increase, multi-core processors also provide greater system density and overall productivity. Multi-core processors use less power and generate less heat per core than the same number of single-core processors because they operate at lower frequencies. Design innovations in multi-core processor architectures bring new optimization opportunities and challenges for the system software. Addressing these challenges will further enhance system performance.

4.1 Memory Bandwidth Problem

Memory latency has been one of the most important performance bottlenecks in single-core processors and nowadays on multi-core processors [9]. In Intel's architecture, all cores share the same front side bus (FSB). All the data that is written or read from the memory travels through the memory bus. Intel's FSB allows up to 8 devices to reside on the bus. As multiple cores are sharing the FSB therefore the bandwidth will be divided among all the cores which is a bottleneck for the performance because each core retrieve its instructions from the main memory. Since multiple cores are most efficiently used when each of the cores is executing one thread therefore it is possible for one memory-intensive job to saturate the shared memory bus thus affecting the overall performance of all the jobs running on that processor. AMD architecture provides its solution by having its own memory controller. However, because of physically distributing the main memory, some other issues arise like memory mapping and cache coherence [4].

4.2 Cache Problems

In current Multi-core architectures, Intel is using a shared L2 cache while the recent AMD architecture is using individual L2 cache for each core. Using high speed shared L2 cache provide important advantages for the processors like increased utilization of cache space and faster inter-core communication. A potential disadvantage of using shared L2 caches is that the heterogeneous data access patterns of memory-intensive tasks running on the cores sharing caches can lead to cache contention and thus producing non optimal performance. The cache contention depends on the shared resources, number of active tasks, and the access sequences of the individual tasks. Intel's Core 2 Duo tries to speed up cache coherence by being able to query the second core's L1 cache and the shared L2 cache simultaneously but the shared L2 cache in the Intel's Core 2 Duo processor removes on-chip L2-level cache coherence [10], [11].

The distributed L1 and L2 cache may also cause some problems. Since each core has its own cache, the copy of the data in that cache may not always be the most up-to-date version. e.g. If one core writes a value to a specific location, then if meanwhile the second core attempts to read that value from its cache it would not have the updated copy because its cache entry is invalidated and a cache miss occurs. AMD's

Athlon 64X2 has to monitor cache coherence in both L1 and L2 caches. AMD uses a HyperTransport interconnect technology for faster inter-chip communication to maintain the cache coherence between the two cores. In addition, the Athlon 64X2 has an on-die memory controller to reduce memory access latency [12], [13], [14].

5 Scheduling in Multi-core systems

In multi-core processors, each processor package has two or more execution cores, with each core having its own registers, execution units, caches, etc. The rapid changes of the processor architecture bring new opportunities and challenges to the system software. In order to get the optimal performance, different components like the task scheduler need to be aware of the multi-core architecture and the task characteristics.

The process scheduler manages CPU resource allocation to tasks and is an integral part of Operating System. The process scheduler typically deals for maximizing system throughput by running multiple tasks at a time and ensuring fairness among the running tasks in the system. It can be assumed that the conventional scheduling process on multi-core processors will be the same but due to the shared resources such as cache hierarchy and memory bandwidth between the cores suggest that the scheduling need to be done in a way to get maximum efficiency. By just allocating a fair amount of CPU time to each task by the process scheduler will not necessarily result in efficient and fair usage of the shared resources. Therefore, this is a challenging situation for the process scheduler because it has to identify and predict the resource needs of each task and schedule them in a way that will minimize the shared resource contention, maximize the overall utilization and therefore getting the advantage of shared resources between cores [15], [16].

6 Conclusion

Multi-core processor is an important addition in the microprocessor timeline and a remedy to high power consumption, price and size issues. The architecture of Intel and AMD, the two most important vendors of modern multi-core systems is highlighted and comparison is made from architectural view point. Similarly, the challenges ahead are discussed from caches, memory bandwidth and scheduling perspective.

7 References

[1] F. Pollack, "New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies," in Micro 32, 1999. http://intel.com/research/mrl/Library/micro32Keynote.pdf.

[2] R. Merritt, "CPU Designers Debate Multi-core Future," EETimes Online, February2008, http://www.eetimes.com/showArticle.jhtml?articleID=206105179.

[3] Intel® Core™2 Duo Desktop Processor and Intel® Pentium® Dual Core Processor at http://www.intel.com/products/processor/core2duo/index.htm

[4] AMD Athlon™ Processor, AMD Opteron™ Processor and AMD Turion™ 64 X2 Dual-Core at http://www.amd.com/us-en/Processors/

[5] AMD, AMD HyperTransport Technology at http://www.amd.com/us-en/Processors/DevelopWithAMD/0,,30_2252_2353,00.html

[6] B. Brey, "The Intel Microprocessors," Sixth Edition, Prentice Hall, 2003

[7] Lu Peng, Jih-Kwon Peir, Tribuvan K. Prakash, Yen-Kuang Chen and David Koppelman, "Memory Performance and Scalability of Intel's and AMD's Dual-Core Processors: A Case Study," IEEE, 2007.

[8] J. Dowdeck, "Inside Intel Core Microarchitecture and Smart Memory Access," Intel, 2006, http://download.intel.com/technology/architecture/sma.pdf

[9] Kunle Olukotun, Basem A Nayfeh, Lance Hammond, Ken Wilson and Kunyung Chang, "The case for a single-chip multiprocessor. Architectural Support for Programming Languages and Operating Systems (ASPLOS)," 2006.

[10] J. Chang, S. Member, and M. Huang, "The 65-nm 16-mb shared on-die l3 cache for the dual-core intel xeon processor 7100 series," Journal of Solid-State Circuits, 42(4), 2007.

[11] L. Peng, J. Peir, T. K. Prakash, Y. Chen, and D. Koppelman, "Memory performance and scalability of intel's and amd's dual-core processors: A case study," in International Performance, Computing, and Communications Conference (IPCCC), 2007.

[12] E. Suh, L. Rudolph, and S. Devadas, "Dynamic partitioning of shared cache memory," The Journal of Supercomputing, vol. 28, no. 1, April 2004, pp. 7–26.

[13] R. Iyer, "CQoS: A framework for enabling QoS in shared caches of CMP platforms," in ICS, 2004.

[14] S. Kim, D. Chandra, and Y. Solihin, "Fair cache sharing and partitioning in a chip multiprocessor architecture," in PACT, 2004.

[15] M. Rajagopalan, B. T. Lewis and T. A. Anderson: "Thread Scheduling for Multi-Core Platforms," in proceedings of the Eleventh Workshop on Hot Topics in Operating Systems, 2007.

[16] Shiyuan Jin, Guy Schiavone and Damla Turgut, "A performance study of multiprocessor task scheduling algorithms," The Journal of Supercomputing, Volume 43, Issue 1, January 2008 pp. 77 - 97.

[17] G. Amdahl, "Validity of the single processor approach to achieveing large scale computing capability," in Proceedings of AFIPS Spring Joint Computer Conference, 1967, pp. 483-485.

[18] Intel, Inside Intel Core Microarchitecture and Smart Memory Access http://download.intel.com/technology/architecture/sma.pdf.

Cybernetics and Causality

Stanislas Bigirimana

Department of Philosophy
Heidelberg University, Germany

Abstract. Information has often been defined in terms of messages and signals. However, Norbert Wiener definition of information as negative entropy introduces a new model of the universe (cybernetics) that defines the universe in terms of material, mechanical, logical, and symbolic characteristics, unlike Newton's mechanics that defined the universe in terms of material bodies in motion. Following this model, there is a possibility of introducing a theory of causality that integrates material, efficient, formal and final causes without necessary leaning back to the Aristotelian teleology that considers the final cause as only the ultimate "Unovable Mover".

1 Introduction

Wiener has elevated the concept of information at the metaphysical level through his theory of cybernetics. While Newton's mechanics defined reality in terms of matter and motion, cybernetics defines reality in terms of information and control through feedback Shifting from mechanics (matter + motion) to cybernetics (matter + motion + logic + symbolism) implies shifting from an energy based model to an information based model. In historical terms, while The steam engine moved us from an era of animation, where motion and action were described in terms of the soul or "anima" in Latin, to an era of automation, likewise we can postulate that the computer revolution introduced the era of programming. Therefore, understanding reality in terms of the integration material and mechanical processes (hardware) with logical and symbolic processes (software) creates a new understanding of causality. There is a need to go beyond Descartes' reduction reality to hardware and causality to its material and efficient dimensions.

Is paradoxical that many Cartesians seem not to be aware that the universal nature of mathematical and logical formulation implies context free (i.e. universal) symbols which are the best exemplifiers of formal causality. Formal causality cannot be expressed in the Aristotelian comparison of formal causality to the idea a sculptor would have of the statue that he or she wants to make, but to the emerging of universal standards and tools such as Universal Turing Machines and computers programs such as the General Problem Solver. These tools are the best embodiment of formal causality because they work with algorithms and algorithms as formal rules that do not depend on the contents of strings of data that stored computer programs process, exemplify in our context, the role of the ideal form of a process. This exemplification amounts to formal causality because the input and the output of the process do not depend on the hardware involved. For instance, Simon when he pointed out that:

The program, the organization of the symbol-manipulating processes, is what determines the transformation of input into output. In fact, provided with only the program output, and without information about the processing speed, one cannot determine what kinds of physical devices accomplished the transformations: whether the program was executed by a solid state-computer, an electron-tube device, and electric relay machine, or a room full of clerks! It is only the organization of the process that is determinate [1].

In fact, Cartesian reductionism originated in the shortening of a fourfold traditional neo-Aristotelian model of reality in order to make it fit within the mechanical model. As Cooney has pointed out:

1. inanimate bodies – objects such as rocks, soil, and even liquids and gases
2. plants- living bodies such as vegetables and even mosses
3. animals – sentient living bodies, including birds, fish, and land animals,
4. humans – sentient living bodies of a specific from with distinctive capabilities such as reason and speech [2].

The cumulative nature of this tetrad resides in the fact that: “Each higher level, fromhe first to the fourth, includes the lower with the addition of a further attribute” [3]. Descartes reductionism resides actually in reducing the second and third levels of the tetrad to the first on the ground that “plants and animals were to be understood merely as machines, complex systems of movable parts that enable them to behave adaptively in their environment.”[4] Therefore,

> His (Descartes) reduction of life and sentience to the motion of material particles created an abyss between the human mind and body. It has the effect of suspending our consciousness in a world of bodies with which it has nothing in common. Each human mind became a ghost in its own machine [5].

We can study material bodies in their constitutive dimensions by identifying some scalar measurement of their characteristics such as the weight or their dimensions in space (length, width, height). However, when these bodies are set in motion, they can no longer be studied solely as matter in a three dimensional space. Setting a material body in motion brings a change in spatial location that introduces time as a fourth dimension. Studying a body in motion includes studying not only characteristics that lend themselves to scalar measurement but also those that are best represented by vectors. Setting a material body in motion introduces other dimensions such as the direction and the sense of the motion, its intensity and its origin. In elementary physics, the study of the notion of force for example follows this pattern.

The same applies when one introduces a criteria of order in the relationship between material bodies, be they stationary or in motion. Order is not an intrinsic characteristic of the bodies since it has to be characterized through the capacity to identify its criteria and to examine whether a given situation meets these criteria. There are many ways of proceeding, but checking whether a certain situation meets a certain criterion calls for an element of human judgment that goes beyond the impersonal nature of mechanical processes. Order then calls for principles of logic and rationality, which were once understood as organizing principles, that amount to negative entropy and can take the form of information, structure and/or pattern. Orderly processes, patterns or structures can also be matched with human goals. While logic makes the processes, structures and patterns meaningful through understanding, goals or purposes make the same processes valuable and in addition to asking the “how question” normally linked with efficient causality we are in a position to ask the why question linked with teleology and hence with final causality. This theory of causality is due to the fact that organized processes convey information because by definition organization means order and negative entropy. Organized processes then, for the sake of being organized, do not occur randomly. They are purposeful. This idea of purpose implies that organized processes always have a goal. Therefore, it is important to take into account these goals when one is studying an organized process. Reintroducing the idea of teleology in the study of causality implies that our goals play the role of an inviting final cause. This existence of inviting final causes upsets the assumption of normative epistemology that causality is linear and that chronologically the causes always precede the effects. In this context, the idea of a final cause should not be understood in an ultimate and absolute sense as it has been associated with the Aristotelian “unmoved mover” and God in the Middle Ages. It should be understood in a more mundane sense as a goal that is not out of human reach. This type of goal has to be SMART i.e. Specific, Measurable, Achievable, Realizable, and Timely.

It is worth noting that the fact that information processing operates on symbols and symbol structures which can functionally be represented regardless of their underlying hardware does not imply that hardware should be done away with. It is important to take into account the role of physical symbolic systems both in the structuring and the knowing of reality. As Simon has noted:

> A physical symbol is a pattern (of chalk, ink, neural connections, electromagnetic fields, or what not) that refers to or designates another pattern or a detectable external stimulus. Printed words on a page are symbols, so are pictures and diagrams, so are numbers. A physical symbol system is a system that is capable of inputting symbols, outputting them, storing them in memory, forming and modifying structures of symbols in memory, comparing pairs of symbols for identity or difference, and branching in its subsequent behavior of the outcomes of such tests [6].

In fact:

> A computer is obviously a physical symbol system. Its ability to perform these processes (and only these processes) can be verified easily from its physical properties and operation. A human brain is (less obviously) a physical symbol system. It can certainly carry out the processes specified in the definition of such as system, but perhaps other processes as well [7].

In brief, while the mechanical model reduces causality to material and efficient causes, cybernetics provides a possibility for a fourfold theory that includes material, efficient, formal and final causes.

References

1. Herbert A. Simon, Economics, Bounded Rationality, and the Cognitive Revolution (Brookfield, Vermont: Edward Elgar, 1992), p. 76.
2. Brian Cooney, Posthumanity: Thinking Philosophically about the Future (Lanham: Rowman & Littlefield, 2004), p. 74.
3. Ibid.
4. Ibid.
5. Ibid.
6. Simon , Economics, Bounded Rationality and the Cognitive Revolution, p. 106. Emphasis in the original.
7. Ibid.

The 32nd Annual Conference on
Artificial Intelligence (KI-2009)
September 15-18, 2009, Paderborn, Germany

Workshop on

Emotion and Computing - Current Research and Future Impact

http://www.emotion-and-computing.de

Dirk Reichardt (Ed.)

KI 2009

AI And Automation

Workshop Organizers and Co-Chairs

Dirk Reichardt University of Cooperative Education Stuttgart, Germany
reichardt@ba-stuttgart.de

Program Committee

Joscha Bach	University of Osnabrück, Germany
Christian Becker-Asano	ATR, Kyoto, Japan
Nicola Henze	University of Hannover, Germany
Manfred Kerber	University of Birmingham, UK
Michael Kipp	DFKI Saarbrücken, Germany
Paul Levi	University of Stuttgart, Germany
John-Jules Charles Meyer	University of Utrecht, Netherlands
Christian Peter	Fraunhofer IGD, Rostock, Germany
Götz Renner	Daimler AG, Customer Research Center, Stuttgart, Germany
Michael M. Richter	University of Calgary, Canada

Preface

Dirk Reichardt

Applied Computer Science and Information Technology
Baden-Württemberg Cooperative State University Stuttgart (DHBW)
Stuttgart, Germany
reichardt@dhbw-stuttgart.de

The workshop series “emotion and computing - current research and future impact” has been providing a platform for discussion of emotion related topics of computer science and AI since 2006. In recent years computer science research has shown increasing efforts in the field of software agents which incorporate emotion.

Several approaches have been made concerning emotion recognition, emotion modelling, generation of emotional user interfaces and dialogue systems as well as anthropomorphic communication agents. Motivations for emotional computing are manifold. From a scientific point of view, emotions play an essential role in decision making, as well as in perception and learning. Furthermore, emotions influence rational thinking and therefore should be part of rational agents as proposed by artificial intelligence research. Another focus is on human computer interfaces which include believable animations of interface agents. From a user perspective, emotional interfaces can significantly increase motivation and engagement which is of high relevance to the games and e-learning industry.

This workshop intends to discuss the scientific methods considering their benefit for current and future applications. Especially when regarding the subject of emotion recognition, this also includes ethical aspects.

This year we are proud to have 6 presentations of ongoing research work and a life demo within our workshop – which is a rather high number regarding the IVA and ACII Conference at Amsterdam at the same time. It has become a tradition to select a discussion topic and integrate an open discussion session on this. Last year we started to integrate web based discussions by providing a discussion (mind mapping) feature on the workshop website http://www.emotion-and-computing.de. We are looking forward to interesting presentations and fruitful discussions.

Dirk Reichardt

The happiness cube paradigm; eliciting happiness through sound, video, light and odor. Assessment of affective state with non-invasive techniques.

Stylianos Giannoulis[1], Dr. Fons Verbeek[1]

[1]Media Technology. Leiden Institute of Advanced Computer Science, Leiden University, Niels Bohrweg 1, 2333 CA, Leiden, the Netherlands
s.giannoulis@umail.leidenuniv.nl, fverbeek@liacs.nl

Emotion elicitation and physiological responses are 2 fields that have been studied extensively the last 20 years. Based on the already existing research, a scientific experiment is described with the goal to elicit emotions of happiness to the participants by the use of video, sound and odorants. Contrary to most already existing research, the goal of this experiment is to elicit just one emotion -happiness-. Moreover, the expected multisensory experience is of great significance since most of the existing research on emotion elicitation is usually focusing only on one or two at most sensory modalities.

Dynamic Facial Expression Classification Based on Human Visual Cues Information

Zakia Hammal[1], Corentin Massot[2]

[1]Universite de Montréal, [2] McGill University
zakia_hammal@yahoo.fr

Compared to human performances automatic recognition of facial expressions in real life situations remains a big challenge for computer systems. Here, we aim at including new strategies into a previously developed model by Hammal *et al.* that has been successfully used for the recognition of the six basic facial expressions. It is based on the Transferable Belief Model (TBM). The TBM allows modeling explicitly the doubt and uncertainty of the facial features detection generating conclusions with confidence that reflects uncertainty of the detection and tracking methods.

In this paper we extend this model by, first, introducing the most important transient facial features used by human observers for facial expressions recognition (such as nasal root winkles and nasolabial furrows). We introduce a new method based on multiscale spatial filtering that estimates the appearance of transient facial features, provides an estimation of the confidence in this estimation and accurately measures the orientation of the detected wrinkles when necessary. The proposed method reaches a precision of 89%.

Second, we introduce a dynamic and progressive fusion process of the permanent facial features (such as eyes, eyebrows and mouth) and of the transient facial features (such as nasal root wrinkles and nasolabial furrows), which allows dealing with asynchronous facial features deformations. Moreover, based on the dynamic TBM fusion process, the proposed model allows the dynamic recognition of pure facial expressions (Happiness, Surprise, Fear, Disgust, Sadness, Anger) and Neutral and explicitly models the doubt between expressions in the case of blends, combinations or uncertainty between two or several expressions.

The classification results were performed on all the six basic facial expression and on three benchmark databases: Cohn-Kanade, STOIC (a total of 182 videos) and CAFE. Compared to the previous model, the introduction of the temporal modeling of all the facial features led to an increase of performances of 12% in average. In order to compare our results with human performances, we validated part of the used videos by human observers. Performances obtained by the proposed model and by humans compared favorably.

Besides the use of the most important human visual cues for facial expressions classification future work will introduce another possible human strategy consisting in weighting each visual cue according to its importance for the expected facial expression.

Classifying Facial Pain Expressions
Individual Classifiers vs. Global Classifiers

Michael Siebers[1], Miriam Kunz[2], Stefan Lautenbacher[2], and Ute Schmid[1]

[1] Faculty Information Systems and Applied Computer Science, University of Bamberg, Germany
[2] Physiological Psychology, University of Bamberg, Germany

In this study we show which classifiers are suitable to predict pain by facial features. Furthermore we investigate the need to individually train a classifier for each subject. Additionally the presented work explorativly surveys data preprocessing, feature selection and generation.

As data we use stills extracted from pain videos. Those videos were taken from a previously conducted psychophysiological study. They showed a total of five subjects undergoing painful and not-painful pressure stimuli.

On the stills 58 reference points are manually marked. Using this point data we calculate relational measures as for example distances, angles or ratios. Additionally we add comparisons of these relational measures with their individual and global average regarding non-pain images. We use this data (point data, relational measures and comparisons) to train and test the classifiers.

We compare the performances of *Decision Trees*, *Naive Bayes*, *Support Vector Machines*, *k-Nearest Neighbours*, *Perceptrons*, *Neural Networks* and *Classification by Regression*. As first step we optimize the classifiers parameters mainly using a grid searching approach. After this we use 10-folded cross-validations to estimate the classifiers performances. For some classifiers attributes we weight (respectively select) features before optimizing parameters. For each classifier this is done once globally and once for each subject.

We show that most classifiers are suitable for the given task. The best performances are achieved by *Support Vector Machines* and *k-nearest Neighbours*. In most cases individually trained classifiers perform better than globally trained.

Towards Facial Mimicry for a Virtual Human

Hana Boukricha and Ipke Wachsmuth

Faculty of Technology
Bielefeld University
33594 Bielefeld, Germany
{hboukric|ipke}@techfak.uni-bielefeld.de

Mimicking others' facial expressions is believed to be important in making virtual humans as more natural and believable. As result of an empirical study conducted with a virtual human a large face repertoire of about 6000 faces arranged in Pleasure Arousal Dominance (PAD-) space with respect to two dominance values (dominant vs. submissive) was obtained. Each face in the face repertoire consists of different intensities of the virtual human's facial muscle actions called Action Units (AUs), modeled following the Facial Action Coding System (FACS). Using this face repertoire an approach towards realizing facial mimicry for a virtual human is topic of this paper. A preliminary evaluation of this first approach is realized with the basic emotions Happy and Angry.

The OCC Model Revisited

Bas R. Steunebrink, Mehdi Dastani, and John-Jules Ch. Meyer

Department of Information and Computing Sciences,
Utrecht University, The Netherlands
bass,mehdi,jj@cs.uu.nl

Although popular among computer scientists, the OCC model of emotions contains a number of ambiguities that stand in the way of a truthful formalization or implementation. This paper aims to identify and clarify several of these ambiguities. Furthermore, a new inheritance-based view of the logical structure of emotions of the OCC model is proposed and discussed.

For Androids, Emotions are just in Time.

Redefining the Need for Mentality, Process Structures and Situation Representation in order to develop real interactive Possibilities for Androids

Wolfgang Gessner
FHNW
gessner@collegium.ethz.ch, wolfgang.gessner@fhnw.ch

Emotions as primarily intersubjective processes have to be analyzed along the dimension in which they unfold: Time. So process philosophy will be the best fitting approach to get a grasp of the development of these special mental events and their individual and social functions. The analysis and construction of algorithms proves to be the appropriate methodology to describe the prerequisites for the elicitation of a single emotion, as well as its dependent modular dimensions (mimics, gestures, action, speech). Already single emotions (and the more sequences of different emotions or emotions mutually reacting on the other's emotion) are time-dependent series of mental events, which are, as processes, embedded in changing situations (i.e., subjectively represented objective events). The reconstruction of human emotions along these lines will open new perspectives for the construction of android emotions which more than just imitate the surface structures of stable pictures of human emotions. Robot emotions have to be appropriate to the generation of human emotions in order to achieve a parallelized functionality which can give rise to the possibility of interaction, communication and cooperation between androids and men. So the task will be twofold: (i) construction of an android mental system for representing situations and coping with them by emotions, and (ii) testing these new abilities in model worlds which are manageable and calculable in order to solve the 'hard problem' of situation representation for androids.

Emotion Recognition Using Optical Flow

Michael Girgis[1], Dirk Reichardt[2]

1
[1]German University Cairo (GUC), Egypt
micheal.girgis@student.guc.edu.eg
[2]Duale Hochschule Baden-Württemberg Stuttgart, Germany
reichardt@dhbw-stuttgart.de

Emotion recognition in faces is based either on features or movements of muscles. In our experiment we applied optical flow on video sequences and determined typical muscle movements in order to detect Ekman's Action Units. The classifier for the eye region is trained by a genetic algorithm and results are tested against the Cohn-Kanade Database. The first experiments with eyebrow movement detection are promising.

The experiment is part of a emotion recognition framework which developed at the DHBW Stuttgart. The framework uses the OpenCV library and is implemented in C++. The head detection and tracking as well as the positioning of so called regions of interest is provided by the framework. In our experiment additional ROI-boxes are placed on the eyebrow region. In this region we place elements to track for an optical flow analysis. The region is divided in three vertical sections. The flow direction and strength is then calculated for each section. To compensate head motion, the flow values of reference points outside the eyebrow region are measured to compute relative motion. The classification of action units [2] AU1, AU2 and AU4 is done by a finite state machine. The state transition thresholds are determined by a genetic algorithm which uses 20 labeled samples from the Cohn-Kanade database [1] as a reference for the fitness function. The approach works quite nicely in sequences with low head motion already. Results and life testing can be shown at the workshop.

References

1. Kanade, T., Cohn, J. F., & Tian, Y. (2000). Comprehensive database for facial expression analysis. Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition (FG'00), Grenoble, France, 46-53.
2. Paul Ekman, Wallace V. Friesen: Facial action coding system: A technique for the measurement of facial movement. Palo Alto, Calif.: Consulting Psychologists Press, 1978

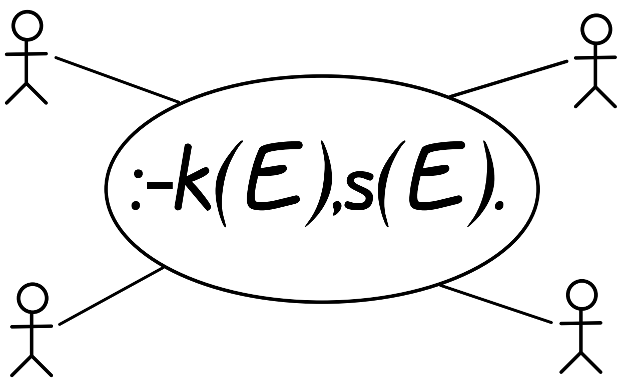
The 32nd Annual Conference on
Artificial Intelligence (KI-2009)
September 15-18, 2009, Paderborn, Germany

Workshop on

Knowledge Engineering and - Software Engineering (KESE 2009)

https://ai.ia.agh.edu.pl/wiki/kese:kese2009

Grzegorz J. Nalepa & Joachim Baumeister (Eds.)



Workshop Organizers and Co-Chairs

Grzegorz J. Nalepa	AGH UST Kraków, Poland gjn@agh.edu.pl
Joachim Baumeister	University Würzburg, Germany joba@uni-wuerzburg.de

Program Committee

Klaus-Dieter Althoff	University Hildesheim, Germany
Joachim Baumeister	University Würzburg, Germany
Stamatia Bibi	Aristotle University of Thessaloniki, Greece
Joaquín Cañadas	University of Almeria, Spain
Ulrich Geske	FhG FIRST, Berlin, Germany
Adrian Giurca	BTU Cottbus, Germany
Rainer Knauf	TU Ilmenau, Germany
Grzegorz J. Nalepa	AGH UST, Kraków, Poland
Frank Puppe	University Würzburg, Germany
Jochen Reutelshoefer	University Würzburg, Germany (external reviewer)
Dietmar Seipel	University Würzburg, Germany
Ioannis Stamelos	Aristotle University of Thessaloniki, Greece
Gerhard Weiss	SCCH, Austria

Preface

Joachim Baumeister and Grzegorz J. Nalepa

Intelligent Systems (Informatik 6)
University of Würzburg
Würzburg, Germany
joba@uni-wuerzburg.de
—
AGH University of Science and Technology
Kraków, Poland
gjn@agh.edu.pl

Intelligent systems have been successfully developed in various domains based on techniques and tools from the fields of knowledge engineering and software engineering. Thus, declarative software engineering techniques have been established in many areas, such as knowledge systems, logic programming, constraint programming, and lately in the context of the Semantic Web and business rules.

The fifth workshop on Knowledge Engineering and Software Engineering (KESE 2009) was held at the KI-2009 in Paderborn, Germany, and brought together researchers and practitioners from both fields of software engineering and artificial intelligence. The intention was to give ample space for exchanging latest research results as well as knowledge about practical experience. Topics of interest includes but were not limited to:

- Knowledge and software engineering for the Semantic Web
- Ontologies in practical knowledge and software engineering
- Business rules design and management
- Knowledge representation, reasoning and management
- Practical knowledge representation and discovery techniques in software engineering
- Agent-oriented software engineering
- Database and knowledge base management in AI systems
- Evaluation and verification of intelligent systems
- Practical tools for intelligent systems
- Process models in AI applications
- Declarative, logic-based approaches
- Constraint programming approaches

This year, we mainly received contributions focussing on the "intelligent web": Pascalau and Giurca introduce a rule engine for web browsers, that is capable to handle DOM (Document Object Model) events within the browser. Cañadas et al. describe an approach for the automatic generation of rule-based web applications, that is based on ideas of the Model Driven Development (MDD). Nalepa and Furmańska propose an ontology that maps the design process of an intelligent application and thus promises efficient development. Reutelshoefer et al. show, how multimodal knowledge appears in knowledge engineering projects and show how such knowledge can be refactored within a Semantic

Wiki. The intelligibility of medical ontological terms is discussed and evaluated by Forcher et al. This year we also encouraged to submit tool presentations, i.e., system descriptions that clearly show the interaction between knowledge engineering and software engineering research and practice. At the workshop, two presentations about current tools were given: Kaczor and Nalepa introduced the toolset HaDEs, i.e., the design environment of the HeKatE methodology. Pascalau and Giurca show-cased the JavaScript rule engine JSON Rules.

This proceedings contain the abstracts of the papers presented at the workshop. The full versions of the presented papers were also published as CEUR proceedings Vol-486 (http://ceur-ws.org/Vol-486).

The organizers would like to thank all who contributed to the success of the workshop. We thank all authors for submitting papers to the workshop, and we thank the members of the program committee as well as the external reviewers for reviewing and collaboratively discussing the submissions. For the submission and reviewing process we used the EasyChair system, for which the organizers would like to thank Andrei Voronkov, the developer of the system. Last but not least, we would like to thank Klaus-Dieter Althoff (U. Hildesheim) as the workshop chair and Bärbel Mertsching (U. Paderborn) as the KI09 conference chair for their efforts and support.

Joachim Baumeister
Grzegorz J. Nalepa

A Lightweight Architecture of an ECA Rule Engine for Web Browsers

Emilian Pascalau[1] and Adrian Giurca[2]

[1]Hasso Plattner Institute, Germany,
emilian.pascalau@hpi.uni-potsdam.de
[2]Brandenburg University of Technology, Germany,
giurca@tu-cottbus.de

Abstract. There is a large literature concerning rule engines (forward chaining or backward chaining). During the last thirty years there were various proposals such as RETE, TREAT and the derived Gator algorithm. Significantly, RETE was embedded into various expert systems such as Clips and its successor Jess, and Drools including in a number of commercial rule engines. RETE was extended various times including with support for ECA rules. However, none of them is able to directly process DOM Events. The goal of this paper is to present the architecture of a forward chaining Event-Condition-Action (ECA) rule engine capable to handle Document-Object-Model Events.
The main goals that this architecture should address are: (1) to move the reasoning process to the client-side resulting in reduced network traffic and faster response; (2) to handle complex business workflows; (3) information can be fetched and displayed in anticipation of the user response; (4) pages can be incrementally updated in response to the user input, including the usage of cached data; (5) to offer support for intelligent user interfaces; (6) enable users to collaborate and share information on the WWW through real-time communication channels (rule sharing and interchange).
The Document Object Model Events (DOM Events) ontology provides a large amount of events types designed with two main goals: (1) the design of an event system allowing registration of event listeners and describing event flow through a tree structure (the DOM), and (2) defining standard modules of events for user interface control and notifications of document mutation, including defined contextual information for each of these event modules. This ontology is already implemented into browsers giving extremely powerful capabilities to RIA's which use it.
This architecture strives to comply with the principles of a SaaS architecture (Bennett et al., 2000). As such the main capabilities considered are: (1) distributed architecture - components can act in different network locations; (2) event-driven architecture - both human agents and software agents interact with this architecture by creating events. It is a live system i.e. an event-based system that is reactive and proactive. It is reactive because it reacts based on the events it receives. It is proactive because by itself generates events, that can be consumed also by other entities being part of the whole system.

A Data Structure for the Refactoring of Multimodal Knowledge

Jochen Reutelshoefer, Joachim Baumeister, Frank Puppe

Institute for Computer Science, University of Würzburg, Germany
{reutelshoefer, baumeister, puppe}@informatik.uni-wuerzburg.de

Abstract. In today's knowledge engineering tasks knowledge of a project is often already available in various forms and formalisms distributed over multiple sources, for instance plain text, tables, flow-charts, bullet lists, or rules. We define this intermixture of different shapes of knowledge at different degrees of formalization as *multimodal knowledge*. However, current state–of–the–art tools are often constrained to a specific knowledge representation and acquisition interface for developing the knowledge base. In consequence, the tools are commonly not flexible to deal with multimodal knowledge. While the evolution of the knowledge system based on a single formalism (e.g., ontology evolution) has been thoroughly studied, the evolution of multimodal knowledge has not yet been sufficiently considered. Particularly, the ability for the efficient refactoring of multimodal knowledge cannot be provided by today's static knowledge engineering tools. In the paper, we propose a data structure and a number of methods, that support the representation and refactoring of multimodal knowledge. To achieve this, the given knowledge elements is transformed into a generic tree-based data structure, that we call *KDOM* (knowledge document object model). We discuss its characteristics and we present a (semi-parsing) algorithm building up the KDOM from the documents containing the knowledge. The algorithm uses a KDOM schema, which describes the expected syntactical patterns of a document. By customizing KDOM schemas, we aim to provide refactoring capabilities for arbitrary multimodal knowledge at low implementation costs. For demonstration purposes, we have implemented the described approach in the semantic wiki KnowWE. We claim that a semantic wiki represents an appropriate knowledge engineering tool to develop multimodal knowledge. In this paper, a selection of refactorings is discussed and its implementation by operations on the KDOM is sketched. We motivate the use of the refactoring methods *Rename Concept* and *Coarsen Value Range* by examples, and we describe their application in the context of the presented data structure. For future work, we propose to apply semi-automated processes either on the construction of the KDOM tree or on refactoring operations on less formalized parts of the knowledge (e.g., using NLP techniques). The presented work is related to other approaches in the disciplines of software engineering and ontology engineering.

Evaluating the Intelligibility of Medical Ontological Terms

Bjrn Forcher[1], Kinga Schumacher[1], Michael Sintek[1], and Thomas Roth-Berghofer[1,2]

[1] Knowledge Management Department, German Research Center for Artificial Intelligence (DFKI) GmbH Trippstadter Strae 122, 67663 Kaiserslautern, Germany

[2] Knowledge-Based Systems Group, Department of Computer Science, University of Kaiserslautern, P.O. Box 3049, 67653 Kaiserslautern

{firstname.lastname}@dfki.de

Abstract. The research project MEDICO aims at developing an intelligent, robust and scalable semantic search engine for medical documents. The search engine of the MEDICO demonstrator RadSem is based on formal ontologies and is designated for different kinds of users, such as medical doctors, medical IT professionals, patients, and policy makers. Since semantic search results are not always self-explanatory, explanations are necessary to support requirements of different user groups. For this reason, an explanation facility is integrated into RadSem employing the same ontologies for explanation generation. In this work, we present a user experiment that evaluates the intelligibility of labels provided by the used ontologies with respect to different user groups. We discuss the results for refining our current approach for explanation generation in order to provide understandable justifications of semantic search results. Here, we focus on medical experts and laymen, respectively, using semantic networks as form of depiction.[3]

Key words: justification, graphical explanation, semantic search, evaluation, medical terms

[3] This research work has been supported in part by the research program THESEUS in the MEDICO project, funded by the German Federal Ministry of Economics and Technology (grant number 01MQ07016). The responsibility for this publication lies with the authors.

HaDEs – Presentation of the HeKatE Design Environment

Krzysztof Kaczor[1] and Grzegorz J. Nalepa[1]

Institute of Automatics,
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
kk@agh.edu.pl, gjn@agh.edu.pl

Abstract. TOOL PRESENTATION: The *HeKatE* project (hekate.ia.agh.edu.pl) aims at providing an integrated methodology for the design, implementation, and analysis of rule-based systems. An important goal of the project is to allow for an easy integration of knowledge and software engineering methods and approaches, thus providing a *Hybrid Knowledge Engineering* methodology. The project delivers several new knowledge representation methods, and a set of practical tools supporting the design process. The HeKatE design process is supported by the design environment called HaDEs. During the presentation given at the KESE 2009 workshop the tools are presented using practical examples.
HeKatE introduces a formalized language for rule representation called XTT (*eXtended Tabular Trees*). The language uses expressions in the so-called ALSV logic. The main phase of the hierarchical XTT rule design process is called the *logical design*. It may be supported by a preceding *conceptual design* phase. In this phase the rule prototypes are built with the use of the so-called *Attribute Relationship Diagrams* (ARD). The practical implementation on the rule base is performed in the *physical design* phase, where the visual XTT model is transformed into an algebraic presentation syntax called HMR (*HeKatE Meta Representation*). Then a custom inference engine, HeaRT runs the XTT model.
The design process is supported by the HaDEs toolset providing the visual design and the automated implementation of rule-based systems. The ARD design is supported by the HJEd visual editor. Once created, the model can be saved in a XML-based HML (*HeKatE Markup Language*) file. VARDA is a prototype semivisual editor for the ARD diagrams implemented in Prolog, with an on-line model visualization with Graphviz. The tool also supports prototyping of the XTT model. HQEd provides a visual support for the logical design. It is able to import a HML file with the ARD model. It allows to edit the XTT structure with on-line support for syntax checking on the table level. HeKatE Run Time (HeaRT) is a dedicated inference engine for the XTT rule bases. It is implemented in Prolog in order to directly interpret the HMR representation generated by HQEd. HeaRT also provides a modularized verification framework, also known as HalVA (*HeKatE Verification and Analysis*). Several plugins are available, including completeness, determinism and redundancy checks.

Design Process Ontology – Approach Proposal

Grzegorz J. Nalepa[1] and Weronika T. Furmańska[1]

Institute of Automatics,
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
gjn@agh.edu.pl, wtf@agh.edu.pl

Abstract. Practical design support is important for intelligent systems. The knowledge acquisition phase and initial conceptual modeling often with help of human experts largely influences the quality of complex systems. In case of knowledge-based systems a hierarchical aspect of this process helps in the design.

ARD+ (*Attribute Relationship Diagrams*) method has been invented to support the conceptual design of the XTT^2-based rules. It is a rule prototyping method in the HeKatE project (`http://hekate.ia.agh.edu.pl`). However, number of limitations of the method have been identified.

In this paper a new approach to the conceptual design of rules is proposed. The main idea comes down to the use of the Semantic Web methods and tools to represent ARD+ design and overcome the limitations of the original method. In the approach proposed in this paper an OWL ontology capturing the semantics of the ARD+ model is proposed. Such an ontology (called *Design Process Ontology*) models the functional dependencies between rule attributes, as well as the history of the design process. At the same time it is more flexible than the original method, opening up possibility of integration with other modeling methods and tools, e.g. UML in the area of software engineering.

DPO is a proposal of a task ontology. Its aim is to capture the system characteristics together with dependencies among them, as well as represent the gradual refinement of the design process. DPO may be specialized by concrete ontologies for specific design tasks. In this case system characteristics (conceptual and physical attributes) subclass the general `Attribute` class. All the characteristics and attributes identified in a system are represented as independent classes. The properties may be specialized accordingly, so that they range over concrete system classes rather than the general `Attribute` class. Proposed ontology consists of concepts being system characteristics and attributes, and roles denoting the dependencies among them. The example ontology has been built in OWL with use of Protege editor and presented in the paper.

Future work concerns further investigation of the possibilities of using ontologies in the design process. This includes formalizing selected ARD+ model features in Description Logics. As the design process progresses, the DPO changes. The transformations between subsequent ontologies will be analyzed and their formalization will be proposed. Use of rules on top of the DPO is considered. These rules could be possibly introduced as DLP (DL Programs) or expressed in SWRL.

A Model-Driven Method for automatic generation of Rule-based Web Applications

Joaquín Cañadas[1], José Palma[2] and Samuel Túnez[1]

[1] Dept. of Languages and Computation. University of Almeria. Spain
jjcanada@ual.es, stunez@ual.es
[2] Dept. of Information and Communication Engineering.
University of Murcia. Spain
jtpalma@um.es

Abstract. In the Semantic Web context, rule languages and inference engines incorporate reasoning in Web information systems. This paper addresses the development of rule-based systems embedded in Web applications to provide Web systems with inference capabilities. The proposed approach is based on Model-Driven Development (MDD) to automate code generation of rule-based Web applications. Development process starts with the specification of a rule-based model that defines the domain expert knowledge and the business logic through a domain ontology and a set of production rules (if-condition-then-action). This model, created in Conceptual Modeling Language (CML), a formalism defined by the CommonKADS methodology, becomes the source model for the model-driven approach.

The proposed method is divided into two processes: (1) code generation of the rule base in Jess rule technology, and (2) the production of the code for the Web architecture, including configuration files, Java classes and Web pages. Both the Jess rule base and the Web architecture are generated from the same rule-based model, although some interaction and presentation Web design features are provided to the source model to improve code generation of the Web components. Demonstrating our proposal, a tool developed in Eclipse Modeling Project supports the creation of rule models and the automatic execution of model-to-model and model-to-code transformations, using Atlas Transformation Language (ATL) and Java Emitter Templates (JET) respectively.

As a result, a rich, functional, rule-based Web architecture is automatically generated from the rule model. This Web application is based on the Model-View-Controller architectural pattern and the JavaServer Faces technology, incorporating JBoss Richfaces components to enhance the user interface with AJAX capabilities, and integrating the Jess rule engine through the appropriate method calls to the Jess Application Programming Interface (API) in the server-side Java code. Functionality of the rule-based Web application is predefined to create, read, update and delete instances (CRUD). In contrast to current tools for automatic generation of CRUD systems that perform those functions on relational databases, our approach executes CRUD operations on the Jess rule engine working memory, enabling the inference mechanism to execute a forward-chaining inference mechanism to drive the reasoning process.

JSON Rules - The JavaScript Rule Engine

Emilian Pascalau[1] and Adrian Giurca[2]

[1]Hasso Plattner Institute, Germany,
emilian.pascalau@hpi.uni-potsdam.de
[2]Brandenburg University of Technology, Germany,
giurca@tu-cottbus.de

Abstract. TOOL PRESENTATION: There is a considerable browser potential in being able to easily wire together different services into new functionality. Usually, developers use JavaScript or related technologies to do browser programming. We present, JSON Rules, a JavaScript rule engine running Event-Condition-Action rules triggered by Document-Object-Model (DOM) Events.

For a better understanding of the context we consider the following situation: *We are looking for a job using the Monster Job Search Service. Once the job is obtained the location is shown on Google Maps.*

The Rule Engine (`http://jsonrules.googlecode.com/`) implementing the JSON Rules (Giurca and Pascalau, 2008) language was designed to fulfill a series of requirements, such as: (1) create and execute rules in a Web browser; (2) support for ECA and PR rules; (3) forward chaining rule engine, influenced by the RETE algorithm; (4) process atomic event-facts; (5) the Working Memory contains beside regular facts, event facts.

The main goal of the rule engine is to empower users with the client side abilities to model/execute web scenarios/applications/mashups by means of business rules. While in Giurca and Pascalau, 2008 the JSON Rules language has been introduced, Pascalau and Giurca, 2009 tackles the problem of creating and executing mashups by means of JSON Rules.

There is an important difference between the actual rule engines and the JavaScript Rule Engine implementing the JSON Rules language for at least two reasons: events facts are ***not static facts*** that require usual operation such as: `delete`, `update` on the Working Memory but they are ***dynamic facts***. They are dynamically consumed based on their appearance time. Second the whole engine is a **live** system: it is **reactive** because reacts based on events and it is **proactive** for by itself produces events.

The JSON Rules language tackles Production Rules and Event-Condition-Action (ECA) rules triggered by DOM events, rules of the form: `RuleID: ON EventExpression IF C1 && ... && Cn DO [A1, ...,Am]`.

Event part is optional, and denotes an event expression matching the triggering events of the rule. It uses a condition language similar with other rule systems, i.e. Drools. The syntax was influenced by the JSON notation. Any JavaScript function calls can be employed as actions.

Since there is a large palette of browsers available the engine is written entirely in JavaScript following the ECMAScript standard constructs.

The 32nd Annual Conference on
Artificial Intelligence (KI-2009)
September 15-18, 2009, Paderborn, Germany

Workshop on

Human-Machine-Interaction

http://www.mmk.ei.tum.de/ waf/ki09hmi.html

Frank Wallhoff & Gerhard Rigoll (Eds.)

KI 2009

AI And Automation

Workshop Organizers and Co-Chairs

Frank Wallhoff	Technische Universitt München, Germany wallhoff@tum.de
Gerhard Rigoll	Technische Universitt München, Germany rigoll@tum.de

Preface

Gerhard Rigoll and Frank Wallhoff

Human-Machine Communication
Technische Universität München
{rigoll|wallhoff}@tum.de

The KI-Workshop “Human-Machine-Interaction - State-of-the-art and Future Steps” will be held on 15. September 2009, as part of the satellite programme on the first day of KI-2009 in Paderborn. This also indicates the strong relationship between Artificial Intelligence and Human-Machine Interaction. In principle, intelligent systems can perform autonomously, but in most cases they require the interaction with humans, mainly because the application task can be only solved in cooperation with a human operator who communicates with the system in a natural dialogue. The capability to interact with humans in a natural way is therefore a typical feature of intelligent, cognitive systems.

On the other hand, interactive systems often make use of Artificial Intelligence methods, such as e.g. intelligent searching, reasoning, symbolic processing, knowledge representation or learning. Therefore, the relation between both areas is intensive, complex and mutual. Also similar to AI, Human-Machine Communication is a complex and manifold research area itself, covering a wide field of topics, related to major disciplines such as engineering, computer science, psychology, arts or biology. This workshop will mainly cover the technical aspects of HMI, but this does not mean at all that its scope will be narrow, because the technical component of HMI includes a large variety of modalities, such as speech, handwriting, graphics, gesture and vision, or haptics. Additionally, these modalities can be employed within a huge spectrum of applications, e.g. in automotive systems, consumer products, entertainment, virtual environments or robotics, just to name a few of the most important application areas.

The workshop programme is thus designed to reflect large parts of the above mentioned spectrum of methods and applications. It starts with a brief introduction into current trends and challenges of HMI research, continues with an overview presentation of a large international project on cognitive interactive robotics and an overview on recent developments of HMI concepts in the automotive domain, where Human-Machine Communication has made one of its most rapid developments during the last decade. A similarly rapid development has been observed for the field of ambient intelligence within the same time frame and the relation between ambient technologies and HMI is investigated in a further presentation. Multimedia and the game industry is yet another emerging application area of interactive systems and topic of another presentation. Besides the fact that HMI has successfully entered important new application areas during the last years, one should not neglect the necessity to further improve the quality of basic input/output technologies for Human-Machine Communication, such as e.g. speech recognition or gesture and emotion recognition. This can be only accomplished by continuing the basic research efforts using mostly pattern recognition and machine learning methods in speech and image processing, which is reflected by the last two presentations of the workshop.

We wish all participants an interesting and exciting event and are looking forward to exchanging with them their views of current challenges and future steps in HMI and many interesting discussions at the workshop.

Gerhard Rigoll
Frank Wallhoff

CompanionAble - Integrated Cognitive Assistive & Domotic Companion Robotic Systems for Ability & Security

Atta Badii

Intelligent Media Systems and Services Research Laboratory
School of Systems Engineering, University of Reading
Reading, UK
atta.badii@reading.ac.uk

There are widely acknowledged imperatives for helping the elderly live at home (semi)-independently for as long as possible. Without cognitive stimulation support the elderly dementia and depression sufferers can deteriorate rapidly and their carers face an increasingly demanding task. Both groups are increasingly at the risk of social exclusion.

CompanionAble will provide the synergy of Robotics and Ambient Intelligence technologies and their semantic integration to provide a Care-Giver's assistive environment. This will support the cognitive stimulation and therapy management of the care-recipient. This is mediated by a robotic companion (mobile facilitation) working collaboratively with a smart home environment (stationary facilitation).

The distinguishing advantages of the CompanionAble Framework Architecture arise from the objective of graceful, scalable and cost-effective integration. Thus CompanionAble addresses the issues of social inclusion and homecare of persons suffering from chronic cognitive disabilities prevalent among the increasing older population in Europe. The objective of the CompanionAble project is to provide a new AAL solution through a combination a mobile service robot companion that is seamlessly integrated with a smart home environment. The combined system will provide support to the Care-Recipient (CR) for the Activities of Daily Living (ADL), in particular for persons with Mild Cognitive Impairment (MCI).

The CompanionAble project will provide a care environment that supports both the CR's life-style activities and the carers, family members and professionals, for: **(i)** Realisation of intelligent day-time management, **(ii)** Content-generation for cognitive stimulation and training and coherent delivery through multiple channels (stationary and mobile), **(iii)** Medication reminders and analysis of data regarding the health status of the CR, and **(iv)** Efficient and natural social communication through networking audio-visual communication with care-givers.

Human-Machine interaction, in particular with the service robot, will be supported by advanced dialogue systems in combination with person tracking based on various modalities, such as vision, audio, as well as other sensors (e.g. infra-red sensors). Emotional awareness, supported by situation awareness, will enable the system to become an accepted companion in the life of the Care-Recipient, supported by the smart home facilities.

Further information on the CompanionAble and the list of project partners is available on www.companionable.net.

Augmented reality in advanced driver assistance systems

Stefan Müller-Schneiders, Stephanie Hold, Mirko Meuter, Dennis Müller, Christian Nunn

Delphi Electronics & Safety, D-42119 Wuppertal, Germany
stefan.mueller-schneiders@delphi.com

Abstract. In this paper recent research trends in the automotive industry which utilize advanced driver assistance systems (ADAS) in combination with augmented reality techniques are presented. Thereby the main focus is on computer vision based applications such as e.g. traffic sign recognition (TSR) and forward collision warning (FCW). Since due to the chosen camera sensor an image of the surrounding exists, which is usually the road ahead, it is an attractive idea to use this image for the generation of user information as augmented reality. Increasingly the ADAS systems utilize a fusion approach where besides the camera sensor, radar sensors and the digital map as a sensor are employed. By adding the digital map, additional attractive ADAS functions can be considered, namely curve speed warning (CSW) and of course GPS-based navigation. Also for these functions, information can be provided to the driver by means of a human machine interface based on virtual reality technology.

Augmented reality in ADAS

In general, an advanced driver assistance system aims at supporting the driver during his driving task. Especially vision based ADAS systems reached a high level of maturity and migrated from the research domain to the product level. As a first example, one might look at the traffic sign recognition (TSR) application (see also [1]). TSR systems support the driver during the tedious task of analyzing and remembering the large amount of traffic signs that are regulating the traffic on the streets. Such a system allows for example to warn the driver if he violates the speed limit. Fig. 1 shows a detected and classified speed limit sign which is highlighted by a red circle and annotated with the recognized class. The visualization of the acquired information as shown in the figure is typical for the development phase of an application. Once the product level is reached, usually only the recognized sign is shown in the dash board or in a head up display (HUD). Due to the fact that the automotive cameras are usually geometrically calibrated (see e.g. [2]), it is also possible to locate the detected signs in real world coordinates. Therefore, it is also possible to overlay the image of the real world with customized versions of the detected traffic signs. This could be e.g. translations of the detected signs into a sign type the driver is familiar with. This case is depicted in Fig. 1 on the right hand side, where a German circular speed sign is translated to the corresponding US sign. This could be beneficial

Fig. 1: Traffic sign recognition system displaying country specific signs

to drivers which drive in foreign countries and in terms of HMI preferably displayed to the driver via a HUD. The performance of the TSR application can be increased by means of fusion with a digital map which also allows additional ADAS functions such as CSW and navigation. In the navigation case, arrows can be overlayed to the real world images which has been done in products already on the market. In CSW dangerous curves can be annotated with virtual speed limit signs that prevent the driver from approaching a dangerous curve too quickly (see Fig. 2). In order to provide a naturally looking scene, when inserting a virtual traffic sign, it is beneficial to analyze the course of the road ahead by means of fusing vision based lane detection with the information of the digital map (see also the yellow markings in Fig. 2). Using this information, the virtual speed limit sign can be positioned more accurately. An alternative to the virtual speed limit sign is a virtual carpet that encodes the position of the dangerous curve as well as the threat level (color coded). This is also illustrated in Fig. 2. A similar visualization can be chosen in the case of forward collision warning (FCW), where the virtual carpet might encode a following distance violation to other road users. The cases mentioned above for utilizing augmented reality techniques in combination with ADAS functionality are first examples. In the near future, many more ideas will be generated by research groups worldwide. However, many technical problems need to be solved too, the HUD technique needs to be improved and image analysis techniques also still face many challenges.

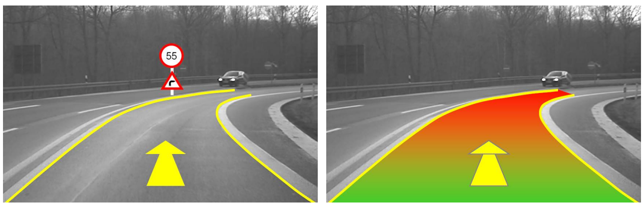


Fig. 2: Curve speed warning application

References

1. S. Müller-Schneiders, C. Nunn, M. Meuter, "Performance Evaluation of a Real Time Traffic Sign Recognition System", *IEEE Intel. Vehicles Symposium*, 2008.
2. S. Hold, S. Görmer, A. Kummert, M. Meuter, and S. Müller-Schneiders, "A novel approach for the online initial calibration of extrinsic parameters for a car mounted camera", *IEEE Intern. Conf. on Intelligent Transportation Systems*, Ohio, 2009.

HMI for aged and disabled people

Dr. Frank Stein

Weimed GmbH
Dueren, Germany
f.stein@weimed.de

The group of aged and disabled humans are classified into visually, cognitive and deaf users of machines. The groups are distributed in 75% aged people (older then 60) and 25% less then 60 years. Parameters to analyze the groups are age, disability, social context, financial context, cognitive background and context or background of profession and education. The conditions for users, which are disabled or aged, are different from the situation of normal users. Both groups have special requirements for human machine interfaces.

Visually disabled users of machines and devices of daily living for housekeeping, work, and personal use have special technical requirements of the front end. Parts of the front end are all input- and output components like display, printer, keyboard or alternative input methods. Both methods need high contrast, positive and inverse contrast, diagnosis of visually disease related colours, high brightness and simple workflow.

Cognitive disabled users of machines and devices of daily living for housekeeping, work and personal use have also special technical requirements of the front end. The interface must be simple for use. To be simple can have different methods of input or output. Some of these methods are known from visually disabled users or simplified methods of not disabled users. Cognitive disability is more different as visually disability. A simple form of cognitive disability is a less powerful brain or the start of dementia. The biggest groups of aged people with cognitive disabilities are patients with apoplexy or aphasia. A part of this group has also a physical disability and lost of voice, depends on neurological and motorically problems of the face and cervical muscles. A small part of this group has also physical problems with the hands.

The group of deaf users has only the requirement to have output or input methods without audible elements. All audible elements must be transferred to optical elements.

There are also groups of combined disabilities. This groups has also the combination of requirements of the single groups.

Examples for HMI are mobile phones, cash machines, self service machines in banks or shops or technical aids for ambient living at home. We developed a technical aid, which substitute the human voice for patients with apoplexy and aphasia. This machine have special input methods (text and symbol) and was rolled out with a field study. The study is not finished but it exists milestone results.

Video Games Can Improve Performance in Sports – An Empirical Study with Wii™Sports Bowling

Andreas Siemon, Rene Wegener, Florian Bader, Thomas Hieber, and Ute Schmid

Faculty Information Systems and Applied Computer Science
Otto-Friedrich University Bamberg, Germany
thomas-wolfgang.hieber@stud.uni-bamberg.de, ute.schmid@uni-bamberg.de

According to studies from the medical domain it can be assumed that the training of sensumotor abilities by the use of video game consoles positively affects performance in real life situations. With this in mind we investigated the assumption that – at least for bowling novices – training of bowling games with the **Nintendo Wii** can have positive effects on their performance on a real bowling alley.

We conducted an empirical study with 32 students (25 male, 7 female) from the University of Bamberg. One group ($n_1 = 15$) of participants took part in a video game training session while the other group ($n_2 = 17$) received no special training. After that, participants of both groups played on the bowling alley. The lag between training and testing on the bowling alley was 1-2 days. Previous experience with bowling was assessed in a questionnaire.

The main dependent variable was the bowling score students achieved during the real bowling game. Results show a significant positive effect ($t = -2,48$, $df = 30$, $p < 0.005$) of console training on the mean scores in bowling for the group which trained with the Nintendo console ($x_1 = 105.41, sd_1 = 25.88$) in contrast to the non-training group ($x_2 = 85,47$, $sd_2 = 18,35$).

The evaluation of the questionnaires shows that there is no evidence for potentially biased group consistency concerning the level of expertise. Both inexperienced and experienced (in our case expert) participants were positively affected by the training received, with the score of novices improving more than the scores of expert participants.

Facial Expressions and Headgestures for intuitive HCI - Word

Christoph Mayer and Bernd Radig

Chair for Informatics IX –
Image Processing and Knowledge-based systems
Technische Universität München
{mayerc|radig}@in.tum.de

Humans-Human communication is known to be very effective in terms of the amount of information passed and the time required. Therefore, bringing traditional human-human communication schemes to human-machine interfaces is a valuable scientific aim and a lot of research has been conducted on this end. However, although computers have been widely deployed to our daily lives, traditional computer-human interaction still lacks intuition. Recent research on computer vision tackles this gap by the integration of vision-based facial expression recognition.

When estimating facial expressions from images and sequences of images automatically, we are tasked with a number of different problems to solve:

Firstly, the face has to be automatically detected within the image.
Secondly, specific information semantic to facial expressions has to be extracted.
Finally, we are tasked with classifying the facial expression shown in the image.

All three steps greatly benefit from the application of machine learning techniques, ranging from decision trees to hidden Markov models (HMMs) and support vector machines. We will present an approach that tackles all sub-steps with machine learning techniques to finally determine the facial expression currently visible in the image.

SPEAKER INDEPENDENT URDU SPEECH RECOGNITION USING HMM

Javed Ashraf[1], Dr Naveed Iqbal[2], Naveed Sarfraz Khattak[3], Ather Mohsin Zaidi[4]
College of Signals, National University of Sciences and Technology (NUST)
Rawalpindi 46000, Pakistan.

[1]javedashraf27@gmail.com ,[2]naveedrao@gmail.com, [3]khattakn@mcs.edu.pk,
[4]athar-mcs@nust.edu.pk

Automatic Speech Recognition (ASR) is one of the advanced fields of Natural Language Processing (NLP). Recent past has witnessed valuable research activities in ASR in English, European and East Asian languages. But unfortunately South Asian Languages in general and “Urdu” in particular have received very less attention. In this paper we present an approach to develop an ASR system for Urdu language. The proposed system is based on an open source speech recognition framework called Sphinx4 which uses statistical based approach (Hidden Markov Model) for developing ASR system. We present a Speaker Independent ASR system for medium sized vocabulary, i.e. fifty two isolated most spoken Urdu words and suggest that this research work will form the basis to develop medium and large size vocabulary Urdu speech recognition system.

The 32nd Annual Conference on
Artificial Intelligence (KI-2009)
September 15-18, 2009, Paderborn, Germany

Workshop on

Machine Learning in Real-time Applications (MLRTA 09)

http://www.hs-owl.de/init/aktuelles/call-for-papers-mlrta-09.html

Volker Lohweg & Oliver Niggemann (Eds.)

Workshop Organizers and Co-Chairs

Volker Lohweg	Ostwestfalen-Lippe University of Applied Sciences, Germany volker.lohweg@hs-owl.de
Oliver Niggemann	Ostwestfalen-Lippe University of Applied Sciences, Germany oliver.niggemann@hs-owl.de

Program Committee

Jürgen Belz	Hella KGaA Hueck & Co., Lippstadt, Germany
Olaf Enge-Rosenblatt	Fraunhofer Institute for Integrated Circuits, Dresden, Germany
Ulrich Heinkel	Chemnitz University of Technology, Germany
Eyke Hüllermeier	Philipps-University Marburg, Germany
Hans Kleine Büning	University of Paderborn, Germany
Henrik Legind Larsen	Esbjerg Institute of Technology, Denmark
Dinh Khoi Le	Wincor Nixdorf International GmbH, Paderborn, Germany
Tim W. Nattkemper	Bielefeld University, Germany
Ralf Salomon	University of Rostock, Germany
Johannes Schaede	KBA-Giori S.A., Lausanne, Switzerland
Dirk Spenneberg	dSPACE GmbH, Paderborn, Germany
Benno Stein	Bauhaus-University Weimar, Germany

Preface

Volker Lohweg, Oliver Niggemann

inIT – Institute Industrial IT
Department of Electrical Engineering and Computer Science
Ostwestfalen-Lippe University of Applied Sciences
Liebigstr. 87, 32657 Lemgo, Germany
{volker.lohweg, oliver.niggemann}@hs-owl.de

The intention of the *Machine Learning in Real-Time Applications* (MLRTA 2009) workshop was to discuss machine learning concepts suitable for real-time applications.

Cognitive systems are successfully applied in different industries, such as Automotive, Telecommunication, Robotics, Image Processing Based Automation as well as Machine and Plant Engineering. The complexity of such systems and situations very often makes manual solutions to tasks such as classification, diagnosis, and model identification more and more unrealistic. Instead, machine learning algorithms are applied.

Industrial cognitive systems deal with processing information from communication and automation systems under the criteria of process real-time, robustness, and limitation of resources. In such industrial environments the machine learning community is facing the problem that more and more tasks must be solved under real-time constraints. In the future, a quality assurance for modern production lines must rely on multi-sensory real-time data analysis and sensor fusion approaches. The research demand viewed in the light of applicability is immense and will rise in the near future.

The focus of the workshop is therefore based upon the description, modelling, and the design of machine learning algorithms that can be implemented efficiently in micro-electronic circuits and resource-limited distributed systems. The following topics for contributions were proposed:

- Practical applications and architectures for machine learning
- Knowledge representation
- Classifier design under hardware and software resource limitations
- Fast machine learning algorithms
- Concepts and strategies for distributed classifiers
- Data logging for real-time machine learning
- Diagnosis under real-time constraints
- Adaptive real-time systems
- The future of distributed system machine learning

The choice of workshop contributions are based on the criteria of paper quality and variety of topics. Therefore, an selection of papers which represents different concepts and approaches was chosen by the TPC.

We are happy to announce as Key Note speakers Dr. Tilmann Seubert and Jürgen Belz from Hella Hella KGaA Hueck & Co. Their Key Note title is *The way to full autonomous road vehicles* The authors are focusing on the topics

- social requirements to reduce fatal injuries,
- sensor fusion,
- assistants to overrule drivers in unavoidable incidents,
- generation with autonomous driving.

In the following sessions a widespread (Medicine, Hardware Implementation, Fuzzy Pattern Classification, Control Systems, and Network Analysis) composition of papers are presented:

- Assisting Telemanipulation Operators via Real-Time Brain Reading
- FPGA-Based Real-Time Monitoring of In-Vitro Nerv Cells
- Aggregation Operator Based Fuzzy Pattern Classifier Design
- Dynamic Rate Adaptation in Self-Adapting Real-Time Control Systems
- Timed Automata for Modeling Network Traffic

We wish all the participants an successful and interesting workshop with many fruitful discussions.

Lemgo, August 2009
Volker Lohweg
Oliver Niggemann

Key Note: The way to full autonomous road vehicles

Tilmann Seubert, Jürgen Belz

Hella KGaA Hueck & Co
Beckumer Str. 130
D-59552 Lippstadt, Germany
{tilmann.seubert, juergen.belz}@hella.com

The first safety systems that appeared on vehicles contributed to minimise effects of collisions,increasing the so called passive safety of the car. Retention belts and airbags belong to that category. Later, systems for vehicle stability, such as ABS and ESP appeared. Acting upon the vehicle dynamics, they contributed to reduce significantly the number of accidents and their evereness. Some new Advanced Driver Assistance Systems (ADAS), which were first only designed for driver convenience, such as Adaptive Cruise Control, Side Obstacle Warning, Lane Departure Warning or other image based functions, contribute to improve road safety, by ensuring that drivers do not put themselves and others at risk. In the next generation of these systems, often based on sensor data fusion, functionality will be enhanced in order to increase both acceptance and availability. In addition, new functionalities such as heading control, hazard warning, pre-crash information and collision mitigation will further enhance driver warnings and vehicle interactions, contributing to further reduction of the number of collisions and their consequences. Beyond these safety aspects those systems can be additionally used for improvements in convenience. It remains to be seen, if autonomous driving is an accepted convenience function for vehicles in future. One main milestone in this direction will be the driver's behaviour adaptation. In any situation where assistants take over, the operator must have the faith to master the situation. This is the biggest challenge since human beings are individuals and it is the right time for intensively starting research in this area.

Dr. Tilmann Seubert is Vice President *Advanced Development.* J"urgen Belz is Director *Processes, Methods and Tools.*

Assisting Telemanipulation Operators via Real-Time Brain Reading

Elsa Andrea Kirchner[1,2], Jan Hendrik Metzen[2], Timo Duchrow[2], Su Kyoung Kim[1,2] and Frank Kirchner[1,2]

[1] Robotics Lab
University of Bremen
Robert-Hooke-Stra"se 5, 28359 Bremen, Germany
[2] Robotics Innovation Center (RIC)
German Research Center for Articial Intelligence (DFKI GmbH)
Robert-Hooke-Stra"se 5, 28359 Bremen, Germany
{elsa.kirchner, jan_hendrik.metzen, timo.duchrow, su_kyoung.kim, frank.kirchner}@dfki.de

In this paper, we present a concept for a new kind of man-machine interface that is based on the monitoring of brain activity and aimed at supporting operators in telemanipulation scenarios. This monitoring that takes place unnoticed by the subject and is called brain reading. A brain reading interface (BRI) is a highly integrated control environment that observes brain signals in real time. Consciously recognized and classified stimuli evoke a certain response in the operators brain activity that will be detected by the BRI. Based on the detection of these changes in brain responses in the electroencephalogram (EEG), a brain reading system is able to discern whether a piece of information that has been presented to the operator was acknowledged or not. Hence, the BRI ensures that environmental alerts are processed and classified by the operator and can thus be a crucial component of control systems ensuring that operators perceive and cognitively process alerts presented to them during highly demanding tasks like complex manipulations. We show that brain activity changes that correlate with the classification of important, task-relevant stimuli in multi-task telemanipulationlike scenarios are stable. Furthermore, we outline a concept for a BR system that allows the detection of these brain activity changes in single trial EEG epochs based on machine-learning methods.

iCAMS: An FPGA-Based System for the Real-Time Monitoring of the Activity of In-Vitro Cells

Marian Lüer, Tom Reimer, Ralf Salomon

Institute of Applied Microelectronics and Computer Engineering
Rostock University
Richard-Wagner Str. 31, 18119 Rostock-Warnem"unde, Germany
{marian.lueder, tom.reimer, ralf.salomon}@uni-rostock.de

In order to employ biological neurons within technical applications, they (1) must expose themselfes to a monitoring system and (2) must allow external (stimuli) signals to guide their growing. A first step towards this goal has recently been made by the development of so called neuro chips, which provide an excellent bio-physical interface. This paper proposes the in-vitro-cell-activity-monitoring system (iCAMS) that links the neuro chip with a regular PC. iCAMS (pre-) processes all the data from all the electrodes in parallel, does further user-specified filtering, analyses, and temporary storage in local memory. iCAMS also provides a USB or Ethernet-based communication link to the PC. Even though, iCAMS is implemented on a low-cost, standard FPGA, it processes the 52 electrodes at about 10 KHz with a resolution of 10 bits.

Aggregation Operator Based Fuzzy Pattern Classifier Design

Uwe Mönks[1], Volker Lohweg[1] and Henrik Legind Larsen[2]

[1] inIT – Institute Industrial IT
Department of Electrical Engineering and Computer Science
Ostwestfalen-Lippe University of Applied Sciences
Liebigstr. 87, 32657 Lemgo, Germany
{uwe.moenks,volker.lohweg}@hs-owl.de
[2] Department of Electronic Systems
Aalborg University
Niels Bohrs Vej 8, 6700 Esbjerg, Denmark
legind@cs.aaue.dk

This paper presents a novel modular fuzzy pattern classifier design framework for intelligent automation systems, developed on the base of the established Modified Fuzzy Pattern Classifier (MFPC) and that allows designing novel classifier models which are hardware-efficiently implementable. The performances of novel classifiers using substitutes of MFPC's geometric mean aggregator are benchmarked in the scope of an image processing application against the MFPC to reveal classification improvement potentials for obtaining higher classification rates.

Dynamic Rate Adaptation in Self-Adapting Real-Time Control Systems

Werner Brockmann, Nils Rosemann and Christian Lintze

Smart Embedded Systems Group
University of Osnabrück
Albrechtstraße 28
49069 Osnabrück, Germany
nils@informatik.uni-osnabrueck.de

Self-tuning and self-optimizing control systems have the potential to greatly ease the engineering effort involved in designing control systems for certain scenarios. These scenarios include processes which are time-variant, or which are subject to different kinds of disturbances and uncertainties. In order to actually perform online or self-optimization, such control systems have to determine changes of their internal parameters incrementally based on the observable closed-loop behavior, i. e., they have to learn in real-time. This paper analyzes the problems of real-time, closed-loop learning concerning the stability-plasticitydilemma. It is then argued that a certain meta-level control of the learning process itself is necessary which goes beyond e. g., simulated annealing. Thus, the FRANCA approach is introduced which dynamically adapts parameters of the learning process towards an appropriate balance between stability and plasticity based on heuristics. Its feasibility is shown by investigations on self-tuning control of a simulated throttle plate for the static case and also for the case of time-variant behavior.

Timed Automata for Modeling Network Traffic

Barath Kumar, Oliver Niggemann and Jürgen Jasperneite

inIT – Institute Industrial IT
Department of Electrical Engineering and Computer Science
Ostwestfalen-Lippe University of Applied Sciences
Liebigstr. 87, 32657 Lemgo, Germany
{barath.kumar, oliver.niggemann, juergen.jasperneite}@hs-owl.de

State based modeling is one successful approach for modeling complex systems. However, statecharts abstract away from timing information and assume an infinitely fast system. As real-world entities consume time for performing operations, it is crucial to have appropriate means to express temporal constraints. Hence, temporal state-based formalisms like timed automata seem to be suitable for modeling complex real-time systems. Timed automata extend automata using a continuous notion of time. Several timed automata have been introduced in the recent years - hence, it is essential to study their potentials so that they can be used appropriately.

The 32nd Annual Conference on
Artificial Intelligence (KI-2009)
September 15-18, 2009, Paderborn, Germany

Workshop on

Planning, Scheduling, Design, and Configuration (PuK 2009)

http://www.hs-owl.de/init/aktuelles/call-for-papers-mlrta-09.html

Jürgen Sauer, Stefan Edelkamp & Bernd Schattenberg
(Eds.)

Workshop Organizers and Co-Chairs

Jürgen Sauer	Universität Oldenburg, Germany juergen.sauer@uni-oldenburg.de
Stefan Edelkamp	Universität Dortmund, Germany stefan.edelkamp@cs.uni-dortmund.de
Bernd Schattenberg	Universität Ulm, Germany bernd.schattenberg@uni-ulm.de

Program Committee

Jürgen Belz	Hella KGaA Hueck & Co., Lippstadt, Germany
Stefan Edelkamp	University of Bremen, TZI, Germany
Lothar Hotz	University of Hamburg, Germany
Thorsten Krebs	University of Hamburg, Germany
Jrgen Sauer	University of Oldenburg, Germany
Bernd Schattenberg	University of Ulm, Germany
René Schumann	University of Frankfurt, Germany

Preface

Jürgen Sauer[1], Stefan Edelkamp[2], Bernd Schattenberg[3]

[1] Universität Oldenburg
Department für Informatik
D-26111 Oldenburg
juergen.sauer@uni-oldenburg.de

[2] TZI - Technologie-Zentrum Informatik
Universität Bremen
D-28357 Bremen
edelkamp@tzi.de

[3] Universität Ulm
Institut für Künstliche Intelligenz
D-89069 Ulm
bernd.schattenberg@uni-ulm.de

The PuK (from **P**lanen **u**nd **K**onfigurieren) workshop has a long tradition in combination with the German KI-conferences. It started in 1987 as Workshop on planning and configuration and extended its scope as well as its name to the areas of scheduling and design. Since 2000 the workshop is held in conjunction with the KI conference.

The PuK workshop is the regular meeting of the special interest group on planning, scheduling, design and configuration within the AI section of the GI. As in the years before the PuK Workshop aims at the exchange of ideas, techniques, methodologies etc. used in the different application areas. Thus the workshop brings together researchers and practitioners of the areas of planning, scheduling, design and configuration.

The general topics of interest of the PuK community include but are not limited to:

- Practical applications of configuration, planning or scheduling systems
- Architectures for planning, scheduling or configuration systems
- Knowledge representation and problem solving techniques, e.g. domain-specific techniques; heuristic techniques; distributed problem solving; constraint-based techniques; iterative improvement; integrating reaction and user-interaction.
- Learning in the context of planning, scheduling and design.

Within the 23rd workshop papers from nearly all areas in scope are presented. Therefore it again gives an actual overview on the research activities in Germany. The papers from the “planning area” describe the application of planning techniques in games (Edelkamp/ Kissmann and Edelkamp/ Messerschmidt/ Sulewski/ Yücel) and the use of AI planning for service composition (Marquardt/ Uhrmacher) . The papers from Schumann/ Sauer and Gebhardt are dealing with reactive scheduling problems, which are actually discussed in this area. The paper of Günther/ Nissen presents a comparison of meta heuristics used for solving a staff scheduling problem. Mies/ Mertens present the extension of an agent based simulation tool. The presentation of Krebs deals with the transition of configuration research results into industrial applications.

This composition of papers is also intended to stimulate a mutual exchange among the researchers and to look for future research directions. With this the workshop is supporting the research planning in all areas.

At least, we have to thank the program committee consisting of

- Stefan Edelkamp, University of Bremen, TZI, D
- Lothar Hotz, University of Hamburg, D
- Thorsten Krebs, University of Hamburg, D
- Jürgen Sauer, University of Oldenburg, D
- Bernd Schattenberg, University of Ulm, D
- René Schumann, University of Frankfurt, D.

Jürgen Sauer
Stefan Edelkamp
Bernd Schattenberg

encoway - Abstract zum eingeladenen Vortrag

Thorsten Krebs, K. Christoph Ranze and Oliver Hollmann

encoway GmbH
Buschhöhe 2, 28357 Bremen
info@encoway.de

1 Einleitung

Die encoway GmbH, führender Anbieter von Softwarelösungen für Angebots- und Konfigurationslösungen in der Investitionsgüterindustrie, wurde im Jahr 2000 als Tochterunternehmen der Lenze AG und Spin-Off des Technologie Zentrums Informatik (TZI) der Universität Bremen gegründet. Die Unternehmensidee entstammt einem Konfigurator-Projekt, das in den Jahren 1995 bis 2000 im TZI der Universität Bremen in einer Kooperation mit dem Industriepartner und Spezialisten für Antriebssystemlösungen, der Lenze AG, erfolgreich durchgeführt wurde. Ziel des Projekts war es, die technische Angebotsklärung für Antriebslösungen mit Hilfe von Methoden aus der Produktkonfiguration zu optimieren. Die damals konzipierte Lösung Lenze Drive Solution Designer wurde 2002 mit dem Deployed Application Award der AAAI ausgezeichnet und wird heute weltweit bei der Antriebsauslegung in der Lenze Gruppe und bei Kunden erfolgreich eingesetzt. Man hat damals im TZI und bei Lenze frühzeitig erkannt, dass die Konfigurator- Kerntechnologie EngCon domänenunabhängig eingesetzt werden kann, so dass sich neben der Antriebstechnik ein breiter Markt für den Einsatz der Technologie abgezeichnet hat. Heute zählt die Lenze AG neben zahlreichen führenden Industrieunternehmen zu den Kunden der encoway GmbH. In diesem Vortrag reflektieren wir zehn Jahre Unternehmenshistorie und insgesamt mehr als zwanzig Jahre Technologiehistorie in der strukturbasierten Konfigurierung. Wir betrachten die Entwicklung und heutige Sicht der Technologie von der ersten Forschung bis hin zur heutigen Anwendung in der Industrie und diskutieren die in dieser Zeit gesammelten Erfahrungen.

2 Technologie

Der EngCon Konfigurator nutzt und erweitert bewährte Konzepte und Methoden aus dem an der Universität Hamburg entwickelten Konfigurationswerkzeug KONWERK, die in zahlreichen Industriepototypen in den 1990er Jahren erfolgreich eingesetzt werden konnten. Das Produktwissen wird in einer Baukasten-Manier in strukturbasierter Form abgelegt. Mit dieser ausdrucksstarken Repräsentation können verschiedene Schwerpunkte in der heutigen Anwendung abgedeckt werden: Produktvarianten zur Selektion, Produktkonfiguration, Vertriebsmodelle und Applikationsmodelle:

- Selektion von in einem Katalog komplett beschriebenen Produktvarianten ist die einfachste Form der Anwendung. Diese Anwendung beinhaltet keine Konfiguration, ist jedoch ein Abfallprodukt der strukturbasierten Repräsentation der Produktdomäne.
- Produktkonfiguration beschreibt das Zusammenfügen von vordefinierten Komponenten basierend auf möglichen Strukturen und definierten Restriktionen für die Kombination. Dies ist die Kernanwendung für encoway.
- Vertriebsmodelle beschreiben die Produktdomäne aus einer für den Kunden verstehbaren Sicht. Hier wird typischer Weise nicht auf technischen Details fokussiert, sondern es rücken für den Kunden interessante Features in den Vordergrund.
- Applikationsmodelle erhöhen die Abstraktion ein weiteres Mal gegenüber Vertriebsmodellen. In einem Applikationsmodell wird ein generelles Ziel beschrieben und mit dieser Information erst später erkannt, mit welcher Art von Produkt dieses Ziel erreicht werden kann.

3 Lessons Learned

Der Vortrag wird neben dem oben kurz eingeführten Ebenenmodell auch detailliert auf die Frage eingehen: Was haben wir gelernt? Dieser Punkt beinhaltet eine Auflistung der Technologien, die zu Beginn und in begleitenden Forschungsarbeiten auch während des Konfigurations-Projekts propagiert wurden und eine Analyse welche dieser Technologien aus heutiger Sicht sinnvoll eingesetzt werden (können) und welche Technologien keine reale Anwendung gefunden haben. Die Repräsentation in strukturbasierter Form ist für alle Anwendungen der encoway ein adäquates Werkzeug. über das im vorigen Abschnitt beschriebene Ebenenmodell kann diese Technologie an das jeweilige Szenario skalierbar eingesetzt werden. Als weitere Technologien werden im Vortrag beispielweise prozedurales Wissen (Agenda-basiertes Vorgehen), unsicheres Wissen, Konfiguration von Software und Erklärungen von Anhängigkeiten oder Konflikten analysiert.

4 Aktueller Forschungsbedarf

Abschließend wird der Vortrag aktuellen Bedarf an Forschung und Entwicklung für den praktischen Einsatz in der Industrie motivieren. Aktuelle Themen in diesem Bereich sind aus Sicht der encoway verteiltes Modellieren, Modellpflege/-Modellierung allgemein, mehrstufige Konfiguration und verteiltes Konfigurieren.

- In realistischen Szenarien ist es häufig wenig pragmatisch, mit einzelnen monolithischen Wissensbasen zu arbeiten; sei es wegen der Datenmenge, verteilter Expertise oder von einander unabhängiger Teil-Domänen.
- Effektive und effiziente Modellpflege ist seit jeher eine der größten Herausforderungen bei der Anwendung von Konfiguratoren. Zu einem Modell gehören darüber hinaus auch immer zusätzliche Informationen wie beispielsweise Dokumentationen, Hilfetexte, Bilder, usw. Die Integration von PIM-Sytemen ist eine vielversprechende Idee auf diesem Weg.

– Wenn konfigurierbare Produkte als Teile von größeren, konfigurierbaren Produkten modelliert werden, dann redet man von mehrstufiger Konfiguration. Die Abbildung solcher Szenarien in die strukturbasierte Konfiguration ist ein aktuelles Thema bei encoway.
– Verteilte Konfiguration beschreibt ein Szenario, in dem mehrere (Teil-)Modelle getrennt voneinander konfiguriert werden, jedoch Informationen über eine definierte Schnittstelle austauschen. über ein solches Szenario wird bei encoway das Zusammenspiel zwischen einem Technikmodell und einem Vertriebsmodell realisiert.

Gamer: A Hybrid General Game Player

Peter Kissmann[1] and Stefan Edelkamp[2]

[1] Fakultät für Informatik, TU Dortmund, Germany
`peter.kissmann@tu-dortmund.de`
[2] Technologie-Zentrum Informatik, Universität Bremen, Germany
`edelkamp@tzi.de`

General Game Playing (GGP) is an interesting research topic, especially in recent years, since Stanford's Logic Group presented the Game Description Language GDL and with it announced the first GGP competition. In GGP, the programmers need to write players that can cope with any game given in GDL. They do not know beforehand, what kind of game will be played, so they cannot add any specialized knowledge.

During the last few years, the competition's winners used an implementation of the UCT algorithm [3]. This was also the case in this year's competition, where we participated with our hybrid game playing agent Gamer for the first time.

Here, we will present the workings and interplay of the different parts of this player. It consists of several parts, one the most important of these being a parallel UCT player using the game description after transforming it to Prolog. Another important part is a general game solver (see, e.g., [1]) that can optimally solve single- and two-player turn-taking games – if they are not too complex. For this solver, a special variable-free input is required, so before starting the solver, we first of all instantiate the game's description [2].

In the competition we found that we could optimally solve 4 of the 12 games played, although only one of these was complex enough that the UCT players could not handle it. But unfortunately, due to a bug in a parser, we only just missed the final round – and in the consolation round ended third of four teams. So, to further improve our player, we are currently working on a UCT player that also uses the instantiated input, as this is typically faster than Prolog's unification mechanism needed with the uninstantiated input.

References

1. S. Edelkamp and P. Kissmann. Symbolic classification of general two-player games. In *KI*, volume 5243 of *LNCS*, pages 185–192, 2008.
2. P. Kissmann and S. Edelkamp. Instantiating general games. In *IJCAI-workshop on General Game Playing (GIGA)*, 2009.
3. L. Kocsis and C. Szepesvári. Bandit based Monte-Carlo planning. In *ECML*, volume 4212 of *LNCS*, pages 282–293, 2006.

Efficiently Solving Games with Perfect Hash Functions

Stefan Edelkamp, Hartmut Messerschmidt, Damian Sulewski
TZI, Universität Bremen, Germany

Cengizhan Yücel
Technische Universität Dortmund, Germany

Abstract

In this paper, we propose an efficient method of solving one- and two-player combinatorial games by mapping each state to a unique bit in memory.

In order to avoid collisions, a concise portfolio of perfect hash functions is provided. Such perfect hash functions then address tables that serve as a compressed representation of the search space and support the execution of exhaustive search algorithms like breadth-first search and retrograde analysis.

Perfect hashing computes the *rank* of a state, while the inverse operation *unrank* reconstructs the state given its rank. Efficient algorithms are derived, studied in detail and generalized to a larger variety of games. We study rank and unrank functions for permutation games with distinguishable pieces, for selection games with indistinguishable pieces, and for general reachability sets. The running time for ranking and unranking in all three cases is linear in the size of the state vector.

To overcome space and time limitations in solving previously unsolved games like *Frogs-and-Toads* and *Fox-and-Geese*, we utilize parallel computing power in form of multiple cores as available on modern central processing units (CPUs) and graphics processing units (GPUs). We obtain an almost linear speedup with the number of CPU cores. Due to its much larger number of cores, even better absolute speed-up are achieved on the GPU.

1 Introduction

Strong computer players for combinatorial games like *Chess* [6] have shown the impact of advanced AI search engines. For many games they play on expert and world championship level, sometimes even better. Some games like *Checkers* [26] have been decided, in the sense that the solvability status of the initial state has been computed. (The game is a draw, assuming optimal play of both players.)

In this paper we consider *solving* a game in the sense of creating an optimal player for *every* possible initial state in the input. This is achieved by computing the game-theoretical value of each state, so that the best possible action can be selected by looking at all possible successor states. In single-agent games the value of a game simply is its goal distance, while for two-player games the value is the best possible reward assuming that both players play optimally.

Our approach is based on perfect hashing, where a perfect hash function is a one-to-one mapping from a set of states to some set $\{0, \ldots, m-1\}$ for a sufficiently small number m. *Ranking* maps a state to a number, while *unranking* reconstructs a state given its rank. One application of ranking and unranking functions is to compress and decompress a state.

Provided the state space on disk, minimum perfect hash functions with a few bits per state can be constructed I/O efficiently. Botelho et al. [4] devise minimal practical hash functions for general state spaces, once the set of reachable states is known. The approach requires some small constant number c of bits per state (typically, $c \approx 4$). Of course, perfect hash functions do not have to be minimal to be space-efficient. Non-minimal hash functions can outperform minimal ones, since the gain in the constant number c of bits per state for the hash function can become smaller than the loss in coverage.

We will see that for many search problems space-efficient perfect hash functions can be constructed prior to executing the search. Sometimes it is even possible to devise a family of perfect hash functions, one for each (forward or backward) search layer. We propose linear time algorithms for invertible perfect hashing for a wide selection of AI search problems, including

- *permutation games*, i.e., games with distinguishable objects. In this class we find *Sliding-Tile* puzzles with numbered tiles, as well as *Top-Spin* and *Pancake* problems. The *parity* of a permutation will prove to be an important concept as it often allows to restrict the range of the hash function to half.

- *selection games*, i.e., games with indistinguishable objects. In this class we find tile games like *Frogs-and-Trouts*, as well as strategic games like *Peg-Solitaire* and *Fox-and-Geese*.

For analyzing the state space, we utilize a bitvector that covers the solvability information of all possible states. Moreover, we apply symmetries to reduce the time- and space-efficiencies of our algorithms. Besides the design of efficient perfect hash functions that apply to a wide selection of games, one important contribution of the paper is to compute successors states on multiple cores on the central processing unit (located on the motherboard) and on the graphics processing unit (located on the graphics card).

This paper extends observations made in [14], where only permutation games have been considered, to a wider selection of state space problems. To the best of our knowledge, the only attempts to use state space search on GPUs was by the authors of this paper in the context of model checking [13, 3]. In [13] they improved large-scale disk-based model checking by shifting complex numerical operations to the graphic card. As delayed elimination of duplicates is the performance bottleneck, the authors performed parallel processing on the GPU to improve the sorting speed significantly. Since existing GPU sorting solutions like Bitonic Sort and Quicksort do not obey any speed-up on state vectors, they propose a refined GPU-based Bucket-Sort algorithm. In [3] algorithms for parallel probabilistic model checking on GPUs were proposed. For this purpose the authors exploit the fact that some of the basic algorithms for probabilistic model checking rely on matrix vector multiplication. Since this kind of linear algebraic operations are implemented very efficiently on GPUs, the new parallel algorithms achieve considerable runtime improvements compared to their counterparts on standard architectures.

The paper is structured as follows. First, we provide preliminaries on perfect, invertible, orthogonal hash functions, on permutation parity and move alternation properties. Then we study constant-bit breadth-first search and constant-bit retrograde analysis. We discuss time-space trade-offs using only one bit per state. Next, we address the design of efficient rank and unrank functions for games with distinguishable and indistinguishable pieces, and adapt the algorithms to a series of state space problems. In order to improve the running time for solving the problems, we parallelize the classification algorithms. We show how to generate successors and how to rank and unrank states on multiple CPU and GPU cores. We provide experimental data for solving the games, discuss the results, and give some final concluding remarks.

2 Preliminaries

In the following, we formalize different characteristics of hash functions.

Definition 1 (Hash Function) *A* hash function h *is a mapping of a universe* U *to an index set* $\{0, \ldots, m-1\}$.

The set of reachable states S of a search problem is a subset of U, i.e., $S \subseteq U$. We are interested in hash functions that are injective[1].

Definition 2 (Perfect Hash Function) *A hash function* $h : S \to \{0, \ldots, m-1\}$ *is* perfect*, if for all* $s \in S$ *with* $h(s) = h(s')$ *we have* $s = s'$.

Definition 3 (Space Efficiency) *The space efficiency of a hash function* $h : S \to \{0, \ldots, m-1\}$ *is the proportion* $m/|S|$ *of available hash values to states.*

Given that every state can be viewed as a bitvector and interpreted as a number, one inefficient design of a perfect hash function is immediate. The space requirements of the corresponding hash table are usually too large. An space-optimal perfect hash function is bijective.

[1] A mapping is injective, if for all $f(x) = f(y)$ we have $x = y$.

Definition 4 (Minimal Perfect Hash Function) *A perfect hash function is* minimal *if its space efficiency is equal to* 1*, i.e., if* $m = |S|$.

Efficient and minimal perfect hash functions allow direct-addressing a bit-state hash table instead of mapping states to an open-addressed or chained hash table. The computed index of the direct access table uniquely identifies the state.

Whenever the averaged number of required bits per state for a perfect hash function is smaller than the number of bits in the state encoding, an implicit representation of the search space is fortunate, assuming that no other tricks like orthogonal hashing are applied.

Definition 5 (Orthogonal Hash Functions) *Two hash functions* h_1 *and* h_2 *are* orthogonal*, if for all states* s, s' *with* $h_1(s) = h_1(s')$ *and* $h_2(s) = h_2(s')$ *we have* $s = s'$.

In case of orthogonal hash functions h_1 and h_2, the value of h_1 can, e.g., be encoded in the file name, leading to a partitioned layout of the search space, and a smaller hash value h_2 to be stored explicitly.

Theorem 1 (Orthogonal Hashing imply Perfect One) *If the two hash functions* $h_1 : S \to \{0, \ldots, m_1 - 1\}$ *and* $h_2 : S \to \{0, \ldots, m_2 - 1\}$ *are orthogonal, their concatenation* (h_1, h_2) *is perfect.*

Proof. We start with two hash functions h_1 and h_2. Let s be any state in S. Given $(h_1(s), h_2(s)) = (h_1'(s), h_2'(s))$ we have $h_1(s) = h_1(s')$ and $h_2(s) = h_2(s')$. Since h_1 and h_2 are orthogonal, this implies $s_1 = s_2$. $\square$

The other important property of a perfect hash function for a state space search is that the state vector can be reconstructed given the hash value.

Definition 6 (Invertible Hash Function) *A perfect hash function* h *is* invertible*, if given* $h(s)$*,* $s \in S$ *can be reconstructed. The inverse* h^{-1} *of* h *is a mapping from* $\{0, \ldots, m - 1\}$ *to* S*. Computing the hash value is denoted as* ranking*, while reconstructing a state given its rank is denoted as* unranking.

For the exploration of the search space, in which array indices serve as state descriptors, invertible hash functions are required.

For the design of minimal perfect hash functions in permutation games, parity will be a helpful concept.

Definition 7 (Inversion) *An* inversion *in a permutation* $\pi = (\pi_1, \ldots, \pi_n)$ *is a pair* (i, j) *with* $1 \leq i < j \leq n$ *and* $\pi_i > \pi_j$.

Definition 8 (Parity) *The* parity *of the permutation* π *is defined as the parity (*mod 2 *value) of the number of inversions in* π*,*

Definition 9 (Parity Preservation) *A permutation game is* parity-preserving*, if all moves preserve the parity of the permutation.*

Parity-preservation allows to separate solvable from insolvable states in several permutation games. If the parity is preserved, the state space can be compressed. For this case we have $|S| = n!/2$.

Definition 10 (Move Alternation Property) *A property $p : S \to \mathbb{N}$ is* move-alternating*, if the parity of p toggles for all actions, i.e., for all s and $s' \in \text{succs}(s)$ we have*

$$p(s') \bmod 2 = (p(s) + 1) \bmod 2.$$

As a result, $p(s)$ is the same for all states s in one BFS layer. In a mixed representation of two subsequent layers, states s' in the next BFS layer can be separated by knowing $p(s') \neq p(s)$.

One example for a move-alternation property is the position of the blank in the Sliding-Tile puzzle. Moreover, many pattern database heuristics [9] have the property either to increase or to decrease by 1 with each applied action.

3 Bitvector State Space Search

As indicated above, perfect hash functions are injective mappings of the set of reachable states to a set of available indices. They are invertible, if the state can be reconstructed given the index. Cooperman and Finkelstein [8] showed that, given a perfect and invertible hash function, two bits per state are sufficient to perform a complete breadth-first exploration of the search space.

3.1 Two-Bit Breadth-First Search

Two-bit breadth-first has first been used to enumerate so-called *Cayley Graphs* [8]. As a subsequent result the authors proved an upper bound to solve every possible configuration of *Rubik's Cube* [23]. By performing a breadth-first search over subsets of configurations in 63 hours together with the help of 128 processor cores and 7 Tera bytes of disk space it was shown that 26 moves always suffice to rescramble it. Korf [20] has applied the two-bit breadth-first search algorithm to generate the state spaces for hard instances of the *Pancake* problem I/O-efficiently.

In the two-bit breadth-first search algorithm (shown in Algorithm 1) every state is expanded at most once. The two bits encode values in $\{0, \ldots, 3\}$ with value 3 representing an unvisited state, and values 0, 1, or 2 denoting the current search depth *mod* 3. This allows to distinguish generated and visited states from ones expanded in the current breadth-first layer.

The running time is determined by the size of the search space times the maximum breadth-first search layer (times the efforts to generate the children).

Algorithm 1 Two-Bit-Breadth-First-Search(m, *init*)

for all $i := 0, \ldots, m-1$ **do**
    BFS-Layer$[i] := 3$
BFS-Layer$[$*rank*$($*init*$)] :=$ *level* $:= 0$
while *BFS-Layer has changed* **do**
    level $:=$ *level* $+ 1$
    for all $i := 0, \ldots, m-1$ **do**
        if *BFS-Layer*$[i] = ($*level* $- 1)$ mod 3 **then**
            succs $:=$ *expand*$($*unrank*$(i))$
            for all $s \in$ *succs* **do**
                if *BFS-Layer*$[$*rank*$(s)] = 3$ **then**
                    BFS-Layer$[$*rank*$(s)] :=$ *level mod* 3

Algorithm 2 One-Bit-Reachability (m, *init*)

for all $i := 0, \ldots, m-1$ **do**
    Open$[i] :=$ **false**
Open$[$*rank*$($*init*$)] =$ **true**
while *Open* **has changed do**
    $i := 0, \ldots, m-1$
    if *Open*$[i] =$ *true* **then**
        succs $:=$ *expand*$($*unrank*$(i))$
        for all $s \in$ *succs* **do**
            Open$[$*rank*$(i)] :=$ **true**

3.2 One-Bit Reachability

Are two bits the best possible compaction for computing the set of all reachable states? Yes and no. The procedure shown in Algorithm 2 illustrates that it is possible to generate the entire state space using one bit per state. However, as it does not distinguish between states to be expanded next (open states) and states already expanded (closed states), the algorithm may expand a state multiple times. Nonetheless, the algorithm is able to determine reachable states. Additional information extracted from a state can improve the running time by decreasing the number of states to be reopened.

If the successor's rank is smaller than the rank of the actual one, it will be expanded in the next scan, otherwise in the same. This observation leads to the following result.

Theorem 2 (Number of Scans in One-Bit Reachability) *The number of scans in the algorithm* One-Bit-Reachability *is bounded by the maximum number of BFS layers.*

Proof. Let *Layer*(i) be the BFS-layer of a state with rank i and *Scan*(i) be the layer in the algorithm *One-Bit-Reachability*. Evidently, *Scan*$($*rank*$($*init*$)) =$ *Layer*$($*rank*$($*init*$)) = 0$. For any path $(s_0, \ldots, s_d)$ generated by BFS, we have *Scan*$($*rank*$(s_{d-1})) \leq$ *Layer*$($*rank*$(s_{d-1}))$ by induction hypothesis. All successors of s_{d-1} are generated in the same iteration (if their index value is larger than i) or in the next iteration (if their index value is smaller than i) such that $Scan(rank(s_d)) \leq Layer(rank(s_d))$. □

3.3 One-Bit Breadth-First Search

For some domains, one bit per state suffices for performing breadth-first search [14]. In *Peg-Solitaire*, the number of remaining pegs uniquely determine the breadth-first search layer, so that one bit per state suffices to distinguish newly generated states from expanded one. This saves space compared to the more general two-bit breadth-first search routine.

In the event of a move-alternation property *alternation*, we, therefore, can perform breadth-first search using only one bit per state.

Algorithm 3 One-Bit-Breadth-First-Search (m, *init*)

for $i = 0, \ldots, m-1$ **do**
  Open$[i]$:= **false**
Open[*rank*(*init*)] := **true**
level := 0
while *Open* **has changed do**
  for all i **with** *Open*$[i]$ = **true do**
    s := *unrank*(i)
    if *alternation*(s) = *level mod* 2 **then**
      succs := *expand*(*unrank*(i))
      for all $s' \in$ *succs* **do**
        Open[*rank*(s')] := **true**
  level = *level* + 1

One important observation is that not all visited states that appear in previous BFS layers are removed from the current search layer. So there are states that are reopened, in the worst case once for each BFS layer. Even though some states may be expanded several times, the following result is immediate.

Theorem 3 (Population Count One-Bit-BFS) *Let the population count pc_l be the number of bits set after the l-th scan in Algorithm* One-Bit-BFS. *Then the number of states in BFS-level l is $|Layer_l| = pc_l - pc_{l-1}$.*

If we were able to store the set of reached states on disk, we could subtract the set of reached states. This, however, would imply that the algorithm no longer consumes one bit per state.

3.4 Two-Bit Retrograde Analysis

Retrograde analysis classifies the entire set of positions in backward direction, starting from won and lost terminal ones. Moreover, partially completed retrograde analyses have been used in conjunction with forward-chaining game playing programs to serve as endgame databases.

Large endgame databases are usually constructed on disk for an increasing number of pieces. Since captures are non-invertible moves, a state to be classified refers only to successors that have the same number of pieces (and thus are in the same layer), and to ones that have a smaller number of pieces (often only one less).

The retrograde analysis algorithm works for all games with this property. In detail: all games, where the game positions can be divided into different layers, and the layers are ordered in such a way that movements are only possible in between a layer or from a higher layer to a lower one.

Additional state information indicating the player to move, retrograde analysis for zero-sum games requires 2 bits per state for executing the analysis on a bitvector representation of the search space: denoting if a state is unsolved, if it is a draw, if it is won for the first, or if it is won for the second player.

Bit-state retrograde analysis applies backward BFS starting from the states that are already decided. Algorithm 4 shows an implementation of the retrograde analysis in pseudo code. For the sake of simplicity, in the implementation we look at two-player zero-sum games that have no draw. (For including draws, we would have to use the unused value 3, which shows, that two bits per state are still sufficient.) Based on the players' turn, the state space is in fact twice as large as the mere number of possible game positions. The bits for the first player and the second player to move are interleaved, so that it can be distinguished by looking at the *mod 2* value of a state's rank.

Under this conditions it is sufficient to do the lookup in the lower layers only once during the computation of each layer. Thus the algorithm is divided into three parts. First an initialization of the layer (lines 4 to 8), here all positions that are won for one of the players are marked, a 1 stands for a victory of player one and a 2 for one of player two. Second a lookup of the successors in the lower layer (lines 9 - 18) is done, and at last an iteration over the remaining unclassified positions is done in lines 19 - 34. In the third part it is sufficient to consider only successors in the same file.

In the second part a position is marked as won if it has a successor that is won for the player to move, here (line 10) `even(i)` checks who is the active player. If there is no winning successor the position remains unsolved. Even if all successors in the lower layer are lost, the position remains unsolved. A position is marked as lost only in the third part of the algorithm, because not until then it is known how all successors are marked. If there are no successors in the third part, then the position is also marked as lost, because it has either only loosing successors in the lower layer, or no successor at all.

In the following it is shown that the algorithm indeed behaves as it is asserted. A winning strategy means that one player can win from a given position no matter how the other player moves.

Theorem 4 *A state is marked as won if and only if there exists a winning strategy for this state.*

A state is marked as lost if and only if it is either a winning situation for the opponent, or all successors are marked as won for the opponent.

Proof. The proof is done with induction over the length of the longest possible path, that is the maximal number of moves to a winning situation. As only two-player zero-sum games are

Algorithm 4 Two-Bit-Retrograde(m, *lost*, *won*)

```
for all i := 0, ..., m − 1 do
   Solved[i] := 0
for all i := 0, ..., m − 1 do
   if won(rank(i)) then
      Solved[i] := 1
   if lost(rank(i)) then
      Solved[i] := 2
   if Solved[i] = 0 then
      succs-smaller := expand-smaller(unrank(i))
      if even(i) then
         for all s ∈ succs-smaller do
            if Solved[rank-smaller(s)] = 2 then
               Solved[rank(i)] := 2
      else
         for all s ∈ succs-smaller do
            if Solved[rank-smaller(s)] = 1 then
               Solved[rank(i)] := 1
while (Solved has changed) do
   for all i := 0, ..., m − 1 do
      if Solved[i] = 0 then
         succs-equal := expand-equal(unrank(i))
         if even(i) then
            allone := true
            for all s ∈ succs-equal do
               if Solved[rank(s)] = 2 then
                  Solved[rank(i)] := 2
               allone := allone & (Solved[rank(s)] = 1)
            if allone then
               Solved[rank(i)] := 1
         else
            alltwo := true
            for all s ∈ succs-equal do
               if Solved[rank(s)] = 1 then
                  Solved[rank(i)] := 1
               alltwo := alltwo & (Solved[rank(s)] = 2)
            if alltwo then
               Solved[rank(i)] := 2
```

considered a game is lost for one player if it is won for the opponent, and as the turns of both players alternate the two statements must be shown together.

The algorithm marks a state with 1 if it assumes it is won for player one and with 2 if it assumes it is won for player two. Initially all positions with a winning situation are marked accordingly, therefore for all paths of length 0 it follows that a position is marked with 1, 2, if and only if it is won for player one, two, respectively. Thus for both directions of the proof the base of the induction holds.

The induction hypothesis for the first direction is as follows:

For all non-final states x with a maximal path length of $n-1$ it follows that:

1. If x is marked as 1 and player one is the player to move, then there exists a winning strategy for player one.

2. If x is marked as 2 and player one is the player to move, then all successors of x are won for player two.

3. If x is marked as 2 and player two is the player to move, then there exists a winning strategy for player two.

4. If x is marked as 1 and player two is the player to move, then all successors of x are won for player one.

Without loss of generality player one is the player to move, the cases for player two are done accordingly.

So assume that x is marked as 1 and the maximal number of moves from position x are n. Then there exists a successor of x, say x', that is also marked as 1. There are two cases how a state can be marked as 1, x' is in a lower layer (lines 17,18)or in the same layer (lines 32,33). In both cases the maximal number of moves from x' is less than n, therefore with the induction hypothesis it follows that all successors of x' are won for player one, therefore there is a winning strategy for player one starting from state x.

Otherwise, if a state x is marked as 2 and the maximal number of moves from position x are n, then there is only one possible way how x was marked by the algorithm (line 34), and it follows that all successors of x are marked with 2, too. Again it follows with the induction hypothesis that there exists a winning strategy for all successors of x, and therefore they are won for player two.

Together the assumption follows.

The other direction is done quite similar, here from a winning strategy it follows almost immediately that a state is marked.

For all paths of length less than n from a state x it follows that:

1. If there exists a winning strategy for player one and player one is the player to move, then x is marked as 1.

2. If all successors of x are won for player two and player one is the player to move, then x is marked as 2.

3. If there exists a winning strategy for player two and player two is the player to move, then x is marked as 2.

4. If all successors of x are won for player one and player two is the player to move, then x is marked as 1.

Assume that the maximal path length from a state x is n. If there exists a winning strategy for player one from x, then this strategy states a successor x' of x such that all successors of x' are won for player one, or x' is a winning situation for player one. In both cases it follows that x' is marked with 1 and therefore x is marked with 1 as well (line 17 or 34).

On the other hand if all successors of x are won for player two. The successors in the lower layer do not effect the value of x because only winning successors change it (lines 16, 17). In line 31 alltwo is set to true and as long as there are only loosing successors which are marked with 2 by induction hypothesis, it stays true, and therefore x is marked with 2 in line 37, too.

□

The algorithm and the theorem can be extended to games with draws by only slightly modifying them, as mentioned above the value 3 can be used to indicate a draw. The problem with a draw is, that it depends on the game how and when it is a draw. Also note that this theorem does not show, that the algorithm always marks all states, because in certain games it is possible to have infinite sequences of moves, and different games have different conditions for this infiniteness; Some leading to draws or preventing cycles by demanding that no game position may occur twice. For these special situations the so called history problem needs to be solved for each game individually.

4 Hashing Permutation Games

In the sequel of this paper, we study efficient perfect hash functions for fast ranking and unranking. We will also look at hash functions that are adapted to the BFS layer.

4.1 Efficient Ranking and Unranking

For ranking and unranking permutations, time- and space-efficient algorithms have already been designed [2].

Definition 11 (Natural or Lexicographic Rank) *The natural or lexicographic rank of a permutation is the position in the lexicographic order of its state vector representation. In the lexicographic ordering of a permutation* $\pi = (\pi_0, \ldots, \pi_{n-1})$ *of* $\{0, \ldots, n-1\}$ *we first have* $(n! - 1)$ *permutations that begin with* 0*, followed by* $(n! - 1)$ *permutations that begin with* 1*, etc. Threfore, we have*

$$\pi_0 \cdot (n-1)! \leq \text{lex-rank}(\pi, n) \leq \pi_0 \cdot (n-1)!.$$

This leads to the following recursive formula: $\text{lex-rank}((0), 1) = 0$ *and*

$$\text{lex-rank}(\pi, n) \leq \pi_0 \cdot (n-1)! + \text{lex-rank}(\pi', n-1),$$

where $\pi'_i = \pi_{i+1}$ *if* $\pi'_i > \pi_0$ *and* $\pi'_i = \pi_i$ *if* $\pi'_i < \pi_0$.

The following result is widely known [2].

Theorem 5 (Inverted Index, Factorial Base) *The lexicographic rank of permutation π (of size n) is determined as lex-rank$(\pi, n) = \sum_{i=0}^{N-1} d_i \cdot (N-1-i)!$ where the vector d of coefficients d_i is called the* inverted index *or* factorial base. *The coefficients d_i are uniquely determined. The parity of a permutation is known to match* $(\sum_{i=0}^{N-1} d_i) \bmod 2$.

In the recursive definition of *lex-rank* the derivation of π' from π creates a burden that makes an according ranking algorithm non-linear. There have been many attempts, e.g. by Trotter and Johnson's minmal-exchange approach, which still have a non-linear time complexity in the worst-case [22].

Korf and Schultze [21] use two lookup tables with a space requirement of $O(2^n \log n)$ bits to compute lexicographic ranks in linear time. More crucially, given that larger tables do not fit into SRAM, the algorithms does not work well on the GPU. Bonet [2] discusses time-space trade-offs and provides a uniform algorithm that takes $O(n \log n)$ time and $O(n)$ space. Algorithms that are linear in time and space for both operations are not known.

Given that existing ranking and unranking algorithms wrt. the lexicographic ordering are rather slow in particular if executed on the graphics card, next we have a detailed look at the more efficient ordering of Myrvold and Ruskey [24]. They devise another ordering based on the observation that every permutation can be generated uniformly by swapping an element at position i with a randomly selected element $j > i$, while i continously increases. The sequence of j's can be seen as the equivalent to the factorial base for the lexicographic rank.

We show that the parity of a permuation can be derived on-the-fly in the unranking algorithm proposed by Myrvold and Ruskey[2]. For fast execution on the graphics card, we additionally avoid recursion.

The ranking algorithm is shown in Algorithm 5. The input is the number of elements N to permute, the permutation π, and its inverse permutation π^{-1}. The output is the rank of π. As a side effect, we have that both π and π^{-1} are modified. The unranking algorithm is shown in Alg. 6.

Algorithm 5 *rank*(n, π, π^{-1})

```
1: for all i in {1, ..., n-1} do
2:     l ← π_{n-i}
3:     swap(π_{n-i}, π_{π^{-1}_{n-i}})
4:     swap(π^{-1}_l, π^{-1}_{n-i})
5:     rank_i ← l
6: return ∏_{i=1}^{n-1} (rank_{n-i+1} + i)
```

Theorem 6 (Parity in Myrvold & Ruskey's Unrank Function) *The parity of a permutation for a rank r in Myrvold & Ruskey's permutation ordering can be computed on-the fly with the unrank function shown in Algorithm 6.*

[2]In all our results, we refer to Myrvold and Ruskey's rank1 and unrank1 functions.

Algorithm 6 *unrank*(r)

```
π := id
parity := false
while n > 0 do
    i := n − 1
    j := r mod n
    if i ≠ j then
        parity := ¬parity
        swap(π_i, π_j)
        r := r div n
    n := n − 1
return (parity, π)
```

Proof. In the *unrank* function swapping two elements u and v at position i and j, resp., with $i \neq j$ we count $2(j-i-1)+1$ transpositions (u and v are the elements to be swapped, x is a wildcard for any intermediate element): $uxx\ldots xxv \rightarrow xux\ldots xxv \rightarrow \ldots \rightarrow xx\ldots xxuv \rightarrow xx\ldots xxvu \rightarrow \ldots \rightarrow vxx\ldots xxu$. As $2(j-i-1)+1$ *mod* $2 = 1$, each transposition either increases or decreases the parity of the number of inversions, so that the parity toggles for each iteration. The only exception is if $i = j$, where no change occurs. Hence, the parity of the permutation can be determined on-the-fly in our algorithm.

□

Theorem 7 (Folding Permutation Table in Myrvold & Ruskey's Approach) *Let $\pi(r)$ denote the permutation returned by Myrvold & Ruskey's* unrank *function given index r. Then $\pi(r)$ matches $\pi(r + n!/2)$ except for swapping π_0 and π_1.*

Proof. The last call to *swap*$(\pi_{n-1}, \pi_{r \textit{ mod } n})$ in Myrvold and Ruskey's *unrank* function is *swap*$(\pi_0, \pi_{r \textit{ mod } 1})$, which resolves to either *swap*(π_1, π_1) or *swap*(π_1, π_0). Only the latter one induces a change.

If $r_1, \ldots, r_{n-1}$ denote the indices of r *mod* n in the iterations $1, \ldots, N-1$ of Myrvold and Ruskey's *unrank* function, then $r_{N-1} = \lfloor \ldots \lfloor r/(n-1) \rfloor \ldots /2 \rfloor$, which resolves to 1 for $r \geq n!/2$ and 0 for $r < n!/2$. □

4.2 Sliding-Tile Puzzle

Next, we consider permutation games, especially the ones shown in Fig. 1.

The $(n \times m)$ sliding-tile puzzle [17] consists of $(nm-1)$ numbered tiles and one empty position, called the blank. In many cases, the tiles are squarely arranged, such that $m = n$.

The task is to re-arrange the tiles such that a certain goal arrangement is reached. Swapping two tiles toggles the permutation parity and, in turn, the solvability status of the game. Thus, only half the $nm!$ states are reachable.

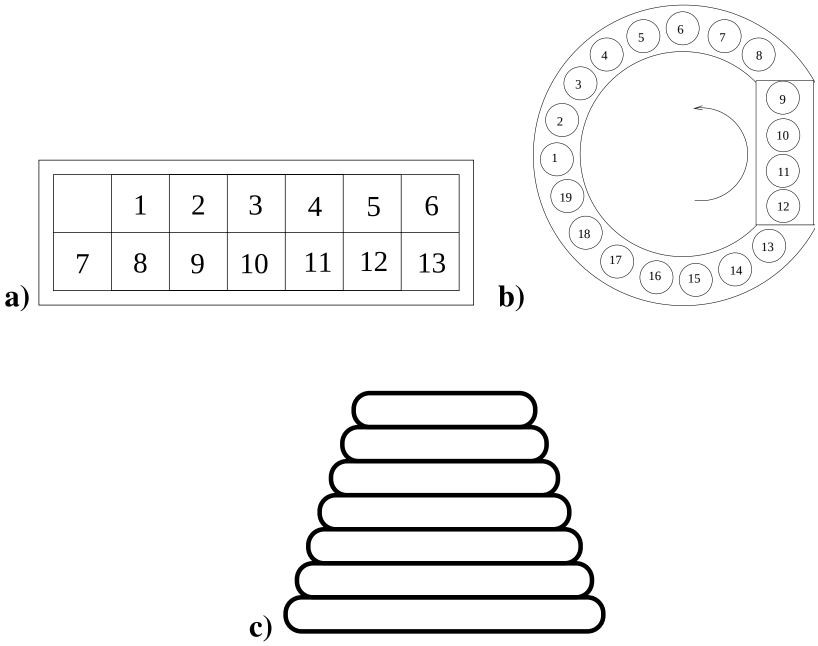


Figure 1: Permutation Games: a) Sliding-Tile Puzzle, b) Top-Spin Puzzle, c) Pancake Problem.

For the Sliding-Tile puzzle, we observe that in a lexicographic ordering every two adjacent permutations with lexicographic rank $2i$ and $2i + 1$ have a different solvability status. In order to hash a sliding-tile puzzle state to $\{0, \ldots, (nm)!/2 - 1\}$, we can, therefore, compute the lexicographic rank and divide it by 2. Unranking is slightly more complex, as it has to determine, which of the two permutations π_{2i} and π_{2i+1} of the puzzle with index i is actually reachable.

There is one subtle problem with the blank. Simply taking the parity of the entire board does not suffice to compute a minimum perfect hash value in $\{0, \ldots, nm!/2\}$, as swapping a tile with the blank is a move, which does not change the parity.

A solution to this problem (shown in Algorithm 7) is to partition the state space wrt. the position of the blank, since for exploring the $(n \times m)$ puzzle it is equivalent to enumerate all $(nm - 1)!/2$ orderings together with the nm positions of the blank. If $S_0, \ldots, S_{nm-1}$ denote the set of "blank-projected" partitions, then each set S_j, $j \in \{0, \ldots, nm - 1\}$ contains $(nm - 1)!/2$ states. Given the index i as the permutation rank and j it is simple to reconstruct the puzzle's state.

As a side effect of this partitioning, horizontal moves of the blank do not change the state vector, thus the rank remains the same. Tiles remain in the same order, preserving the rank.

Since the parity does not change in this puzzle we need another move alternating property, and find it in the position of the blank. The partition into buckets $S_0, \ldots, S_{nm-1}$ has the additional advantage that we can determine, whether the state belongs to an odd or even layer and which bucket a successor belongs to [27]. We observe that in puzzles with an odd number of columns at an even breadth-first level the blank position is even and at an odd breadth-first level the blank

Algorithm 7 One-Bit-Breath-First-Search-Sliding-Tile (*init*)

1: **for** $\mathit{blank} = 0, \ldots, nm-1$ **do**
2:   **for** $i = 0, \ldots, (nm-1)!/2 - 1$ **do**
3:     $\mathit{Open}[\mathit{blank}][i] :=$ **false**
4: $\mathit{Open}[\mathit{blank}(\mathit{init})][\mathit{rank}(\mathit{init})$ **mod** $(nm-1)!/2] :=$ **true**
5: $\mathit{level} := 0$
6: **while** Open **has changed do**
7:   $\mathit{blank} := \mathit{level}$ **mod** 2
8:   **while** $\mathit{blank} \leq nm$ **do**
9:     **for all** $d \in \{R, L, D, U\}$ **do**
10:       $\mathit{dst} := \mathit{newblank}(\mathit{blank}, d)$
11:       **if** $d \in \{L, R\}$ **then**
12:         $\mathit{Open}[\mathit{dst}] := \mathit{Open}[\mathit{dst}] \ \vee \ \mathit{Open}[\mathit{blank}]$
13:       **else**
14:         **for all** i **with** $\mathit{Open}[\mathit{blank}][i] =$ **true do**
15:           $(\mathit{valid}, \pi) := \mathit{unrank}(i)$
16:           **if** $\neg\mathit{valid}$ **then**
17:             $\mathit{swap}(\pi_0, \pi_1)$
18:           $\mathit{succ} := \mathit{expand}(\pi, d)$
19:           $r := \mathit{rank}(\mathit{succ})$ **mod** $(N-1)!/2$
20:           $\mathit{Open}[\mathit{dst}][r] :=$ **true**
21:     $\mathit{blank} = \mathit{blank} + 2$
22:   $\mathit{level} = \mathit{level} + 1$

position is odd.

For such a factored representation of the sliding-tile puzzles, a refined exploration in Algorithm 3 retains the breadth-first order, by means that a bit for a node is set for the first time in its BFS layer. The bitvector *Open* is partitioned into nm parts, which are expanded depending on the breadth-first *level* (line 7).

As mentioned above, the rank of a permutation does not change by a horizontal move of the blank. This is exploited in line 11 writing the ranks directly to the destination bucket using a bitwise-or on the bitvector from layer $\mathit{level} - 2$ and *level*. The vertical moves are unranked, moved and ranked from line 13 onwards. When a bucket is done, the next one is skipped and the next but one is expanded. The algorithm terminates when no new successor is generated.

4.3 Top-Spin Puzzle

The next example is the (n, k)-Top-Spin Puzzle [7], which has n tokens in a ring. In one twist action k consecutive tokens are reversed and in one slide action pieces are shifted around. There are $n!$ different possible ways to permute the tokens into the locations. However, since the puzzle is cyclic only the order of the different tokens matters and thus there are only $(n-1)!$ different states in practice. After each of the n possible actions, we thus normalize the permutation by cyclically shifting the array until token 1 occupies the first position in the array.

Theorem 8 (Parity in Top-Spin Puzzle) *For an even value of k (the default) and odd value of $n > k + 1$, the (normalized) (n, k) Top-Spin Puzzle has $(n - 1)!/2$ reachable states.*

Proof. We first observe that due to the normalization for an even value of k, only a twist at the start/end of the normalized array can change the parity. Otherwise, the twist involves reversing k adjacent numbers, an operation with even parity.

Let $n = 2m + 1$ and $(x_0, x_1, \ldots, x_{2m})$ be the normalized state vector. Thus, due to normalization, $x_0 = 0$.

First of all, we observe that the modification of 0 is not counted as a transposition in the normalized representation, so only $k - 1$ elements actually change their relative position and lead to an odd number of transpositions.

Without loss of generality, we look at $k = 4$, which simplifies notation. Larger values of k only increase the number of cases, but lead to no further insight. Assuming $k = 4$, three elements change their relative position and lead to three transpositions.

We now look at the effect of normalization. For $(0, x_1, x_2, x_3, \ldots, x_{2m})$ we have four critical successors:

- $(x_3, x_2, x_1, 0, x_4....x_{2m})$,
- $(x_2, x_1, 0, x_{2m}, x_3, ..., x_{2m-1})$,
- $(x_1, 0, x_{2m-1}, x_{2m}, x_2, ..., x_{2m-2})$, and
- $(0, x_{2m-2}, x_{2m-1}, x_{2m}, x_1, ..., x_{2m-3})$.

In all cases, normalization has to move 3 elements either the ones with low index to the end of the array to postprocess the twist, or the ones with large indices to the start of the array to preprocess the operation. The number of transpositions for one such move is $2m - 1$. In total we have $3(2m - 1) + 3$ transpositions. As each transposition changes the parity and the total of $6m$ transpositions is even, all critical cases have even priority. □

As the parity is even for a move in the (normalized) (n, k) Top-Spin Puzzle for an odd value of $n > k + 1$, we obtain the entire set of $(n - 1)!$ reachable states.

4.4 Pancake Problem

The n-Pancake Problem [10] is to determine the number of flips of the first k pancakes (with varying $k \in \{1, \ldots, n\}$) necessary to put them into ascending order. The problem has been analyzed e.g. by [16]. It is known that $(5n + 5)/3$ flips always suffice, and that $15n/14$ flips are necessary.

In the *n-Burned-Pancake* variant, the pancakes are burned on one side and the additional requirement is to bring all burned sides down. For this version it is known that $2n - 2$ flips always suffice and that $3n/2$ flips are necessary. Both problems have n possible operators. The pancake problem has $n!$ reachable states, the burned one has $n!2^n$ reachable states. For an even value of $\lceil (k-1)/2 \rceil$, $k > 1$ the parity changes, while for an odd one, the parity remains the same.

5 Hashing Selection Games

Figure 2: Initial States of the Two-Player Turn-Taking Game *Fox-and-Geese*.

Fox-and-Geese is a two-player zero-sum game. The lone fox attempts to capture the geese, while the geese try to hem the Fox, so that he can't move. It is played upon a cross-shaped board consisting of a 3×3 square of intersections in the middle with four 2×3 areas adjacent to each face of the central square. One board with the initial layout is shown in Fig. 2. Pieces can move to any empty intersection around them (also diagonally). The fox can additionally jump over a goose to capture it. Geese cannot jump. The geese win if they surround the fox so that it cannot move. The fox wins if it captures enough geese that the remaining geese cannot surround him.

Fox-and-Geese belongs to the set of asymmetric strategy games played on a cross shaped board. The first probable reference to an ancestor of the game is that of *Hala-Tafl*, which is mentioned in an Icelandic saga and which is believed to have been written in the 14th century[3]. To the authors' knowledge, *Fox-and-Geese* has not been solved yet. The chances for 13 geese are assumed to be an advantage for the fox, while for 17 geese the chances are assumed to be roughly equal.

The game requires a strategic plan and tactical skills in certain battle situations. The portions of tactic and strategy are not equal for both players, such that a novice often plays better with the fox than with the geese. A good fox detects weaknesses in the set of goose (unprotected ones, empty vertices, which are central to the area around) and moves actively towards them.

[3] see *The Online Guide to Traditional Games*

Potential decoys, which try to lure the fox out of his burrow have to be captured early enough. The geese have to work together in form of a swarm and find a compromise between risk and safety. In the beginning it is recommended to choose safe moves, while to the end of the game it is recommended to challenge the fox to move out in order to fill blocked vertices.

Figure 3: Initial State of the Single-Player Game *Peg-Solitaire*.

Fox-and-Geese extends *Peg-Solitaire* (see Fig. 3), a single-agent problem invented in the 17th century. The game asks for the minimum number of pegs that is reachable from a given initial state. The set of pegs is iteratively reduced by jumps. Solutions for the initial state (shown in Fig. 3) with one peg remaining in the middle of the board are widely known [1]. An optimal player for all possible states has been generated by [12].

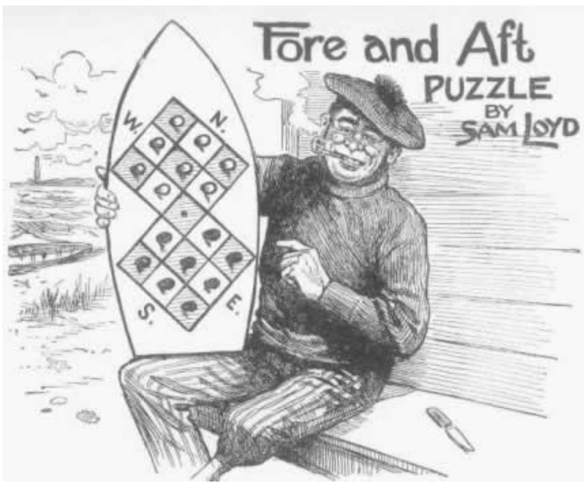


Figure 4: Initial State of the Single-Player Game *Fore and Aft*.

The *Fore and Aft* puzzle (see Fig. 4) has been made popular by the American puzzle creator Sam Loyd. It is played on a part of the 5×5 board consisting of two 3×3 subarrays at diagonally opposite corners. They overlap in the central square. One square has 8 black pieces and the other has 8 white pieces, with the centre left vacant. The objective is to reverse the positions of pieces in the lowest number of moves. Pieces can slide or jump over another pieces of any colour. This game was originally an English invention, having been designed by an English sailor in the 18th century. Henry Ernest Dudeney discovered a quickest solution of just 46 moves. *Frogs-and-Toads* generalizes *Fore and Aft* and larger versions are yet unsolved.

As the number of pegs shows the progress in playing the game *Peg-Solitaire*, we may aim at representing all boards with k of the $n-1$ possible pegs, where n is the number of holes. In fact, the breadth-first level k contains at most $\binom{n}{k}$ states. In contrast to permutation games, pegs are indistinguishable, and call for a different design of a hash function and its inverse.

Such an invertible perfect hash function of all states that have $k = 1, \ldots, n$ pegs remaining on the board reduces the RAM requirements for analyzing the game. As successor generation is fast, we will need an efficient hash function (rank) that maps bitvectors $(s_0, \ldots, s_{n-1}) \in \{0,1\}^n$ with k ones to $\{0, \ldots, \binom{n}{k} - 1\}$ and back (unrank). There is a trivial ranking algorithm that uses a counter to determine the number of bitvectors passed in their lexicographic ordering that have k ones. It uses linear space, but the time complexity by traversing the entire set of bitvectors is exponential. The unranking algorithm works similarly with matching exponential time performance.

The design of a linear time ranking and unranking algorithm is not obvious. The pieces on the board are not labeled, their relative ordering does not matter.

5.1 Hashing with Binomial Coefficients

An efficient solution for perfect and invertible hashing of all bitvectors with k ones to $\{0, \ldots, \binom{n}{k} - 1\}$ is shown in Algorithms 8 and 9. The algorithms utilize binomial coefficients that can either be precomputed or determined on-the-fly. The algorithms rely on the observation that once a bit at position i in a bitvector with n bits and with j zeros is processed, the binomial coefficient $\binom{i}{j-1}$ can be added to the rank. The notation $\max\left\{0, \binom{i}{zeros-1}\right\}$ is shorthand notation to say, if $zeros < 1$ take 0, otherwise take $\binom{i}{zeros-1}$.

The time complexities of both algorithms are $O(n)$. In case the number of zeros exceeds the number of ones, the rank and unrank algorithms can be extended to the inverted bitvector representation of a state.

The correctness argument is based on representing of the binomial coefficients in a grid graph of nodes $B_{i,j}$ with i denoting the position in the bit-vector and j denoting the number of zeros already seen. Let $B_{i,j}$ be connected via a directed edge to $B_{i-1,j}$ and $B_{i-1,j-1}$ corresponding to a zero and an one processed in the bit-vector. Starting at $B_{i,j}$ there are $\binom{i}{j}$ possible non-overlapping paths that reach $B_{0,z}$. These pathcount-values can be used to determine the index of a given bitvector in the set of all possible ones. At the current node (i, j) in the grid graph in case of the state at position i containing a

- 1: all path-counts at $B_{i-1,j-1}$ are added.

Algorithm 8 Ranking-Binomial-Coefficients(s, *ones*)

$r := 0; i := n; \mathit{zeros} := n - \mathit{ones}$
while $(i > 0)$ **do**
  $i := i - 1$
  if $(s[i] = 0)$ **then**
    $\mathit{zeros} := \mathit{zeros} - 1$
  else
    $\mathit{value}_0 := \max\left\{0, \binom{i}{\mathit{zeros}-1}\right\}$
    $r := r + \mathit{value}_0$
return r

Algorithm 9 Unranking-Binomial-Coefficients(r, n, *ones*)

$i := n; \mathit{zeros} := n - \mathit{ones}$
while $(i > 0)$ **do**
  $i := i - 1$
  $\mathit{value}_0 := \max\left\{0, \binom{i}{\mathit{zeros}-1}\right\}$
  if $(r < \mathit{value}_0)$ **and** $(\mathit{zeros} > 0)$ **then**
    $\mathit{zeros} := \mathit{zeros} - 1$
    record $s[i] := 0$
  else
    record $s[i] := 1$
    $r := r - \mathit{value}_0$
return s

- 0: nothing is added.

5.2 Hashing with Multinomial Coefficients

The perfect hash functions derived for games like *Peg-Solitaire* are often insufficient in games with pieces of different color like *Tic-Tac-Toe* and *Nine-Men-Morris*. For this case, we have to devise a hash function that operates on state vectors of size n that contain zeros (location not occupied), ones (location occupied by pieces of the first player) and twos (location occupied by pieces of the second player). We will determine the value of a position by hashing all state with a fix number of z zeros, and o ones and $t = n - z - o$ twos to a value in $\{0, \ldots, \binom{n}{z,o,t} - 1\}$, where the multinomial coefficient $\binom{n}{z,o,t}$ is defined as

$$\binom{n}{z, o, t} = \frac{n!}{z! \cdot o! \cdot t!}.$$

The implementations of the rank and unrank functions are shown in Algorithms 10 and 11. They naturally extend the code derived for binomial coefficients.

Algorithm 10 Ranking-Multinomial-Coefficients($s, n, \mathit{ones}, \mathit{twos}$)

$r := 0$; $\mathit{zeros} := n - \mathit{ones} - \mathit{twos}$; $i := n$
while $(i > 0)$ **do**
  $i := i - 1$
  if $(s[i] = 0)$ **then**
    $\mathit{zeros} := \mathit{zeros} - 1$
  else
    if $s[i] = 1$ **then**
      $\mathit{value}_0 := \max\left\{0, \binom{i}{\mathit{zeros}-1, \mathit{ones}, i-\mathit{zeros}-\mathit{ones}-1}\right\}$
      $r := r + \mathit{value}_0$
      $\mathit{ones} := \mathit{ones} - 1$
    else
      $\mathit{value}_0 := \max\left\{0, \binom{i}{\mathit{zeros}-1, \mathit{ones}, i-\mathit{zeros}-\mathit{ones}-1}\right\}$
      $\mathit{value}_1 := \max\left\{0, \binom{i}{\mathit{zeros}, \mathit{ones}-1, i-\mathit{zeros}-\mathit{ones}-1}\right\}$
      $r := r + \mathit{value}_0 + \mathit{value}_1$
return r

The correctness argument relies on representing the multinomial coefficients in a 3D grid graph of nodes $B_{i,j,l}$ with i denoting the index position in the vector and j denoting the number of zeros j, and l denoting the number of ones already seen. The number of twos is then immediate. Let $B_{i,j,l}$ be connected via a directed edge to $B_{i-1,j,l}$, $B_{i-1,j,l-1}$ and $B_{i-1,j-1,l}$ corresponding to a value 2, 1 or 0 processed in the bit-vector, respectively. There are $\binom{i}{j,l,n-j-l}$ possible non-overlapping paths starting from each node $B_{i,j,l}$ that reach $B_{0,z,o}$. These pathcount-values can be used to determine the index of a given bitvector in the set of all possible ones. At the current node (i, j, l) in the grid graph in case of the node at position i containing a

- 1: all path-counts values at $B_{i-1,j-1,l}$ are added.
- 2: all path-counts values at $B_{i-1,j,l-1}$ are added.
- 0: nothing is added.

6 Parallelization

Parallel processing is the future of computing. On current personal computer systems with multiple cores on the CPU and (graphics) processing units on the graphics card, parallelism is available "for the masses". For our case of solving games, we aim at fast successor computation. Moreover, ranking and unranking take substantial running time are executed in parallel.

To improve the I/O behavior the partitioned state space was distributed over multiple hard disks. This increased the reading and writing bandwidth and to enable each thread to use its own hard disk. In larger instances that exceed RAM capacities we additionally maintain write buffers

Algorithm 11 Unranking-Multinomial-Coefficients$(r, n, \mathit{ones}, \mathit{twos})$

$i := n$; $\mathit{zeros} := n - \mathit{ones} - \mathit{twos}$
while $i > 0$ **do**
  $i := i - 1$
  $\mathit{value}_0 := \max\left\{0, \binom{i}{\mathit{zeros}-1, \mathit{ones}, i-\mathit{zero}-\mathit{ones}-1}\right\}$
  $\mathit{value}_1 := \max\left\{0, \binom{i}{\mathit{zeros}, \mathit{ones}-1, i-\mathit{zero}-\mathit{ones}-1}\right\}$
  if $(r < \mathit{value}_0)$ and $(\mathit{zeros} > 0)$ **then**
    $\mathit{zeros} := \mathit{zeros} - 1$
    record $s[i] := 0$
  else
    if $(r < \mathit{value}_0 + \mathit{value}_1)$ and $(\mathit{ones} > 0)$ **then**
      $\mathit{ones} := \mathit{ones} - 1$
      record $s[i] := 1$
      $r := r - \mathit{value}_0$
    else
      record $s[i] := 2$
      $\mathit{twos} := \mathit{twos} - 1$
      $r := r - \mathit{value}_0 - \mathit{value}_1$
return s

to avoid random access on disk. Once the buffer is full, it is flushed to disk. In one streamed access, all corresponding bits are set.

6.1 Multi-Core Computation

Nowadays computers have multiple cores, which reduce the run-time of an algorithm via distribution the workload to concurrently running threads.

We use *pthreads* as additional multi-threading support.

Let S_p be the set of all possible positions in *Fox-and-Geese* (*Frogs-and-Toads*) with p pieces, which together with the fox (blank) position and the player's turn uniquely address states in the game. During play, the number of pieces decreases (or stays) such that we partition backward (forward) BFS layers into disjoint sets $S_p = S_{p,0} \cup \ldots \cup S_{p,n-1}$. As $|S_{p,i}| \leq \binom{n-1}{p}$ is constant for all $i \in \{0, \ldots, n-1\}$, a possible upper bound on the number of reachable states with p pieces is $n \cdot \binom{n-1}{p}$. These states will be classified by our algorithm.

In two-bit retrograde (bfs) analysis all layers $\mathit{Layer}_0, \mathit{Layer}_1, \ldots$ are processed in partition form. The fixpoint iteration to determine the solvability status in one backward (forward) BFS level $\mathit{Layer}_p = S_{p,0} \cup \ldots \cup S_{p,n-1}$ is the most time consuming part. Here, we can apply a multi-core parallelization using pthreads. In total, n threads are forked and joined after completion. They share the same hash function, and communicate for termination.

For improving space consumption we urge the exploration to flush the sets $S_{p,i}$ whenever

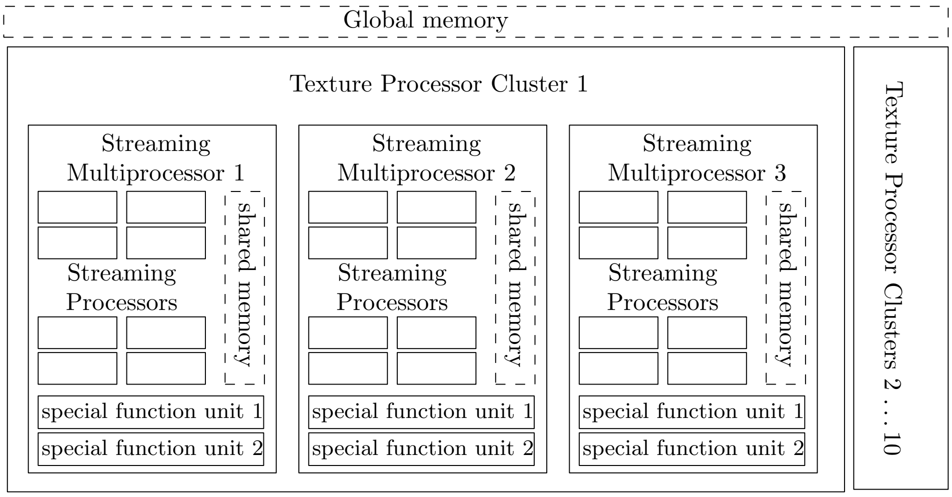


Figure 5: Sample GPU Architecture (G200 Chipset).

possible and to load only the ones needed for the current computation. In the retrograde analysis of *Fox-and-Geese* the access to positions with a smaller number of pieces S_{p-1} is only needed during the initialization phase. As such initialization is a simple scan through a level we only need one set $S_{p,i}$ at a time. To save space for the fixpoint iteration, we release the memory needed to store the previous layer. As a result, the maximum number of bits needed is $\max\{|S_p|, |S_p|/n + |S_{p-1}|)$.

6.2 GPU Computation

In the last few years there has been a remarkable increase in the performance and capabilities of the graphics processing unit. Modern GPUs are not only powerful, but also parallel programmable processors featuring high arithmetic capabilities and memory bandwidths. Deployed on current graphic cards, GPUs have outpaced CPUs in many numerical algorithms. The GPU's rapid increase in both programmability and capability has inspired researchers to map computationally demanding, complex problems to the GPU.

GPUs have multiple cores, but the programming and computational model are different from the ones on the CPU. Programming a GPU requires a special compiler, which translates the code to native GPU instructions. The GPU architecture mimics a single instruction multiply data (SIMD) computer with the same instructions running on all processors. It supports different layers for memory access, forbids simultaneous writes but allows concurrent reads to one memory cell.

If we consider the G200 chipset, as found in state-of-the-art NVIDIA GPUs and illustrated in Figure 5, a core is a streaming processor (SP) with 1 floating point and 2 arithmetic logic units. 8 SPs are grouped together with a cache structure and two special function units (performing e.g. double precision arithmetics) to one streaming multiprocessor (SM), and used like ordinary SIMD processors. Each of the 10 texture processor clusters (TSCs) combines 3 SMs, yielding

240 cores in one chip.

Memory, visualized shaded in the figure, is structured hierarchically, starting with the GPU's global memory (video RAM, or VRAM). Access to this memory is slow, but can be accelerated through *coalescing*, where adjacent accesses with less than 64 bits are combined to one 64-bit access. Each SM includes 16 KB of memory (SRAM), which is shared between all SPs and can be accessed at the same speed as registers. Additional registers are also located in each SM but not shared between SPs. Data has to be copied from the systems main memory to the VRAM to be accessible by the threads.

The GPU programming language links to ordinary C-sources. The function executed in parallel on the GPU is called *kernel*. The kernel is driven by threads, grouped together in *blocks*. The TSC distributes the blocks to its SMs in a way that none of them runs more than $1,024$ threads and a block is not distributed among different SMs. This way, taking into account that the maximal *blockSize* of 512, at most 2 blocks can be executed by one SM on its 8 SPs. Each SM schedules 8 threads (one for each SP) to be executed in parallel, providing the code to the SPs. Since all the SPs get the same chunk of code, SPs in an else-branch wait for the SPs in the if-branch, being idle. After the 8 threads have completed a chunk the next one is executed. Note that threads waiting for data can be parked by the SM, while the SPs work on threads, which have already received the data.

To profit from coalescing, threads should access adjacent memory contemporary. Additionally, the SIMD like architecture forces to avoid if-branches and to design a kernel which will be executed unchanged for all threads. This facts lead to the implementation of keeping the entire or partitioned state space bitvector in RAM and copying an array of indices (ranks) to the GPU. This approach benefits from the SIMD technology but imposes additional work on the CPU. One additional scan through the bitvector is needed to convert its bits into integer ranks, but on the GPU the work to unrank, generate the successors and rank them is identical for all threads. To avoid unnecessary memory access, the rank given to expand should be overwritten with the rank of the first child. As the number of successors is known beforehand, with each rank we reserve space for its successors. For smaller BFS layers this means that a smaller amount of states is expanded.

For solving games on the graphics card [14], storing the bitvector on the GPU yields bad exploration results. Hence, we forward the bitvector indices from the CPU's host RAM to the GPU's VRAM, where they were uploaded to the SRAM, unranked and expanded, while the successors were ranked. At the end of one iteration, all successors are moved back to CPU's host RAM, where they are perfectly hashed and marked if new.

7 Experiments

We divide the presentation of the experiments in permutation games (mostly showing the effect of multi-core GPU computation) and selection games (also showing the effect of multi-core CPU computation).

7.1 Permutation Games

We conducted the permutation game experiments on an AMD Athlon 64 X2 Dual Core Processor 3800+ system with 2 GB RAM. The GPU we used was an NVIDIA graphics card with G-200 chipset and 1 GB VRAM. In all cases we perform forward breadth-first search to generate the entire state space.

For measuring the speed-up on a matching implementation we compare the GPU performance with a CPU emulation on a single core. This way, the same code and work was executed on the CPU and the GPU. For a fair comparison, the emulation was run with GPU code adjusted to one thread. This minimizes the work for thread communication on the CPU. Moreover, we profiled that the emulation consumed most CPU time for state expansion and ranking.

Sliding-Tile Puzzle The results of the first set of experiments shown in Table 1 illustrate the effect of bitvector state space compression with breadth-first search in rectangular *Sliding-Tile* problems of different sizes.

We run both the one- and two-bit breadth-first search algorithms on the CPU and GPU. The 3×3 version was simply too small to show significant advances, while even in partitioned form a complete exploration on a bit vector representation of the 15-Puzzle requires more RAM than available.

We first validated that all states were generated and equally distributed among the possible blank positions. Moreover, as expected, the numbers of BFS layers for symmetric puzzle instances match (53 for 3×4 and 4×3 as well as 63 for 2×6 and 6×2).

For the 2-Bit BFS implementation, we observe a moderate speed-up by a factor between 2 and 3, which is due to the fact that the BFS-layers of the instances that could be solved in RAM are too small. For such small BFS layers, further data processing issues like copying the indices to the VRAM is rather expensive compared to the gain achieved by parallel computation on the GPU. Unfortunately, the next larger instance (7×2) was too large for the amount of RAM available in the machine (it needs $3 \times 750 = 2,250$ MB for *Open* and 2 GB for reading and writing indices to the VRAM).

In the 1-Bit BFS implementation the speed-up increases to a factor between 7 and 10 in the small instances. Many states are re-expanded in this approach, inducing more work for the GPU and exploiting its potential for parallel computation. Partitions being too large for the VRAM are split and processed in chunks of about 250 millions indices (for the 7×2 instance). A quick calculation shows that the savings of GPU computation are large. We noticed that the GPU has the capability to generate 83 million states per second (including unranking, generating the successors and computing their rank) compared to about 5 million states per second of the CPU. As a result, for the CPU experiment that ran out of time (o.o.t), which we stopped after one day of execution, we predict a speed-up factor of at least 16, and a running time of over 60 hours.

Top-Spin Problems The results for the (n, k)-Top-Spin problems for a fixed value of $k = 4$ are shown in Table 7.1 (o.o.m denotes out of memory, while o.o.t denotes out of time). We see that the experiments validate the theoretical statement of Theorem 1 that the state spaces are of

Problem	2-Bit Time GPU	2-Bit Time CPU	1-Bit Time GPU	1-Bit Time CPU
(2×6)	1m10s	2m56s	2m43s	15m17s
(3×4)	55s	2m22s	1m38s	13m53s
(4×3)	1m4s	2m22s	1m44s	12m53s
(6×2)	1m26s	2m40s	1m29s	18m30s
(7×2)	o.o.m.	o.o.m.	226m30s	o.o.t.

Table 1: Comparing GPU with CPU Performances in 1-Bit and 2-Bit BFS in the Sliding-Tile Puzzle Domain.

n	States	GPU Time	CPU Time
6	120	0s	0s
7	360	0s	0s
8	5,040	0s	0s
9	20,160	0s	0s
10	362,880	0s	6s
11	1,814,400	1s	35s
12	39,916,800	27s	15m20s

Table 2: Comparing GPU with CPU Performances for Two-Bit-BFS in the Top-Spin Domain.

size $(n-1)!/2$ for n being odd[4] and $(n-1)!$ for n even. For large values of n, we obtain a significant speed-up of more than factor 30.

Pancake Problems The GPU and CPU running time results for the n-Pancake problems are shown in Table 7.1. Similar to the Top-Spin puzzle for a large value of n, we obtain a speed-up factor of more than 30 wrt. running the same algorithm on the CPU.

7.2 Selection Games

Experiments are drawn on a Linux-PC with an Intel i7 processor having 8 cores running at 2.66 Ghz. The computer is equipped with 12 GB RAM. It has a NVIDIA graphics card with G-200 Chipset and 1GB VRAM.

Peg-Solitaire The first set of results, shown in Table 4, considers *Peg-Solitaire*. For each BFS-layer, the state space is small enough to fit in RAM. The exploration result show that there are 5 positions with one peg remaining (of course there is none with zero pegs), one of which has the peg in the goal position.

[4]At least the Top-Spin implementation of Rob Holte and likely the one of Ariel Felner/Uzi Zahavi do not consider parity compressed state spaces.

n	States	GPU Time	CPU Time
9	362,880	0s	4s
10	3,628,800	2s	48s
11	39,916,800	21s	10m41s
12	479,001,600	6m50s	153m7s

Table 3: Comparing GPU with CPU Performances in Two-Bit-BFS in Pancake Problems.

In *Peg-Solitaire* we find symmetry, which applies to the entire state space. If we invert the board (exchanging pegs with holes or swapping the colors), the goal and the initial state are the same. Moreover, the entire forward and backward graph structures match.

Hence, a call of backward breadth-first search to determine the number of states with a fixed goal distance is not needed. The number of states with a certain goal distances matches the number of states with a the same distance to the initial state. The total number of reachable states is 187,636,298.

We parallelized the game expanding and ranking states on the GPU. The total time for a BFS we measured was about 12m on the CPU and 1m8s on the GPU.

As the puzzle is moderately small, we consider the GPU speed-up factor of about 6 wrt. CPU computation as being significant.

For validity of the results, we compared the exploration results match with the ones obtained in [12]. For this case we had to alter the reward structure to the one that is imposed by the general game description language that was used there. We found that the number of expanded states matches, but – as expected – the total time to classify the states using the specialized player on the GPU is much smaller than in the general player of [12] running on one core of the CPU.

Frogs-and-Toads Similar to *Peg-Solitaire* if we invert the board (swapping the colors of the pieces), the goal and the initial state are the same, so that forward breadth-first search sufficies to solve the game.

In a BFS of about 0.19 seconds we validated the result of Dudeney for the *Fore and Aft* problem that reversing black and white takes 46 moves. There are two patterns which require 47 moves, namely, after reversing black and white, put one of the far corner pieces in the center. Table 5 also shows that there are 218,790 possible patterns of the pieces.

As *Frogs-and-Toads* generalizes *Fore and Aft*, we next considered the variant with 15 black and 15 white pieces on a board with 31 squares. The BFS outcome computed in 148m is shown in Table 6. We monitored that reversing black and white pieces takes 115 steps (in a shortest solution) and see that the worst-case input is slightly harder and takes 117 steps. A GPU parallelization leading to the same exploration results required about half an hour run-time.

Fox-and-Geese The next set of results shown in Table 7 considers the *Fox-and-Geese* game, where we applied retrograde analysis. For a fixed fox position the remaining geese can be binomially hashed. Moves stay in the same partition.

Holes	Bits	Space	Expanded
0	1	1 B	–
1	33	5 B	1
2	528	66 B	4
3	5,456	682 B	12
4	40,920	4,99 KB	60
5	237,336	28,97 KB	296
6	1,107,568	135 KB	1,338
7	4,272,048	521 KB	5,648
8	13,884,156	1.65 MB	21,842
9	38,567,100	4.59 MB	77,559
10	92,561,040	11.03 MB	249,690
11	193,536,720	23.07 MB	717,788
12	354,817,320	42.29 MB	1,834,379
13	573,166,440	68.32 MB	4,138,302
14	818,809,200	97.60 MB	8,171,208
15	1,037,158,320	123 MB	14,020,166
16	1,166,803,110	139 MB	20,773236
17	1,166,803,110	139 MB	26,482,824
18	1,037,158,320	123 MB	28,994,876
19	818,809,200	97.60 MB	27,286,330
20	573,166,440	68.32 MB	22,106,348
21	354,817,320	42.29 MB	15,425,572
22	193,536,720	23.07 MB	9,274,496
23	92,561,040	11.03 MB	4,792,664
24	38,567,100	4.59 MB	2,120,101
25	13,884,156	1.65 MB	800,152
26	4,272,048	521 KB	255,544
27	1,107,568	135 KB	68,236
28	237,336	28.97 KB	14,727
29	40,920	4.99 KB	2529
30	5,456	682 B	334
31	528	66 B	33
32	33	5 B	5
33	1	1 B	-

Table 4: One-Bit-BFS Results for *Peg-Solitaire*.

Depth	Expanded	Depth	Expanded	Depth	Expanded	Depth	Expanded
1	1	13	1,700	25	15,433	37	1,990
2	8	14	2,386	26	14,981	38	1,401
3	13	15	3,223	27	14,015	39	914
4	14	16	4,242	28	12,848	40	557
5	32	17	5,677	29	11,666	41	348
6	58	18	7,330	30	10,439	42	202
7	121	19	8,722	31	9,334	43	137
8	178	20	10,084	32	7,858	44	66
9	284	21	11,501	33	6,075	45	32
10	494	22	12,879	34	4,651	46	4
11	794	23	13,997	35	3,459	47	11
12	1,143	24	14,804	36	2,682	48	2

Table 5: BFS Results for *Fore and Aft*.

In spite of the work for classification being considerable, it was feasible to complete the analysis with 12 GB RAM. In fact, we observed that the largest problem with 16 geese required resistant space in main memory of 9.2 GB RAM.

The first three levels do not contain any state won for the geese, which matches the fact that four geese are necessary to block the fox (at the middle boarder cell in each arm of the cross). We observe that after a while, the number of iterations shrinks for a raising number of geese. This matches the experience that with more geese it is easier to block the fox.

Recall that all potentially drawn positions that couldn't been proven won or lost by the geese, are devised to be a win for the fox. The critical point, where the fox looses more than 50% of the game seems to be reached at currently explored level 16. This matches the observation in practical play, that the 13 geese are too less to show an edge for the geese.

The total run-time of about 730h (about a month) for the experiment is considerable. Without multi-core parallelization, more than 7 month would have been needed to complete the experiments. Even though we parallelized only the iteration stage of the algorithm, the speed-up on the 8-core machine is larger than 7, showing an almost linear speed-up.

The total of space needed for operating an optimal player is about 34 GB, so that in case geese are captured we would have to reload data from disk. This strategy yields a maximal space requirement of 4.61 GB RAM, which might further be reduced by reloading data in case of a fox moves.

8 Discussion

In this section we discuss further applications of the above approach.

The games have different applications in moving target search. For example, *Fox-and-Geese* is prototypical for chasing an attacker, with applications to computer security, where an intruder

Depth	Expanded	Depth	Expanded	Depth	Expanded	Depth	Expanded
1	1	31	4,199,886	61	171,101,874	91	4,109,157
2	8	32	5,447,660	62	170,182,837	92	3,156,288
3	17	33	6,975,087	63	168,060,816	93	2,387,873
4	26	34	8,865,648	64	164,733,845	94	1,780,521
5	46	35	11,138,986	65	160,093,746	95	1,307,312
6	78	36	13,881,449	66	154,297,247	96	948,300
7	169	37	17,060,948	67	147,342,825	97	680,299
8	318	38	20,800,347	68	139,568,855	98	484,207
9	552	39	25,048,652	69	131,146,077	99	340,311
10	974	40	29,915,082	70	122,370,443	100	235,996
11	1,720	41	35,382,942	71	113,415,294	101	160,153
12	2,905	42	41,507,233	72	104,380,748	102	107,024
13	4,826	43	48,277,767	73	95,379,850	103	69,216
14	7,878	44	55,681,853	74	86,375,535	104	44,547
15	12,647	45	63,649,969	75	77,534,248	105	27,873
16	19,980	46	72,098,327	76	68,891,439	106	17,394
17	31,511	47	80,937,547	77	60,672,897	107	10,256
18	49,242	48	89,999,613	78	52,953,463	108	6,219
19	74,760	49	99,231,456	79	45,889,798	109	3,524
20	112,218	50	108,495,904	80	39,482,737	110	2,033
21	166,651	51	117,679,229	81	33,751,896	111	1,040
22	241,157	52	126,722,190	82	28,607,395	112	532
23	348,886	53	135,363,894	83	24,035,844	113	251
24	497,698	54	143,534,546	84	19,957,392	114	154
25	700,060	55	150,897,878	85	16,394,453	115	42
26	974,219	56	157,334,088	86	13,306,659	116	19
27	1,337,480	57	162,600,933	87	10,695,284	117	10
28	1,812,712	58	166,634,148	88	8,521,304	118	2
29	2,426,769	59	169,360,939	89	6,738,557		
30	3,214,074	60	170,829,205	90	5,286,222		

Table 6: BFS Results for *Frogs-and-Touds*.

Geese	States	Space	Iterations	Won	Time Real	Time User
1	2,112	264 B	1	0	0.05s	0.08s
2	32,736	3.99 KB	6	0	0.55s	1.16s
3	327,360	39 KB	8	0	0.75s	2.99s
4	2,373,360	289 KB	11	40	6.73s	40.40s
5	13,290,816	1.58 MB	15	1,280	52.20s	6m24s
6	59,808,675	7.12 MB	17	21,380	4m37s	34m40s
7	222,146,996	26 MB	31	918,195	27m43s	208m19s
8	694,207,800	82 MB	32	6,381,436	99m45s	757m0s
9	1,851,200,800	220 MB	31	32,298,253	273m56s	2,083m20s
10	4,257,807,840	507 MB	46	130,237,402	1,006m52s	7,766m19s
11	8,515,615,680	1015 MB	137	633,387,266	5,933m13s	46,759m33s
12	14,902,327,440	1.73 GB	102	6,828,165,879	4,996m36s	36,375m09s
13	22,926,657,600	2.66 GB	89	10,069,015,679	5,400m13s	41,803m44s
14	31,114,749,600	3.62 GB	78	14,843,934,148	5,899m14s	45,426m42s
15	37,337,699,520	4.24 GB	73	18,301,131,418	5,749m6s	44,038m48s
16	39,671,305,740	4.61 GB	64	20,022,660,514	4,903m31s	37,394m1s
17	37,337,699,520	4.24 GB	57	19,475,378,171	3,833m26s	29,101m2s
18	31,114,749,600	3.62 GB	50	16,808,655,989	2,661m51s	20,098m3s
19	22,926,657,600	2.66 GB	45	12,885,372,114	1,621m41s	12,134m4s
20	14,902,327,440	1.73 GB	41	8,693,422,489	858m28s	6,342m50s
21	8,515,615,680	1015 MB	36	5,169,727,685	395m30s	2,889m45s
22	4,257,807,840	507 MB	31	2,695,418,693	158m41s	1,140m33s
23	1,851,200,800	220 MB	26	1,222,085,051	54m57	385m32s
24	694,207,800	82 MB	23	477,731,423	16m29s	112m.35s
25	222,146,996	26 MB	20	159,025,879	4m18s	28m42s
26	59,808,675	7.12 MB	17	44,865,396	55s	5m49s
27	13,290,816	1.58 MB	15	10,426,148	9.81s	56.15s
28	2,373,360	289 KB	12	1,948,134	1.59s	6.98s
29	327,360	39 KB	9	281,800	0.30s	0.55s
30	32,736	3.99 KB	6	28,347	0.02s	0.08s
31	2,112	264 B	5	2001	0.00s	0.06s

Table 7: Retrograde Analysis Results for *Fox-and-Geese*.

has to be catched. In a more general setting, the board games are played with tokens on a graph $G = (V, E)$. For example, one move corresponds to pass a token along an edge $(i, j) \in E$. The space complexities of the bit-state analysis now depends on the number of tokens played and the number of nodes. For particular types of Petri nets like safe nets this might yield an appropriate compression for their exploration.

8.1 Symmetries

Symmetries are helpful to reduce the time and space consumption of a classification algorithm.

In many board games we find reflection along the main axes or along the diagonals. If we look at the four possible rotations on the board for *Peg-Solitaire* and *Fox-and-Geese* plus reflection, we count 8 symmetries in total.

The exploitation of state symmetries are of various kinds. For *Fox-and-Geese* we can classify all states that share a symmetrical fox position by simply copying the result obtained for the existing one. Besides the savings of time for not expanding states, this can also save the number of positions that have to be kept in RAM during fixpoint computation.

If the forward and backward search graphs match (as in *Peg Solitaire* and *Frogs-and-Toads*) we may also truncate the breadth-first search proceedure to the half of the search depth. In two-bit BFS, we simply have to look at the rank of the inverted unranked state. Moreover, with the forward BFS layers we also have the minimal distances of each state to the goal state, and, hence, the classification result.

8.2 Frontier Search

Frontier search is motivated by the attempt of omitting the *Closed* list of states already expanded. It mainly applies to problem graphs that are directed or acyclic but has been extended to more general graph classes. It is especially effective if the ratio of *Closed* to *Open* list sizes is large.

Frontier search requires the *locality* of the search space [28] being bounded, where the locality (for breadth-first search) is defined as $\max\{\mathit{layer}(s) - \mathit{layer}(s') + 1 \mid s, s' \in S; s' \in \mathit{succs}(s)\}$, where $\mathit{layer}(s)$ denotes the depth d of s in the breadth-first search tree.

For frontier search, the *space efficiency* of the hash function $h : S \to \{0, \ldots, m-1\}$ boils down to $m / \left(\max_d |\mathit{Layer}_d| + \ldots + |\mathit{Layer}_{d+l}|\right)$, where Layer_d is set of nodes in depth d of the breadth-first search tree and l is the locality of the breadth-first search tree as defined above.

For the example of the Fifteen puzzle, i.e., the 4×4 version of *Sliding-Tile*, the predicted amount of 1.2 TB hard disk space for 1-bit breadth-first search is only slightly smaller than the 1.4 TB of frontier breadth-first search reported by [21].

As frontier search does not shrink the set of states reachable, one may conclude, that frontier search hardly cooperates well with a bitvector representation of the entire state space. However, if layers are hashed individually, as done in all selection games we have considered, a combination of bit-state and frontier search is possible.

8.3 Pattern Databases

The breadth-first traversal in a bitvector representation of the search space is also essential for the construction of compressed pattern databases [5]. The number of bits per state can be reduced to $\log 3 \approx 1.6$. For this case, 5 values $\{0, 1, 2\}$ are packed into a byte, given that $3^5 = 243 < 255$. The observation that $\log 3$ are sufficient to represent all mod-3 values possible and the byte-wise packing was already made by [8].

The idea of pattern database compression is to store the mod-3 value (of the backward BFS depth) from abstract space, so that its absolute value can be computed incrementally in constant time. For the initial state, an incremental computation for its heuristic evaluation is not available, so that a backward construction of its generating path can be used. As illustrated in [5], for an undirected graph a shortest path predecessor with mod-3 of BFS depth k appears in level $k - 1$ *mod* 3.

As the abstract space is generated anyway for generating the database, one could alternatively invoke a shortest path search from the initial state, without exceeding the time complexity of database construction.

By having computed the heuristic value for the projected initial state as the goal distance in the inverted abstract state space graph, as shown in [5] all other pattern database lookup values can then be determined incrementally in constant time, i.e., $h(v) = h(u) + \Delta(v)$, with $v \in \textit{succs}(u)$ and $\Delta(v)$ found using the mod-3 value of v. Given that the considered search spaces in [5] are undirected, the information to evaluate the successors with $\Delta(v) \in \{-1, 0, 1\}$ is possible.

For directed (and unweighted) search spaces more bits are needed to allow incremental heuristic computation in constant time. It is not difficult to see that the locality in the inverted abstract state space determines the maximum difference in h-values $h(v) - h(u)$, $v \in \textit{succs}(u)$ in original space.

Theorem 9 (Locality determines Number of Bits for Pattern Database Compression) *In a directed (but unweighted) search space, the (dual) logarithm of the (breadth-first) locality of the inverse of the abstract state space graph plus 1 is an upper bound on the number of bits needed for incremental heuristic computation of bit-vector compressed pattern databases, i.e., for locality* $l_A^{-1} = \max\{\textit{layer}^{-1}(u) - \textit{layer}^{-1}(v) + 1 \mid u, v \in A; v \in \textit{succs}^{-1}(u)\}$ *in abstract state space graph* A *of* S *we require at most* $\log\lceil l_A^{-1}\rceil + 1$ *bits to reconstruct the value* $h(v)$ *of a successor* $v \in S$ *of any chosen* $u \in S$ *given* $h(u)$.

Proof. First we observe that the goal distances in abstract space A determine the h-value in original state space, so that the locality $\max\{\textit{layer}^{-1}(u) - \textit{layer}^{-1}(v) + 1 \mid u, v \in A; v \in \textit{succs}^{-1}(u)\}$ is bounded by $h(u) - h(v) + 1$ for all u, v in original space with $u \in \textit{succs}(v)$, which is equal to the maximum of $h(v) - h(u) + 1$ for $u, v \in S$ with $v \in \textit{succs}(u)$. Therefore, the number of bits needed for incremental heuristic computation equals $\lceil \max\{h(v) - h(u) \mid u, v \in A; v \in \textit{succs}^{-1}(u)\}\rceil + 2$ as all values in the interval $[h(u) - 1, \ldots, h(v)]$ have to be accommodated for. Thus for the incremental value $\Delta(v)$ added to $h(u)$ we have $\Delta(v) \in \{-1, \ldots, h(v) - h(u)\}$, so that $\lceil\log(\max\{h(v) - h(u) + 2 \mid u, v \in S; v \in \textit{succs}(u)\})\rceil = \log\lceil l_A^{-1}\rceil + 1$ bits suffice to reconstruct the value $h(v)$ of a successor $v \in S$ for every $u \in S$ given $h(u)$. $\square$

For undirected search spaces we have $\log l_A^{-1} = \log 2 = 1$, so that $1 + 1 = 2$ bits suffice to be stored for each abstract pattern state according to the theorem. Using the tighter packing of the $2 + 1 = 3$ values into bytes provided above, $8/5 = 1.6$ bits are sufficient.

If not all states in the search space that has been encoded in the perfect hash function are reachable, reducing the constant-bit compression to a lesser number of bits might not always be available, as unreached states cannot easily be removed. For this case, the numerical value remaining to be set for an unreachable states in the inverse of abstract state space will stand for h-value infinity, at which the search in the original search space can stop.

For problems with discretized costs, more general notions of locality based on cost-based backward construction have been developed [18]. More formally, the best-first locality has been defined as $\max\{\textit{cost-layer}(s) - \textit{cost-layer}(s') + \textit{cost}(s, s') \mid s, s' \in S; s' \in \textit{succs}(s)\}$, where *cost-layer*$(s)$ denotes the smallest accumulated *cost*-value from the initial state to s. The theoretical considerations on the number of bits needed to perform incremental heuristic evaluation extend to this setting.

8.4 Other Games

We distinguish between permutation games and selection games, and add remarks on general games for which a functional representation of the state space exists.

8.4.1 Permutation Games

Rubik's Cube, invented in the late 1970s by Erno Rubik, is a known challenge for single-agent search [19]. Each face can be rotated by 90, 180, or 270 degrees and the goal is to rearrange a scrambled cube such that all faces are uniformly colored.

Solvability invariants for the set of all dissembled cubes are:

- a single corner cube must not be twisted
- a single edge cube must not be twisted and
- no two cube must be exchanged

For the last issue the parity of the permutation is crucial and leads to $8! \cdot 3^7 \cdot 12! \cdot 2^{11}/2 \approx 4.3 \cdot 10^{19}$ solvable states. Assuming one bit per state, an impractical amount of $4.68 \cdot 10^{18}$ bytes for performing full reachability is needed. For generating upper bounds, however, bitvector representations of subspaces have been shown to be efficient [23].

8.4.2 Selection Games

The binomial and multinomial hashing approach is applicable to many other pen-and-paper and board games.

- In *Awari* [25] the two player redistribute seeds among 12 holes according to the rules of the game, with an initial state having uniformly four seeds in each of the holes. When all seeds are available, all possible layouts can be generated in an urn experiments with 59 balls, where 48 balls represent filling the current hole with a seed and 11 balls indicate changing from the current to the next hole. Thus the binomial hash function applies.
- In *Dots and Boxes* players take turns joining two horizontally or vertically adjacent dots by a line. A player that completes the fourth side of a square (a box) colors that box and must play again. When all boxes have been colored, the game ends and the player who has colored more boxes wins. Here, the binomial hash suffices. For each edge we denote whether or not it is marked. Together with the marking, we denote the number of boxes of at least one player. In difference to other games, all successors are in the next layer, so that one scan suffices to solve the current one.
- *Nine-Men's-Morris* is one of the oldest games still played today. Boards have been found on many historic buildings throughout the world. One of the oldest dates back to about 1400 BC [15]. The game naturally divides in three stages. Each player has 9 pieces, called men, that are first placed alternately on a board with 24 locations. In the second stage, the men move to form mills (a row of three pieces along one of the board's lines), in which case one man of the opponent (except the ones that form a mill) is removed from the board. In one common variation of the third stage, once a player is reduced to three men, his pieces may "fly" to any empty location. If a move has just closed a mill, but all the opponent's men are also in mills, the player may declare any stone to be removed. The game ends if a player has less than three men (the player loses), if a player cannot make a legal move (the player loses), if a midgame or endgame position is repeated (the game is a draw).

  Besides the usual symmetries along the axes, there is one in swapping the inner with the outer circle. Gassner has solved the game by computing large endgame databases for the last two phases together with alpha-beta search for the first phase [15]. His results showed that, assuming optimal play of both players, the game ends in a draw. For this game the multinomial hash is applicable.

8.5 General Games

It is not difficult to extend the above functions to more than two different sets of pieces on the board. For *Chinese Checkers*, for example, three and more colors are needed. In this case a larger multinomial coefficient has to be built, but the construction remains similar to the one above.

We now look at general games with state spaces provided in functional representation. This setting complements the explicit-state setting of Botelho et al.. The state space is present in so-called *functional representation*. It has been constructed in symbolic forward search and a bijection of all states S reached to $\{0, \ldots, |S| - 1\}$ is computed together with its inverse. This approach will have potential applications in action planning, general game playing, and model checking.

Algorithm 12 Rank-BDDs(s, v)

if v is 0-sink **then**
  return 0
if v is 1-sink **then**
  return 1
if v is node labeled x_i with 0-succ. u and 1-succ. w **then**
  if $s[i] = 1$ **then**
    return sat-count(v) + rank(s, w)
  if $s[i] = 0$ **then**
    return rank(s, u)

Algorithm 13 Unranking-BDDs(r)

$i := 1$
start at root
while $i \leq n$ **do**
  at node v for x_i with 0-succ. u and 1-succ. w
  if $r \geq$ *sat-count*(u) **then**
    $r := r -$ *sat-count*(u)
    follow 1-edge to w, record $s[i] := 1$
  else
    follow 0-edge to u, record $s[i] := 0$
  $i := i + 1$

The above algorithms are special cases of according ranking and unranking functions developed for BDDs [11]. For the sake of completeness, the according rank and unrank algorithms are shown in Algorithm 12 and Algorithm 13. The BDD for representing the $\binom{n}{k}$ structure is of polynomial size. Secondly, up to the links to the zero sink that do not contribute to counting the number of satisfying paths, the BDD is quasi-reduced by means that all variables appear on every path.

For simple reachability analysis this does not provide any surplus, but in case of more complex algorithms, like the classification of two-player games, perfect hash function based on BDDs show computational advantages in form of (internal or external) memory gains.

9 Conclusion

In this work we presented and analyzed linear time ranking and unranking functions for games in order to execute breadth-first search and retrograde analysis on sparse memory. We reflected that such constant-bit state space traversal to solve games is applicable, only if invertible and perfect hash functions are available. As an interesting time-space trade-off we studied one-bit reachability and one-bit breadth-first search.The latter imposes the presence of a move alternation property. Some previously unresolved games were solved, mostly in RAM, but sometimes using I/O-efficient strategies.

The approach featured parallel explicit-state traversal of two challenging games on limited space. We studied the application of multiple-core CPU and GPU computation and accelerated the analysis. The speed-ups compare well with alternative results combining external-memory and parallel search on multiple cores [21, 29].

In our experiments, the CPU speed-up is almost linear in the number of cores. For this we exploited independence in the problem, using an appropriate projection function. The GPU speed-up often exceeds the number of CPU cores considerably.

To compute invertible minimal perfect hash functions for permutation games, we extended the already efficient method by Myrvold and Ruskey. For selection games, with binomial and multinomial hashing we proposed an approach that has been inspired by counting the number of paths (lexicographic smaller to the given assignment) in a BDD. Due to the small amount of available shared RAM of 16 KB on the GPU, we prefer the space requirements for the ranking and unranking functions to be small.

References

[1] E. R. Berlekamp, J. H. Conway, and R. K. Guy. *Winning Ways*. Academic Press, 1982.

[2] B. Bonet. Efficient algorithms to rank and unrank permutations in lexicographic order. In *AAAI-Workshop on Search in AI and Robotics*, 2008.

[3] D. Bosnacki, S. Edelkamp, and D. Sulewski. Efficient probabilistic model checking on general purpose graphics processors. In *Model Checking Software (SPIN)*, 2009.

[4] F. C. Botelho and N. Ziviani. External perfect hashing for very large key sets. In *ACM Conference on Information and Knowledge Management (CIKM)*, pages 653–662, 2007.

[5] T. Breyer and R. Korf. 1.6-bit pattern databases. In *Symposium on Combinatorial Search (SOCS)*, 2009.

[6] M. Campbell, J. A. J. Hoane, and F. Hsu. Deep blue. *Artificial Intelligence*, 134(1-2):57–83, 2002.

[7] T. Chen and S. Skiena. Sorting with fixed-length reversals. *Discrete Applied Mathematics*, 71(1–3):269–295, 1996.

[8] G. Cooperman and L. Finkelstein. New methods for using Cayley graphs in interconnection networks. *Discrete Applied Mathematics*, 37/38:95–118, 1992.

[9] J. C. Culberson and J. Schaeffer. Pattern databases. *Computational Intelligence*, 14(4):318–334, 1998.

[10] M. Dweighter. Problem e2569. *American Mathematical Monthly*, (82):1010, 1975.

[11] S. Edelkamp and M. Diezfelbinger. Perfect hashing for state spaces in BDD representation. In *German Conference on Artificial Intelligence (KI)*, 2009.

[12] S. Edelkamp and P. Kissmann. Symbolic classification of general two-player games. In *KI*, pages 185–192, 2008.

[13] S. Edelkamp and D. Sulewski. Model checking via delayed duplicate detection on the GPU. Technical Report 821, TU Dortmund, 2008.

[14] S. Edelkamp and D. Sulewski. State space search on the GPU. In *Symposium on Combinatorial Search (SOCS)*, 2009.

[15] R. Gassner. Solving Nine-Men-Morris. *Computational Intelligence*, (12):24–41, 1996.

[16] W. H. Gates and C. H. Papadimitriou. Bounds for sorting by prefix reversal. *Discrete Mathematics*, 27:47–57, 1979.

[17] E. Hordern. *Sliding Piece Puzzles*. Oxford University Press, 1986.

[18] S. Jabbar. *External Memory Algorithms for State Space Exploration in Model Checking and Planning*. PhD thesis, University of Dortmund, 2008.

[19] R. E. Korf. Finding optimal solutions to Rubik's Cube using pattern databases. In *National Conference on Artificial Intelligence (AAAI)*, pages 700–705, 1997.

[20] R. E. Korf. Minimizing disk I/O in two-bit-breath-first search. In *National Conference on Artificial Intelligence (AAAI)*, pages 317–324, 2008.

[21] R. E. Korf and T. Schultze. Large-scale parallel breadth-first search. In *National Conference on Artificial Intelligence (AAAI)*, pages 1380–1385, 2005.

[22] D. L. Kreher and D. R. Stinson. *Combinatorial Algorithms*. Discrete Mathematics and Its Applications, 1984.

[23] D. Kunkle and G. Cooperman. Twenty-six moves suffice for Rubik's cube. In *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 235 – 242, 2007.

[24] W. Myrvold and F. Ruskey. Ranking and unranking permutations in linear time. *Information Processing Letters*, 79(6):281–284, 2001.

[25] J. W. Romein and H. E. Bal. Awari is solved. *Journal of the ICGA*, 25:162–165, 2002.

[26] J. Schaeffer, Y. Björnsson, N. Burch, A. Kishimoto, and M. Müller. Solving checkers. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 292–297, 2005.

[27] R. Zhou and E. A. Hansen. Structured duplicate detection in external-memory graph search. In *National Conference on Artificial Intelligence (AAAI)*, pages 683–689, 2004.

[28] R. Zhou and E. A. Hansen. Breadth-first heuristic search. *Artificial Intelligence*, 170(4-5):385–408, 2006.

[29] R. Zhou and E. A. Hansen. Parallel structured duplicate detection. In *National Conference on Artificial Intelligence (AAAI)*, pages 1217–1222, 2007.

Toward Unobtrusive Service Composition in Smart Environments based on AI Planning

Florian Marquardt, Adelinde M. Uhrmacher

University of Rostock, Albert-Einstein-Str. 21, 18059 Rostock, Germany
fm@informatik.uni-rostock.de, lin@informatik.uni-rostock.de

Abstract. Using AI planning for service composition in smart environments requires fast and reliable implementations of planning algorithms. Experimentally the behavior of several planners were investigated with respect to the number of operators, of state variables, of true propositions in the initial world, of goals, and of preconditions and effects by executing several millions of experiment runs on about 55.000 generated problem configurations. Based on the experimental results we outline a heuristic that in addition leverages the knowledge of phase shifts. When a planner's execution time crosses a certain threshold, it is canceled. It is shown that this strategy leads to a significant speedup without losing too many solutions and thus, presents a promising step toward realizing unobtrusive service composition in smart environments.

1 Motivation

The term *Smart Environment* (SE) is used in many different ways, e.g. for Smart Houses [4], Smart Health Care [1], Smart Laboratories [22] and Smart Rooms [12]. Thereby, the term "smart" refers to services that are offered to the user in an unobtrusive manner, and the term "environment" refers to a dynamic ensemble of devices responsible for offering the services. This kind of "smartness" typically requires the cooperation of several devices and the identification of a user's intention. The process of user assistance in smart environments relies on strategy synthesis [12]. Based on goals, that are derived from user intentions, the strategy synthesis tries to infer a suitable sequence of actions, to be executed on the available devices in the environment. The strategy synthesis, which is responsible for composing services, has to deal with a dynamic environment. The goals and the availability of services may change over time, new services might emerge whereas others disappear. Therefore, automated service composition is a vital part of ubiquitous environments. To keep composition imperceptible for the user a fast reaction time is crucial.

Different methods for composing services in smart environments have been applied [3], e.g. Workflow methods and AI planning [19]. So far overviews and comparisons have focused on a qualitative perspective, e.g. general robustness and applicability [18]. The quantitative analysis of different composition methods for smart environments is an open challenge [2], but mandatory to enable imperceptible support for the user. Referring to the dynamic nature of smart environments, AI planners are of particular interest for composing services in smart environments.

2 Using AI planning in SEs

First a mapping between AI planning and the problem of service composition in smart environments is given. We can characterize SEs by the involved services, the context information, and the intentions of the users. This characterization is based on the following assumptions. First, every functionality of the environments can be accessed as a service, second, all context information is available anytime and third, user intentions can be predicted. Although at this time neither of them can be taken for granted and are as such topics of current research [14, 1, 9, 5], based on these assumptions, fast and reliable composition by AI planning is a building block for user assistance in smart environments. To be able to apply AI planning, an initial world state, which is expressed using a predefined set of state variables (propositions), a set of operators, and a set of goals are needed. The services of the environment correspond to planning operators. The measurements and values of the context represent the world state and the user intentions form the goal for the planning process. Figure 1 illustrates this mapping.

3 Requirements

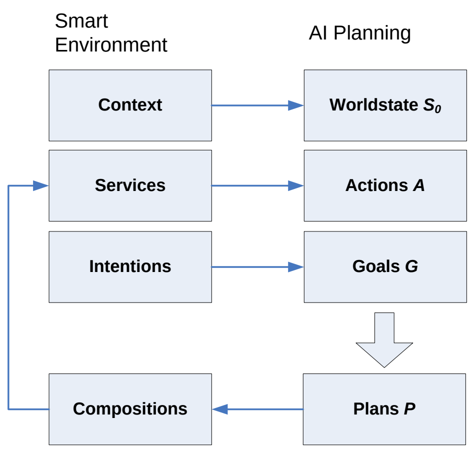


Fig. 1. Mapping Service compostion in SEs to AI planning.

We want to examine if planners react fast enough, if their reaction time is robust and not too volatile, and if their results are reliable. To keep the planning and composition process as imperceptible as possible its runtime has to be minimized. I.e. if the state of the environment, a goal, or the available services changes the smart environment should react as quickly as possible to avoid distracting the user. However in some cases the runtimes of planning processes tend to take unusually longer than comparable runs. We call these cases peaks. They tend to occur in case of hard problems. Thereby the value of the peaks is much higher than the mean of comparable problems. Furthermore the planning process has to be robust. We need planning implementations that do not crash and accept all well formated inputs. The planners have to be able to understand a common language for data exchange. The current state of the art for this is PDDL [11]. Finally the results gathered by the planning algorithms have to be reliable. If planners return a plan for a given domain and problem, this plan is correct[1], nevertheless it is not necessarily optimal in terms of operator length or other quality measures. In case a planner returns that no plan exists it can not be inferred inevitably that really no plan exists, even though this outcome can be trusted in the majority of cases. This uncertainty is based on internal limits of planners that may cancel a planning run, albeit a solution exists, or due to externally introduced thresholds (see section "Introducing thresholds for early canceling of planning runs"). First small experiments [16] and a literature review on evaluation of planners indicated the following hypothesis and questions that might be of relevance in using AI planners for service composition in smart environments.

1. If a planning run results in a peak, then its outcome is supposed to be no plan and the run can be canceled early.
2. Different planners run into different peaks.
3. Is there a significant area of problems that surrogates a bigger problem space in order to gain key characteristics of a planner with less experiments and thus to help predict the planners' performance?

When implementing service composition in SEs it is not feasible to reimplement planning algorithms and all of their published refinements from scratch, one would rather use an available implementation "as is". We used three planners with their most recent versions (state end of 2008), namely SGP [24], blackbox [15], and LPG [10], all of which support PDDL and are more recent implementations of planning algorithms. We confined ourself to use three planners only, as every planner causes nearly seven 24-hour days of experiment runs. Referring to hard problems and runtime peaks the so called phase transitions are of interest and require further investigation.

4 Related Work

Phase transitions are a phenomenon that all problems with complexity of at least NP exhibit [8]. As planning with at least one precondition and one effect is in NP (or worse with more preconditions or effects) [6], phase shifts occur for planning problems, too [7]. The theory of phase transitions predicts that in a critical region of the problem instances the problems may become extremely difficult. That causes algorithms to take unusually long time to solve these problems. Bylander defined theoretical lower and upper bounds that help to identify phase shift regions for planning problems. These borders differ by a factor that is exponential in the number of preconditions and effects. He furthermore showed that outside the phase transition simple algorithms are able to solve planning problems easily and hence quickly: a statement definitively of relevance for smart

[1] All planners claim to be sound. In practice we revealed at least one bug that led to incorrect plans in a very narrow number of cases.

environments. Being able to determine whether a given problem is inside a phase transition or not would help in ensuring reliability of service composition based on AI planning. Rintanen [20] conducted an experimental evaluation of planners based on generated problems. He explicitly explored the behavior of planners at the phase transitions. His experiments showed that planners based on SAT-Solvers outperform planners using heuristic local search. A very comprehensive benchmark for planning algorithms is provided by the established ICAPS (formerly AIPS) planning competition (IPC), wherein the best planners in several categories are nominated bi-annually. Although these publications are very helpful starting points for a selection of fast planners for service composition in SEs, those evaluations are not concerned with the special requirements of this application. Their test sets do not reflect the dynamic circumstances of smart environments. In [21] an evaluation of 28 planners on 4726 different problems including problems of the IPC test set was conducted. The goal of their work was to examine empirically which planner works best for which problems and learn models of planner's performance in terms of propapility of success and estimated runtime. So the purpose of the study is similar, however not focused on smart environments. The study confirmed that no silver bullet exists but different planners excel for different kinds of problems. Unlike Rintanen, Roberts et al. have not taken into account the phenomenon of phase transitions in AI planning.

5 Experimental Setup

Our approach partly overlaps with the work of Bylander [7] and Rintanen [20]. We also generated random problems rather than relying on predefined problems like the competition problems. The first and foremost reason for using generated problem instances is the lack of a representative problem set for service composition in smart environments. In addition, since smart environments are still a young area of research, we are not able to define characteristic structures of relevant planning problems. The drawback of generating problems is their relevance [23]. Thus we need to explore a very wide area of problems, in order to cover all relevant problem regions. To constrain the parameter range for generating composition problems in smart environments we conducted a small survey [16].

Rintanen introduced three models for generating planning problems. We use the model B also used by Bylander. The number of preconditions and effects (PEs) is constant for each operator. To generate an effect of an operator randomly a fixed number of state variables was chosen. One half of these state variables form the positive and the other half the negative effects. Thereby, many instances that can be easily classified as insoluble are generated. This was criticized by Rintanen and accordingly, he introduced alternative models to exclude many insoluble problems. However, due to the fact that in smart environments it can not be assumed that plans can be found to support the intentions of the user, model B is suitable in this context, and therefore was selected. Only parameter-free operators were used for our domain descriptions. In order to limit the overall runtime of the experiments to a predictable and reasonable amount, we decided to limit the maximal runtime for each planning run to 2000 ms.

To define a planning problem, we adapt the notation of Bylander: with o (number of operators), n (number of propositions), i (number of true initial world states), g (number of goals), r and s (number of preconditions and effects). In our setting the number of preconditions and effects of an operator is equal, i.e. $s = r$, and even. The effects are divided into two equally-sized lists of positive and negative effects. Hence we just use r to identify the number of preconditions and effects of an operator. During our experiments we increased the number of all variables. PEs ranged from 2 to 20 with a stepsize of 2. Similar to r the number of true propositions in the initial world state i ranged from 2 to 20 with a stepsize of 2. As a result we gain 55 reasonable combinations of r and i.[2] For each combination we iterated over o (10 to 100, stepsize=10), n (10 to 100, stepsize=10) and g (1 to 10, stepsize=1) resulting in 1000 parameterizations[3] for each of the 55 prior combinations. Due to that we identified about 55.000 planning problem configurations in terms of the 5 variables o, n, r, i, g. Using each setting we ran 100 replications with different seeds for the random number generator resulting in about 5.500.000 different problem instances. As we used three planners we conducted about 16.5 million experiment runs overall. All experiments were run on three machines with equal characteristics (Intel Core2Duo E8200 @ 2.66GHz with 2GB memory using WindowsXP ServicePack 3). The duration of the experiment runs was approximately three weeks.

6 Experimental Results

For each planning run we recorded different characteristics. First we stored the runtime. Second we gathered the result description that was returned by the planner. This is either a resulting plan or a statement that no plan can be found, possibly with a reason. Furthermore we stored flags whether the planner returned a plan and if the planner was canceled.

All planners were robust during our experiments and encountered no abnormal terminations. However we were not able to run domains with more than 341 operators either on LPG or on blackbox. In that case a syntax error was returned. In contrast SGP was able to parse and solve bigger domains. In very rare cases blackbox returns evidently wrong plans, which is presumably caused by a postprocessing step. To avoid wrong service compositions, plans should be validated after the planning process. A suitable validation algorithm is very simple. Given the initial state of the world it sequently applies all operators of the plan and compares the result with the demanded goal state. If they are not

[2] E.g. the combination of an initial world state with 10 true propositions and operators with 12 preconditions each is senseless as no operator would ever be applicable in that case.

[3] Comparable to r and i not all of these combinations are suitable, e.g. having 10 goals but just 10 propositions is not feasible, as in this case no propositions would be left for the initial world state.

equal, the plan is wrong. The runtime complexity of this validation is linear in the length of the resulting plan and thus not problematic compared to the antecedent planning run. All planners return reasons for failed planning runs, whereas LPG is more verbose and returns the most precise descriptions.

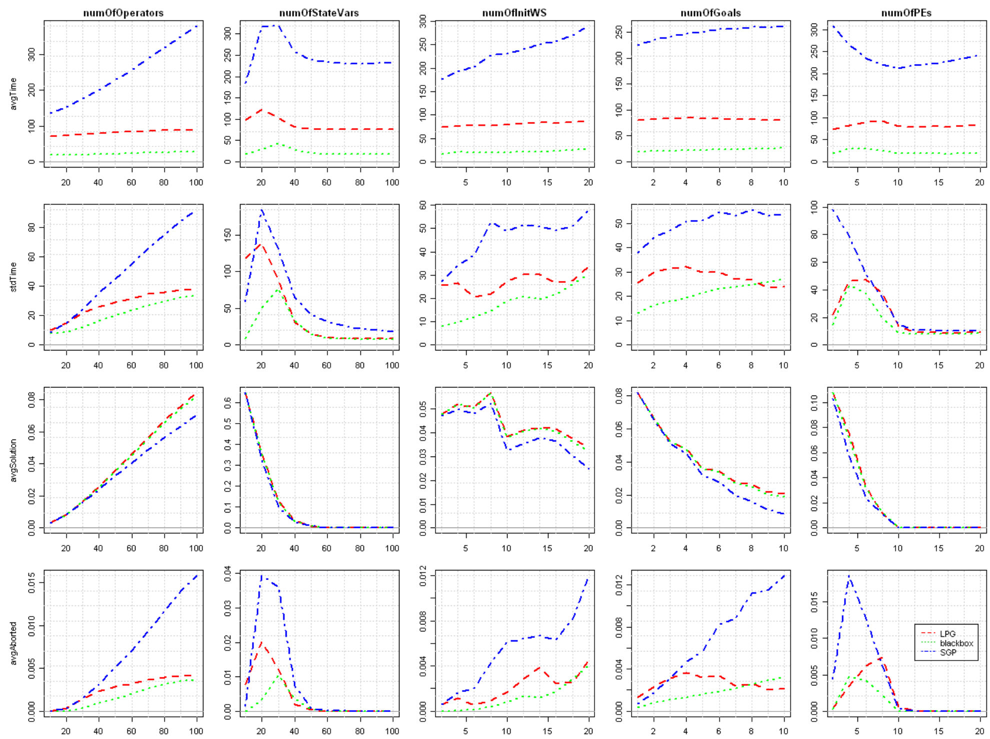


Fig. 2. Characteristic curves for average runtime, standard deviation of runtime, quota of found solutions, and quota of canceled runs against each of the five experiment variables

When aggregating all 100 replications of a problem instance four different characteristics were derived: average runtime (AVG), standard deviation of the runtime (STD), quota of found solutions, and quota of runs that exceed our time limit (aborted problems). Based on these values we are able to create plots for each of the four characteristics and each planner against the variables of the planning experiments (n, o, i, g, and r). Figure 2 shows the result.

The first row in Figure 2 depicts the average runtime of the planners. Blackbox is always fastest, followed by LPG and SGP. Except of the last plot where the number of preconditions and effects is considered, also the plots show a similar behaviour pattern between the planners. The standard deviation is a good indicator for regions of hard problems. If the runtime of the runs differs very much and thus the deviation is high we assume this set of problems to be hard. Thus, the second row is of most interest to us. Although blackbox excels here too, the differences between the planners shrink and the observed variances show more irregularities. Regarding the number of state variables in the second plot all planners exhibit a maximum. As LPG and SGP reach their maximum at 20 state variables, blackbox standard deviation grows until 30 state variables. The fourth plot in this row reveals that LPG becomes more robust with an increasing number of goals whereas blackbox steadily increases its instability in runtime. SGP remains on a high level. The number of found solutions in the third row depicts that the plots of all planners are very close. I.e. all planners find nearly the same number of plans. SGP's results are always a bit below blackbox's and LPG's, this is due to the fact that SGP in general is slower than the two others and as we canceled runs that last longer than 2000 ms in some cases SGP was not able to find solutions whereas LPG and blackbox had already completed their runs. The quota of average aborted runs depicted in the fourth row is similar to the standard deviation (row two) to a large extend. Both measures are indicators for regions with hard problems.

7 Reviewing our three hypothesis

Now let us look at the hypothesis we wanted to validate.

1. Peaks are supposed to be no plan.

If a planning run lasts significantly longer than comparable runs, then its outcome is supposed to be no plan, and the run can be canceled early. At first it is of interest how often peaks occur. Table 1 shows the ratio of peaks compared to the total number of runs. All runs above 500 ms runtime are threated as peaks in this case. We furthermore solely considered runs that was not aborted. Due to its too high runtimes SGP is not considered here. All problems with more than 60 operators are chosen and the number of state variables is increased by 10 from 20 to 50 in Tables 1 and 2. Therefor all regions of the phase shift are covered.

	blackbox	LPG
$n = 20$	0.1568 %	0.4118 %
$n = 30$	0.3943 %	0.2431 %
$n = 40$	0.0734 %	0.0894 %
$n = 50$	0.0121 %	0.0358 %

Table 1. Quota of occurrence of peaks for $o > 60$

	blackbox Overall	In peaks	LPG Overall	In peaks
$n = 20$	54.20 %	54.02 %	55.40 %	67.83 %
$n = 30$	23.08 %	95.53 %	23.74 %	66.13 %
$n = 40$	6.84 %	100.00 %	7.44 %	29.56 %
$n = 50$	1.74 %	92.86 %	1.84 %	10.84 %

Table 2. Probabilities to find a solution for $o > 60$

In Table 2 the probabilities to find a solution are shown. The probability that a peak results in a plan is as high as for the remaining run before the phase shift region, i.e. where the probability to find a solution is high in general. Albeit the overall probability to find a solution decreases from about 50% to less than 2% while increasing the number of propositions (n) from 20 to 50, in contrast the probability of the (extremly few) peaks to yield a plan is significantly higher. Interrestingly for n = 30, 40, 50 nearly all peaks result in a plan for blackbox whereas the according probabilities decreases for LPG.

Our first guess that the outcome of a peak is based on the location of the problem relative to the phase shift and thereby simply relates to the overall probability of a solution for this problem region is definitely wrong. At both sides of the phase shift as well as in the phase shift region peaks tend to result in a plan. Thus based on our experimental data we must reject our first hypothesis. However, as the likelihood of peaks is so low and the runtime needed by peaks so high, canceling a planning process might still be a valuable option.

2. Different planners run into different peaks.
Unfortunately, this could not be shown in our experiments. Most often in the transition phase the planners behave similarly when running into peaks. So our first intention to simply swap between planners in the moment one planner needs too long seems not a promising strategy, at least not as long as no further structure about the problem space of compositions in smart environments is known. In addition, in nearly all our experiments one planner outperformed the other, i.e. blackbox (see Figure 2). So there seems currently little need to swap between planners. However, if we look closely we see that if the problem size increases, e.g. referring to number of goals, LPG might become a good alternative. Thus, with the number of services that are offered and the number of intentions that can be identified in smart environments, the benefit of selecting among different planners will increase as well.

3. Significant area with key characteristics.
Can we find a subset of the space of problems such that performance on this subset adequately characterizes a planner's performance on the space as a whole. This refers to the region of phase transitions, where the probability to find a solution drops from close to one to nearly zero. Concerning our data we can isolate a smaller subset of problems, wherein the standard deviation is high (dark/red) in order to identify hard problems (see matrix plots in Figure 3)[4]. The subset is located within the following ranges: $r < 10$ and $n < 50$. All remaining problems are characterized by a constant low standard deviation (bright/green).

8 Introducing thresholds for early canceling of planning runs

As mentioned under hypothesis 1, cancelling a planning process when it crosses a threshold might be an option. However, what does this mean for the runtime and possibly lost solutions? For the evaluation of the thresholds we only use the subset of problems with high standard deviation. The plots in Figure 4 show the changes in saved runtime as well as found solutions when increasing the threshold for each planner. As SGP was the slowest planner for our test set, it loses most solutions for low thresholds. Whereas even a threshold of 100 ms is not affecting the number of found plans of blackbox and LPG too much.

[4] Please notice that the scales of all matrix plots in this paper are adaptive and may differ between the plots due to the fact that we are interested in revealing tendencies. Thus, legends with absolute values are omitted.

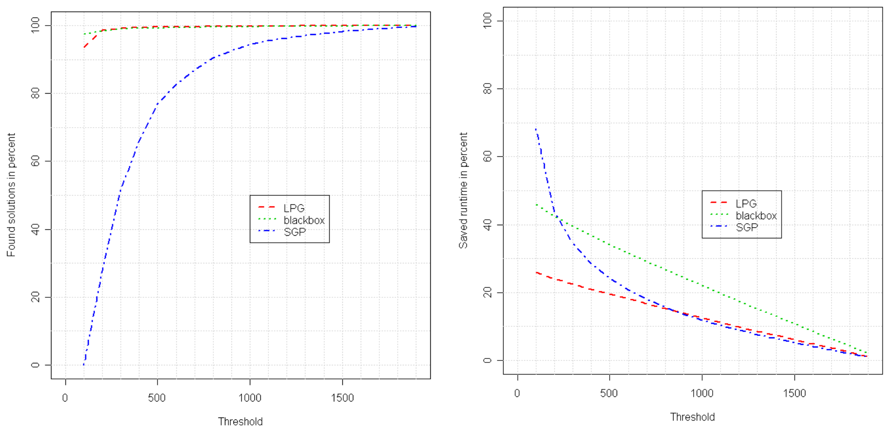


Fig. 4. Characteristic lines showing the quotas of found solutions and saved runtime depending on the threshold level

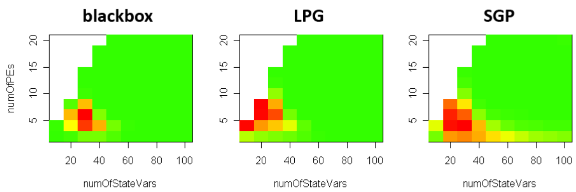


Fig. 3. Standard deviation of planning run times identifying the region of hard problems

We can obviously find a threshold where we can cancel planning runs saving a significant amount of runtime, without losing too many solutions (see Figure 4). Using 300 ms as threshold we save about 23% of runtime for LPG and about 40% for blackbox, while losing about 1% of solutions for each of the two planners. However no threshold value results in a reasonable rate of time saving and missed solutions for SGP. Due to the fact that the difference between the average behavior and the peak behavior (in particular for LPG and blackbox) is quite large, a heuristic that identifies a suitable threshold can save significant runtime, which is essential in smart environments.

As described above we conducted 100 replications of each single problem in our experiments. When comparing the diagrams of *avgSolution* and *stdTime* for the number of state variables (n) and the number of goals (g) in Figure 2 the phase transition characteristics are observable. When plotting the standard deviation of two variables in a matrix plot this behavior becomes more vivid (see Figures 5 and 6). Evaluating two variables we are also confronted with two phase shift directions. Hence this visualization allows comparing the different impact of both variables.

The probability for a solution drops in both figures from a high value (dark/red) to nearly zero (bright/green) proportional to the number of state variables n. It is furthermore proportional to the number of operators o and inverse proportional to the number of goals g.

9 Discussion and conclusion

To support a flexible composition of services in smart environments, planning algorithms appear particularly promising. Services, user intentions, and the given situation in smart environments can easily be mapped to AI planning. However, results of first studies showed that the execution time of planners for realistically sized smart environments behaves at some points erratically. Therefore, based on these first results, we defined three hypotheses for the application of planners in smart environments that deserve a more systematic and thorough analysis. To this end, a test suite with 55.000 problem configurations, several millions of runs, and three planners has been executed and evaluated.

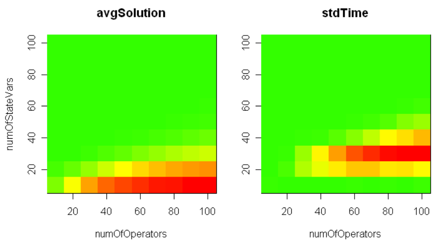


Fig. 5. Average found solutions and standard deviation of planner runtime (blackbox) indicating the phase transition defined by n and o.

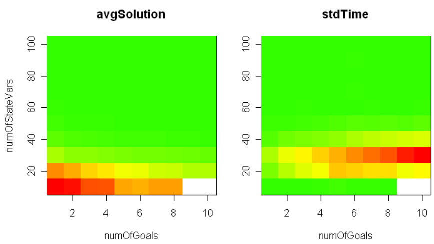


Fig. 6. Average found solutions and standard deviation of planner runtime (blackbox) indicating the phase transition defined by n and g.

The results confirmed our first observations that, whereas in general the planners are quite reliable, at certain times peaks occur that might endanger the unobtrusive service composition in smart environments. However, the other hypotheses that arose from our first observations could not be confirmed that easily revealing the value of large scale systematic experiments. Our first hypothesis, i.e. that if an execution peak occurs, it is rather likely that no plan can be found, could not be confirmed, in fact encountering a peak likely implies a solution. A simple strategy to keep the planning costs at bay could be devised though. Due to the low likelyhood of a peak to occur and its out of scale runtime, it still makes sense to cancel a planning run if its execution time crosses a certain threshold. For the planners in our experiment we can directly derive the size of the error implied by selecting a certain threshold. Therefore, characteristic lines can be used which depicts time savings due to canceling planners above thresholds and missed solutions. e.g selecting a threshold of 300 ms for blackbox and LPG a reasonable reductions of run time could be achieved at the cost of only 0.1 % missed solutions.

The experiments give a first indication how to calculate such a threshold on-line as well. As planners behave similarly, apart from rare occasions of peaks, only few plan executions would be required to empirically determine a suitable threshold for the smart environment and an arbitrary planning algorithm at hand. Neither our second hypothesis turned out to be true, i.e. that different planners run into different peaks. Thus, unfortunately to switch planners in case of overlong execution runs might not lead to better results in smart environments. The third hypothesis assumes the existence of a representative problem area for smart environments, so that the number of experiments can be reduced without loss of valuable information. The experiments indicated a possible reduction to the area of high variance where solutions exist. This area subsumes the area of the phase shifts, thereby confirming theoretical findings. Unfortunately, this area is still too large for executing on-line experiments in smart environments, and thus calculating characteristic lines on the fly is not feasible. In addition determine phase shift regions is not straight forward [20].

In our experiments blackbox almost always outperformed the other planners. However, in the more complex problem configurations of our experiments a decrease in the variance of LPG compared to blackbox can be identified. This indicates that for larger problems a specialization of planners occurs, as already observed in the experiments by Howe [13]. Thus, with the number of services and possible intentions to be considered in smart environments, the need to be able to switch between planners will be increasing.

Additional experiments are planned to possibly identify problem regions where switching planners is beneficial. However, our next steps will be to evaluate the found strategies in the composer module of our smart environment [17].

10 Acknowledgments

This research is supported by the German Research Foundation (DFG) within the context of the project MuSAMA (Multimodal Smart Appliance Ensembles for Mobile Applications).

References

1. S. Agarwal, A. Joshi, T. Finin, Y. Yesha, and T. Ganous. A pervasive computing system for the operating room of the future. *Mobile Networks and Applications*, 12(2-3):215–228, 2007.
2. F. Amigoni, N. Gatti, C. Pinciroli, and M. Roveri. What planner for ambient intelligence applications? *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, 35(1):7–21, 2005.
3. J. Bronsted, K. M. Hansen, and M. Ingstrup. A survey of service composition mechanisms in ubiquitous computing. *UbiComp 2007 Workshop Proceedings*, pages 87–92, 2007.
4. B. Brumitt, B. Meyers, J. Krumm, A. Kern, and S. Shafer. Easyliving: Technologies for intelligent environments. In P. J. Thomas and H.-W. Gellersen, editors, *Proceedings of the 2nd international symposium on Handheld and Ubiquitous Computing*, volume 1927 of *LNCS*, pages 12–29. Springer-Verlag London, UK, September 2000.
5. C. Burghardt and T. Kirste. Inferring intentions in generic context-aware systems. In *MUM*, pages 50–54, 2007.
6. T. Bylander. Complexity results for extended planning. In *Proceedings of the first international conference on Artificial intelligence planning systems*, pages 20–27, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

7. T. Bylander. A probabilistic analysis of propositional strips planning. *Artificial Intelligence*, 81:241–271, 1996.
8. P. Cheeseman, B. Kanefsky, and W. M. Taylor. Where the really hard problems are. In J. Mylopoulos and R. Reiter, editors, *Proc. of IJCAI-91*, volume 1, pages 331–337. Morgan Kauffman, 1991.
9. G. Chen and D. Kotz. Context aggregation and dissemination in ubiquitous computing systems. In *In Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications*, pages 105–114. IEEE Computer Society Press, 2002.
10. A. Gerevini and I. Serina. LPG: A planner based on local search for planning graphs. In *Proc. of AIPS'02*, Toulouse, France, 2002. AAAI Press.
11. M. Ghallab, A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld, and D. Wilkins. PDDL - the planning domain definition language, July 1998.
12. T. Heider and T. Kirste. Supporting goal based interaction with dynamic intelligent environments. In *ECAI*, pages 596–600, 2002.
13. A. E. Howe, E. Dahlman, C. Hansen, M. Scheetz, and A. V. Mayrhauser. Exploiting competitive planner performance. In *Proceedings of the Fifth European Conference on Planning*, pages 62–72. Springer, 1999.
14. V. Issarny, D. Sacchetti, F. Tartanoglu, F. Sailhan, R. Chibout, N. Levy, and A. Talamona. Developing ambient intelligence systems: A solution based on web services. *Automated Software Engineering*, 12(1):101–137, Januar 2005.
15. H. Kautz and B. Selman. BLACKBOX: A new approach to the application of theorem proving to problem solving. In *Working notes of the Workshop on Planning as Combinatorial Search*, pages 58–60, 1998.
16. F. Marquardt and A. Uhrmacher. Evaluating AI Planning for Service Composition in Smart Environments. In *Proceedings of the 7th International ACM Conference on Mobile and Ubiquitous Multimedia*, Umea, Sweden, December 2008.
17. F. Marquardt and A. M. Uhrmacher. An AI-Planning based Service Composition Architecture for Ambient Intelligence. Accepted for publication at AITAmI'09, July 2009.
18. J. Peer. Web service composition as AI planning - a survey. Technical report, University of St. Gallen, Switzerland, 2005.
19. J. Rao and X. Su. A survey of automated web service composition methods. In *Proceedings of the 1st International Workshop on Semantic Web Services and Web Process Composition, SWSWPC2004, LNCS*, San Diego, USA, 2004.
20. J. Rintanen. Phase transitions in classical planning: an experimental study. In *In Principles of Knowledge Representation and Reasoning*, pages 710–719. AAAI Press, 2004.
21. M. Roberts and A. Howe. Learned models of performance for many planners. In *Proc. of Workshop on Artificial Intelligence Planning and Learning*, Providence, Rhode Island, USA, September 2007.
22. K. Thurow, B. Göde, U. Dingerdissen, and N. Stoll. Laboratory information management system for life science applications. *Organic Process Research & Development*, 8:970–982, 2004.
23. K. Weihe. On the differences between "practical" and "applied". pages 1–10. 2001.
24. D. S. Weld, C. R. Anderson, and D. E. Smith. Extending graphplan to handle uncertainty & sensing actions. In *Proceedings of AAAI '98*, 15, pages 897 – 904. The AAAI Press, 1998.

Verfahren der Reaktiven Ablaufplanung

Birthe Gebhardt
Universität Oldenburg
Department für Informatik
26131 Oldenburg
b.gebhardt@scriptsite.de

Übersicht Der Einsatz von reaktiver Ablaufplanung in unterschiedlichen Einsatzgebieten wird immer wichtiger. Pläne werden aus wirtschaftlichen Gründen optimiert. Die Optimierung ermöglicht nur noch einen geringen Spielraum, um Störungen ohne Planänderungen abzufedern. Für die Auswahl eines geeigneten reaktiven Verfahrens müssen drei Fragen beantwortet werden: i) welche Strategie soll eingesetzt werden, ii) was ist der Auslöser für das Starten der Reparatur und iii) welche Korrektur Methode soll angewendet werden. Im Nachfolgenden werden diese Fragen näher betrachtet.

Einleitung

Die Aufgabe des Planens ist es, eine Folge von Aktivitäten zu ermitteln, die das Planziel abbildet. Ein Plan stellt die Aktivitäten, zugeordnet zu einer Anzahl von Ressourcen unter Berücksichtigung von Nebenbedingungen, dar.

Pläne besitzen durch die unterschiedliche Arten von Aktivitäten viele Formen, unter anderem die Aktionsplanung und die Ablaufplanung (Ghallab et al., 2004). Bei der Aktionsplanung werden Aktivitäten in eine Folge überführt. Die Folge stellt einen Plan dar, der durch die Aktivitäten einen Startzustand in einen Zielzustand überführt. Die auf den Faktor Zeit bezogenen Eigenschaften von Aktivitäten, werden bei der Aktionsplanung jedoch nicht berücksichtigt. Diese Art der Planung wird zum Beispiel in der Robotersteuerung eingesetzt (Sauer, 2004). Eines der bekanntesten Planungssysteme STRIPS (STanford Research Institute Problem Solver) wurde bereits in den siebziger Jahren entwickelt (Fikes & Nilsson, 1971).

Im Gegensatz zu der oben beschriebenen Planung, findet eine zeitliche Zuordnung von Aktivitäten zu Ressourcen in der Ablaufplanung statt. Ein klassisches Beispiel für diese Art der Planung ist in der Produktion zu finden. Aufträge werden auf Maschinen verteilt, um das Ziel der Produkterstellung zu erreichen. Produktionsumgebungen sind jedoch nicht statisch. Der Plan kann durch neue Aufträge oder einen Maschinenausfall verändert werden. Eine Umplanung wird erzwungen.

(Vieira et al., 2003) definiert die Korrektur eines Plans als einen Prozess zur Aktualisierung eines bestehenden Produktionsplans als Reaktion auf eine Störung oder Änderung. Dieser Prozess beschränkt sich nicht nur auf Aktualisierungen, sondern kann auch einen neuen Plan erzeugen und diesen durch den alten ersetzen. Die reaktive Ablaufplanung wird verwendet, um die Störung zu beheben.

In diesem Artikel werden die Möglichkeiten der Plankorrektur in der Ablaufplanung durch Störungen in Flugplänen erläutert.

Proaktive und reaktive Ablaufplanung

Ein Ablaufplan besteht aus einer Menge von Ressourcen, Aktivitäten und Regeln, die ein Planungsziel erreichen sollen. Eine Ressource ist die Einrichtung, die die Aktivität durchführt. In einem Flugplan wird als Ressource das Flugzeug definiert. Die Aktivitäten in einem Plan sind Aufträge, die von den Ressourcen ausgeführt werden. In unserem Beispiel ist ein Flug eine Aktivität auf der Ressource Flugzeug. Die Zuordnung für die Begriffe Ressource und Aktivität ist abhängig von der Domäne in der sich der Plan befindet.
Die Regeln stellen Beziehungen zwischen den Aktivitäten dar. Es wird zwischen zwei Arten von Regeln unterschieden i) Regeln die unbedingt erfüllt werden müssen und ii) Regeln die erfüllt werden können. Ein Plan, der alle Muss-Regeln erfüllt ist gültig und zulässig. Werden zusätzlich die Kann-Regeln eingehalten, ist der Plan gültig und konsistent. Muss-Regeln werden auch als Hard Constraints und Kann-Regeln als Soft Constraints bezeichnet.
In dem Beispiel des Flugplans kann als Hard Constraint die Regel, dass nicht mehr als ein Flug pro Flugzeug zur selben Zeit ausgeführt werden darf, definiert werden. Im Gegensatz dazu kann die Regel, dass ein Flug nicht mehr als 5 Minuten verspätet werden darf, als Soft Constraint angelegt werden. Diese Regel muss nicht eingehalten werden, um einen gültigen Plan zu erhalten.

Das Erstellen eines initialen Plans erfolgt in einer statischen Umgebung. In dieser Phase sollte bereits ein robuster Plan erstellt werden, der kleine Störungen (Regelverletzungen) ignoriert. Ein robuster Plan enthält Pufferzonen in dem Bereichen, die besonders anfällig für Störungen sind (Kohl et al., 2007). Kleine Störungen erzwingen keine Änderungen in dem bestehenden Plan. Der Vorteil eines robusten Plans ist es, dass die Abarbeitung des Plans reibungsloser und ohne größere Änderungen erfolgen kann.

Sind die Störungen gravierend, sodass der Plan ungültig wird, muss ein Plankorrektur Verfahren gestartet werden. Hierfür stehen zwei Ansätze zur Verfügung, abhängig davon, wann die Störung den Plan beeinflusst.
Eine proaktive Ablaufplanung berücksichtigt ungeplante Ereignisse, bevor diese als Störungen im Plan auftreten (Davenport & Beck, 2000). Das Ziel der proaktiven Techniken ist es, die Anfälligkeit gegenüber Störungen zu verringern und einen robusten Plan zu erstellen. Dieses Verfahren wird unter anderem bei der Einplanung von Wartungsintervallen von Flugzeugen durchgeführt.
Im Gegensatz dazu, wird eine reaktive Ablaufplanung durchgeführt, wenn der Plan bereits durch ein Ereignis ungültig geworden ist. (Sauer, 2004) beschreibt drei Zielsetzungen, i) schnelle Reaktion auf die Störung, ii) keine Verschlechterung der Gesamtqualität und iii) minimale Änderungen im initialen Plan.
Beide Verfahren versuchen einen neuen qualitativ hochwertigen Plan zu generieren, trotz der Ungewissheit der benötigten Ausführungszeit. Neben der Zeit, die für die Korrektur des Plans benötigt wird, ist ebenfalls die Art der Störung ein Faktor der nicht eindeutig vorhersagbar ist und als Ungewiss angesehen werden muss.

Störungen können durch unterschiedliche Ursachen auftreten, abhängig von der Domäne in der der Plan aufgestellt worden ist.
Im vorgestellten Szenario können unter anderem folgende Störungen im Flugplan auftreten:

- defektes Flugzeug
- Verspätung des Fluges durch schlechtes Wetter oder zusätzliche Tankstops
- Streiks auf Flughäfen, bei der Fluggesellschaften oder der Flugsicherung
- zusätzliche unvorhergesehene Flüge

Störungen sind schwer vorhersagbar und erzeugen unterschiedliche Reaktionen in dem Plan. Zusätzlich zu dem eigentlichen Störeffekt, der verursacht wird, kann eine Störung ungenaue Informationen besitzen. Der Zeitpunkt der Beendigung der Störung ist neben dem Startzeit die wichtigste Information für das Plankorrektur Verfahren. Diese Zeitpunkte geben an, welche Aktivitäten durch die Störung beeinflusst werden. Der unsichere Endzeitpunkt kann zu einem vorübergehend korrigierten Plan führen. Das Ausmaß der Störung dient als Indikator für den Einsatz eines Plankorrektur Verfahrens. Kleine Störungen werden ignoriert. Die Klassifizierung der Störung ist Domäne abhängig (J. W. HERRMANN, 2006).
Störung besitzt folgende Kriterien/Eigenschaften:

- schwer vorhersagbar
- unbekannte Auswirkung auf das aktuell gültige Planungssystem
- Ausmaß der Störung
- mögliche unsichere Informationen über die Störung

Die Durchführung eines Plankorrektur Verfahrens kann die Performance des Planungssystems verschlechtern. Andere Tätigkeiten, wie zum Beispiel die Eingabe von aktuellen Informationen zu den Aktivitäten oder die manuelle Eingabe von neuen Aktivitäten, können nicht ausgeführt werden, da das System in dem betroffenen Planungsbereich blockiert ist (J. W. HERRMANN, 2006). Dieser Punkt muss bei der Auswahl des geeigneten Verfahrens berücksichtigt werden, besonders wenn es sich um ein Echtzeitsystem handelt.
Das Ziel des Plankorrektur Verfahrens ist es, aus einem durch Störung verursachten ungültigen Plan einen gültigen und konsistenten Plan aufzustellen. Der ursprüngliche Plan soll weitestgehend bestehen bleiben. Änderungen sollen nur minimal anfallen. Eine Messmethode wird auf der Grundlage der Plan Stabilität in (FOX ET AL., 2006) dargestellt. Alle Änderungen zwischen dem ungültigen Plan und den anschließend gültigen Plan werden gezählt.

Im nachfolgenden Abschnitt wird nur die reaktive Ablaufplanung berücksichtigt.

Reaktive Ablaufplanung

Ein Plankorrektur Verfahren besteht aus drei einzelnen Bereiche, die im folgenden näher erläutert werden (J. W. HERRMANN, 2006). Die Bereiche des Verfahrens untergliedern sich in:

- Strategie
- Auslöser für den Start einer Plankorrektur
- Korrektur-Methode

Die Strategie unterteilt sich in drei Gruppen. Die Reparatur Strategie verändert den original Plan durch Hinzufügen und Entfernen von Aktivitäten. Bei der Umplanung Strategie wird ein defekter Plan von Grund auf neu erstellt (ARANGÚ ET AL., 2008). Die dritte Strategie, die Wiederverwendung von Planteilen, verändert nur einen Ausschnitt des Plans. Gültige Teile werden in den neuen Plan eingearbeitet (SZELKE & MÁRKUS, 1993).
Der zweite Punkt legt den Auslöser für das Starten einer Korrektur Methode fest. Hierzu werden zwei Ansätze verwendet, das dynamische und das periodische Vorgehen. Ein Trigger überwacht in Echtzeit Aktivitäten. Bei der Überschreitung von Grenzwerten startet die Plan Korrektur. Die zweiten Möglichkeit definiert Zeitintervalle, in denen der Plan auf Störungen überprüft wird.
Nach Auswahl der Strategie und des Auslösers muss die eigentliche Korrektur Methode definiert werden. Zum Einsatz kommen unter anderem Heuristiken, regelbasierte Ansätze, Fuzzy Techniken, Neuronale Netze, Suchverfahren und genetische Algorithmen. Auch Lösungsansätze aus dem Operational Research (OR) werden verwendet.

In (LUETHI ET AL., 2006) wird die Implementierung des Prozesses für die Reaktive Ablaufplanung im Zugverkehr beschrieben. Die Komplexität besteht in der Ausnutzung des Schienennetzes, der Pünktlichkeit der Züge und des Zusammenspiels der Zuggeschwindigkeit mit den Routen Informationen. Tritt eine Störung auf, wird diese auf definierte Bereiche (geografische, zeitliche) reduziert. Die Autoren haben drei wichtige Fragen für das Aufstellen des Verfahrens erarbeitet: i) wann soll die Korrektur gestartet werden, ii) ist der Korrektur Prozess unterbrechbar und iii) soll ein nicht validierter Zeitplan veröffentlicht werden. Trotz der offenen Fragen wurde gezeigt, dass durch den Einsatz eines Plan Korrekturverfahrens die Kapazität und die Stabilität in der Planungsdomäne verbessert worden ist. Neben dem eigentlichen Definieren des Korrektur Verfahrens ist die anschließende Kommunikation des neuen Plans unerlässlich.

Strategie

Für die reaktive Ablaufplanung werden drei Strategien eingesetzt: Reparatur, Umplanung und Wiederverwendung (CUSHING & KAMBHAMPATI, 2005) (ARANGÚ ET AL., 2008). Die Strategie bestimmt den Lösungsweg und die Auswirkungen auf den original Plan.

Bei der Reparatur des Plans werden Aktivitäten hinzugefügt bzw. entfernt, um einen gültigen Plan zu erhalten (VAN DER KROGT & DE WEERDT, 2005)(ARANGÚ ET AL., 2008). Das Entfernen bezieht sich nicht nur auf Aktivitäten, sondern auch auf die Regeln von Aktivitäten. In (VAN DER KROGT & DE WEERDT, 2005) wird der Ansatz verfolgt, durch Aktivierung und Deaktivierung von Regeln einen gültigen Plan zu erstellen und damit einen ungültigen Plan zu reparieren. Die Zuordnung der Aktivitäten zu den Ressourcen und die Zeitangaben bleiben hierbei im Plan unverändert. Im Beispiel des Flugplans kann die Regel für die Reaktion auf Verspätungen von Flügen deaktiviert bzw. die Grenzwerte für das Auslösen der Regelverletzung verändert werden.

Die Verwendung der Umplanungs Strategie erstellt einen neuen Plan von Grund auf neu. Die Frage, die sich unter anderem beim Einsatz dieser Strategie stellt, ist ob genügend Zeit vorhanden ist den ungültigen Plan mit dieser Strategie umzuplanen. Je nach Größe des Plans, gekennzeichnet durch die Anzahl von Aktivitäten und

Ressourcen über einen Zeitraum, ist der Einsatz dieser Strategie geeignet. Der Zeitraum wird bestimmt durch den Aufbau des Plans unter dem Aspekt eines Ressourcen basierten Plans oder eines Planziel basierten Plans. Ein Planziel basierter Plan kann zum Beispiel ein Projektplan sein, der die einzelnen Aktivitäten in einem Projekt auflistet und die Ressourcen zuordnet. Nach Beendigung des Projektes ist ebenfalls der Plan beendet, obwohl die Ressource weiter verfügbar sein kann. Bei einem Ressourcen basierten Plan endet der Plan mit dem Deaktivieren der Ressource. Diese Art von Plan kann über eine längere Zeitdauer verfügen als bei einem Planziel basierten Darstellung.
Der Flugplan wird als Ressourcen basierter Plan in einem Balkendiagramm dem Anwender angezeigt. Dieser Plan startet mit der Gründung einer Fluggesellschaft und Endet mit deren Auflösung. Die Strategie der Umplanung ist nur einsetzbar, wenn ein Zeitfenster in dem Flugplan definiert wird, dass vollständig neu geplant werden soll.

Die dritte Strategie befasst sich mit der Wiederverwendung von gültigen Teilen aus dem initialen Plan. Ziel ist es, möglichst viele Aktivitäten unverändert aus dem initialen Plan in den neuen Plan zu übernehmen, um möglichst wenig Modifikationen für einen neuen gültigen Plan durchführen zu müssen (Szelke & Márkus, 1993)(Vieira et al., 2003).
Diese Strategie wird in dem Beispiel der Flugplanung verwendet. Der Flugplan besitzt viele Abhängigkeiten zu anderen Bereichen, unter anderem zu der Crewplanung. Finden Änderungen im Flugplan statt, kann es zu Änderungen in der Crewplanung kommen. Durch die Abhängigkeit zwischen Flugplanung und Crewplanung ist das Ziel, Modifikationen im Flugplan gering zu halten.

Um eine geeignete Strategie für eine Domäne einzusetzen, sollten die Randbedingungen wie Plan Stabilität und Qualität untersucht werden.
(Fox et al., 2006) hat die Plan Stabilität zwischen Reparatur und Umplanung untersucht. Die Untersuchungen ergaben, dass der Einsatz der Reparatur bevorzugt wird, sobald die Plan Stabilität berücksichtigt werden muss. Die Stabilität kann eine entscheidende Rolle spielen, wenn Abhängigkeiten zu dem Plan bestehen. Die Anzahl der Änderungen kann in abhängigen Plänen ansteigen, wenn die Anzahl der Änderungen für die Korrektur des ungültigen Plans ansteigt.

Auslöser

Der Auslöser in einem Plan Korrektur Verfahren bestimmt, wann die Korrektur Methode gestartet werden soll. Es werden drei Arten von Auslösern verwendet (J. W. Herrmann, 2006). Für eine Echtzeit Reaktion wird ein dynamischer Mechanismus mittels Trigger verwendet. Sobald eine Störung den Plan beeinflusst, zündet der Trigger und startet die Korrektur Methode. Bei dem Einsatz eines periodischen Auslösers wird der Plan in einem Zeitintervall auf Störungen überprüft. Die Korrektur erfolgt zeitversetzt zu dem Eintreffen der Störung. Die dritte Art ist eine Kombination aus den beiden zuvor erwähnten Auslösern. Der Hybrid Auslöser verwendet überwiegend die periodische Variante und nur bei bestimmten Störungen wird ein Trigger ausgelöst. Die Störungen werden priorisiert, um die geeignete Art des Auslösers zuzuordnen.

Nicht nur die Störung selber kann für die Auswahl herangezogen werden, auch der Zeitpunkt, an dem die Störung auftritt, muss berücksichtigt werden. Je näher die Störung der aktuellen Zeitlinie kommt, umso schneller muss gehandelt werden.

Anhand von Kriterien können in einem Plan Zeitabschnitte definiert werden, die angeben, welche Art von Auslöser verwendet werden soll. Als Kriterium wird der Zeitpunkt der Erstellung eines neuen Plans definiert. Zwei solcher Punkte bilden ein Zeitabschnitt. Dieses Verfahren wird in (PFEIFFER ET AL., 2008) beschrieben.

Das Finden einer geeigneten Lösung ist bereits in diesem Punkt des Plankorrektur Verfahrens schwierig. Die Fragen, wann die Korrektur Methode gestartet und beendet bzw. abgebrochen werden soll und wie sich das System verhalten soll, wenn während eines Korrektur Ablaufs weitere Störungen das System destabilisieren, sind nur zwei von den Faktoren, die mit berücksichtigt werden müssen (LUETHI ET AL., 2006).

In dem Beispiel für diesen Artikel wird der Flugplan in Zeitabschnitte unterteilt. In dem zeitkritischen Abschnitt '*der aktuelle Tag*' wird ein Trigger basierter Auslöser verwendet. Die Korrektur in dem Plan kann Einfluss auf Flüge haben, die außerhalb dieses Zeitfensters liegen. Das wird bewusst in Kauf genommen. Außerhalb dieses Intervalls wird ein periodischer Auslöser verwendet bzw. die Störungen werden dem Anwender nur angezeigt, ohne eine Reaktion auszulösen. Das hat den Vorteil, dass Pläne, die von dem Flugplan abhängig sind, koordiniert verändert werden. Dieses Verfahren minimiert die Korrektur Aktivitäten in abhängigen Plänen.

Korrektur-Methode

Die Korrektur Methode besteht aus einem oder mehreren Algorithmen, die den ungültigen Plan in einen gültigen Plan überführen. Der zuvor beschriebene Auslöser startet die Korrektur Methode.

Wie in (SAUER, 2004) beschrieben, können verschiedene Korrektur Methoden aus den unterschiedlichsten Techniken verwendet werden. Zu den Techniken gehören unter anderem die Zeitenverschiebung von Aktivitäten, die Verwendung von Heuristiken und regelbasierten Ansätzen, Fuzzy-Techniken, Neuronale Netze und genetische Algorithmen. Ebenso das Hinzufügen bzw. Entfernen von Aktivitäten in Plänen wird als Lösungsansatz verwendet.
Im nachfolgenden werden Beispiele aus dem Transportbereich erläutert.

In (LIM ET AL., 2005) wird das Lösen der Flugsteig Zuordnung zu einem Flugzeug bearbeitet. Für das Zuordnungsproblem wird ein Tabu Search Algorithmus mit den Nachbarschaftsfunktionen *Insert Move* und *Interval Exchange Move* verwendet. Diese Funktion verschiebt die Flüge auf der Zeitachse nach Links bzw. Rechts um eine Überlappung zu verhindern. Nicht nur Zeitverschiebungen werden mit diesem Algorithmus bearbeiten, auch das Hinzufügen von Flügen in dem Plan wird mit dem Tabu Search Algorithmus gelöst. Hierbei wird im ersten Schritt eine Lücke in dem Plan gesucht, anschließend werden die umliegenden Flüge auf der Zeitachse verschoben, sodass der neue Flug eingepflegt werden kann.

Auf die Korrektur von Flugplänen wird in dem Artikel (LOVE ET AL., 2002) eingegangen. Die realisierten Heuristiken werden auf ein Netzwerk angewendet, dass sich aus dem Flugplan extrahieren lässt. Verwendet wird die Zeitachse, alle Flughäfen, Flüge und Flugzeuge. Mehrere Heuristiken wurden implementiert, um eine Lösung innerhalb von 3 Minuten zu erhalten. Heuristiken wie Iterated Local Search (ILS), Steepest Ascent Local Search (SALS) und Repeated

SALS wurden in die Lösung implementiert. Der SALS Algorithmus benötigte die geringste Zeit um ein lokales Optimum zu finden.

Eine Auflistung von Flugzeug Störungen und deren Auswirkungen wird in(CLAUSEN ET AL., 2009) behandelt. Lösungsansätze wie der Branch-and-Bound Algorithmus, Tabu Search oder der Greedy Algorithmus werden eingesetzt. Aus dem Operational Research (OR) wird die Lineare Programmierung verwendet.

Ausblick

Auf der Basis der dargestellten Punkte für das Verfahren der reaktiven Ablaufplanung lassen sich unterschiedliche Kombinationen realisieren. Eine Analyse der Planungsdomäne ist notwendig, um das optimale Plan Korrektur Verfahren zu finden. Eine einzige Korrektur Methode kann es durch das jeweils spezifische Einsatzgebiet nicht geben, da bereits in den aufgezeigten Beispielen verschiedene Korrektur Methoden verwendet worden sind. Es ist wünschenswert, dass in Systemen mehrere Verfahren zur Verfügung stehen (SAUER, 2002).
Für die Auswahl in den einzelnen Bereichen des reaktiven Ablaufplanung Verfahrens müssen unter anderem folgende Fragen geklärt werden: Bei welchen Störungen soll welches Verfahren aus der Sammlung eingesetzt werden? Wann soll die Korrektur Methode gestartet bzw. unterbrochen werden?
Diese und weitere Fragen sind in diesem Zusammenhang bei weiteren Untersuchungen zu klären.

Zusammenfassung

In diesem Artikel wurde aufgezeigt, das ein Plan Korrektur Verfahren nicht trivial ist. Plan Korrektur Verfahren beschreiben wie ein Plan generiert und geändert werden kann (J. W. HERRMANN, 2006).
(CLAUSEN ET AL., 2009) verdeutlicht, dass die Auswahl der Korrektur Methode abhängig von der Komplexität und des Zieles des Plans ist. Die Auswahl in den einzelnen Bereiche (Strategie, Auslöser und Korrektur Methode) des Verfahrens ist abhängig von der verwendeten Domäne.
Für den Einsatz eines Plan Korrektur Verfahrens als ein Modul in einem IT System muss eine Strategie, ein Auslöser und eine Korrektur Methode definiert werden. Ob schließlich eine einzige Lösung für ein Problem in der Domäne gefunden werden kann oder eine Kombination eingesetzt wird, ist im Einzelfall zu prüfen.

Literaturverzeichnis

ARANGÚ, M. ; GARRIDO, A. ; ONAINDIA, E.: A General Technique for Plan Repair, IEEE Computer Society Washington, DC, USA, 2008, pp 515-518

CLAUSEN, J. ; LARSEN, A. ; LARSEN, J. ; REZANOVA, N.: Disruption management in the airline industry—Concepts, models and methods. In: *Computers and Operations Research* (2009)

CUSHING, W. ; KAMBHAMPATI, S.: Replanning: A new perspective, Proc. of ICAPS, 2005

DAVENPORT, A. ; BECK, J.: *A survey of techniques for scheduling with uncertainty*. URL http://www.eil.utoronto.ca/profiles/chris/chris.papers.html. - Unpublished manuscript, 2000

FIKES, R. ; NILSSON, N.: STRIPS: A new approach to the application of theorem proving to problem solving. In: *Artificial intelligence* vol. 2 (1971), Nr. 3/4, pp 189-208

FOX, M. ; GEREVINI, A. ; LONG, D. ; SERINA, I.: Plan stability: Replanning versus plan repair, AAAI Press, Proc. ICAPS, 2006, pp 212-221

GHALLAB, M. ; NAU, D. ; TRAVERSO, P.: *Automated Planning: theory and practice* : Morgan Kaufmann Publishers, 2004 – ISBN 1-55860-856-7

HERRMANN, J. W.: Rescheduling Strategies, Policies, and Methods. In: *INTERNATIONAL SERIES IN OPERATIONS RESEARCH AND MANAGEMENT SCIENCE* vol. 89 (2006), p 135

KOHL, N. ; LARSEN, A. ; LARSEN, J. ; ROSS, A. ; TIOURINE, S.: Airline disruption management—Perspectives, experiences and outlook. In: *Journal of Air Transport Management* vol. 13 (2007), Nr. 3, pp 149-162

VAN DER KROGT, R. ; DE WEERDT, M.: Plan repair as an extension of planning, Proc. of the Int. Conf. on Automated Planning and Scheduling, 2005

LIM, A. ; RODRIGUES, B. ; ZHU, Y.: Airport gate scheduling with time windows. In: *Artificial Intelligence Review* vol. 24 (2005), Nr. 1, pp 5-31

LOVE, M. ; SORENSEN, K. ; LARSEN, J. ; CLAUSEN, J.: Disruption management for an airline-rescheduling of aircraft. In: *Lecture Notes in Computer Science* (2002), pp 315-324

LUETHI, M. ; WEIDMANN, U. ; LAUBE, F. ; MEDEOSSI, G.: Rescheduling and Train Control: A New Framework for Railroad Traffic Control in Heavily Used Networks, 86th Transportation Research Board Annual Meeting, 2006

PFEIFFER, A. ; KÁDÁR, B. ; MONOSTORI, L. ; KARNOK, D.: Simulation as one of the core technologies for digital enterprises: assessment of hybrid rescheduling methods. In: *International Journal of Computer Integrated Manufacturing* vol. 21 (2008), Nr. 2, pp 206-214

SAUER, J.: Intelligente Planungsverfahren. In: *Industrie Management* vol. 18 (2002), Nr. 6

SAUER, J.: *Intelligente Ablaufplanung in lokalen und verteilten Anwendungsszenarien* : Teubner, 2004 – ISBN 978-3-519-00473-8

SZELKE, E. ; MÁRKUS, G.: Reactive scheduling-An intelligent supervisor function, North-Holland Publishing Co. Amsterdam, The Netherlands, The Netherlands, Proceedings of the IFIP TC5/WG5. 7 International Workshop on Knowledge-Based Reactive Scheduling, 1993, pp 125-146

VIEIRA, G. ; HERRMANN, J. ; LIN, E.: Rescheduling manufacturing systems: a framework of strategies, policies, and methods. In: *Journal of Scheduling* vol. 6 (2003), Nr. 1, pp 39-62

Online Planning: Challenges and Lessons Learned

René Schumann[1] and Jürgen Sauer[2]

[1] Goethe University, Information Systems and Simulation,
Robert-Mayer-Str. 10, 60325 Frankfurt am Main, Germany
reschu@informatik.uni-frankfurt.de
[2] Carl von Ossietzky University, Business Engineering,
Ammerländer Heerstr. 114-118 26129 Oldenburg
juergen.sauer@uni-oldenburg.de

Abstract. Shifting the focus from offline to online planning implies changes that increase complexity tremendously. First of all the optimality can only be proven ex-post. Events that trigger planning actions must be abstracted from primitive data that can be collected, e.g. by RFID bulk scans. Moreover effects like shop floor nervousness, where no stable plan exist, must be avoided. A key aspect in this challenge is the notion of robustness. On the one hand robustness reduces the number of needed repair steps but on the other hand robustness is in conflict with the objective function and can only be computed ex-post. In this article we point out the aspects mentioned above and survey how these issues are tackled in fields of planning and scheduling in artificial intelligence, so far.

1 Introduction

In this paper we discuss specific challenges that arise when dealing with dynamic scenarios in planning and scheduling. Thereby we focus especially on the challenges in transportation planning and shop floor scheduling. Even if these application domains differ the resulting problems are shifting the scope from a static environment to a dynamic one are quite similar. In both cases it is assumed that a plan exists and that the plan execution deviates from the plan. The deviation is typically indicated by an event that arises and changes the environment. If the existing plan is not feasible for the changed environment it has to be repaired, that is typically the task of online planning or reactive planning. In the following the terms online planning and reactive planning / scheduling will be used synonymously. The goals of reactive planning are typically threefold [6]:

1. The reaction to an event should be fast. The current plan should become feasible quickly. This is motivated for two reasons. First the plan execution should not be stopped (to long) until a new valid plan is available. Ideally the execution is going on, while the plan for future activities is adopted. Second in a dynamic environment, like the shop floor or the transportation domain, events are rather frequent. If the system would react to slow the "new" plan is overtaken by reality as it is computed.

2. The existing plan should be widely conserved. This is required as it is intended that steps that have already been performed, should be considered in the new plan, if possible. So work done so far should not be discarded. Moreover this enforces plan stability that is important for the coordination of the planning with other planning problems, like procurement, and for the reputation of the planning systems.
3. The third goal is to maintain the plan's quality. A key motivation for using enhanced planning systems is, of course, the ability to compute plans of good quality in a short time. Lower evaluated plans lead to inefficient transportation or production and in consequence to operational loss of the company. It goes without saying, that this has to be avoided. Therefore to be practical applicable planning systems have to perform at least acceptable and comparable to human planners in dynamic environments and this is measured in plan quality.

As mentioned above it becomes clear now, that these requirements are partially contradicting. Computing a good plan, according to an objective function, and having a fast reaction to an event is obviously hard to achieve. And conserving large parts of a plan while maintaining its quality can become hard, as well.

Currently the research about online planning, even if its established for some years (see [21] for a review of early work in transportation or [29] for job shop scheduling) is still in an initial constructive phase of research. Approaches based on techniques like local search [29, 23], metaheuristics [7, 17], multiagent systems [8, 12, 16] or hybrid approaches [31] are presented. In consequence a variety of different approaches exist for online planning.

If one have to select a technique for a given problem the selection of one approach from given set of techniques becomes a hard task. This is because of different scopes of what online planning compromises, a lack of benchmarks and metrics.

In fact what is needed are means that can help to compare different online planning approaches. This comparison should be a multidimensional one. It is not enough, and in dynamic environment typically not feasible, to find that method a is superior to method b. Which method performs better typically depends on characteristics of the application domain and on characteristics of the changes within the environment. Thus for a more engineering oriented approach it would be necessary to identify those characteristics and evaluate the abilities of existing methods according to them.

In the following we are first clarifying the terminology used in different fields. This is especially important as different terms are used interchangeable even if they may have different semantics. In section 3 we discuss current challenges that have to be faced towards a more engineering approach for the design and application of online planning techniques. Finally we summarize our hypothesis and outline future work.

2 Existing terminology and techniques

2.1 Terminology

The problem of keeping existing plans in a dynamic environment valid is addressed in different fields of research and applications. The fields addressed here are mainly operations research and artificial intelligence. The applications mentioned here are vehicle routing problems and (job-shop) scheduling problems. Different terminologies have been established and terms are used partially interchangeable. Therefore it is necessary to clarify the terminology. As already mentioned the terms online planning and reactive planning are used here synonymously. It can be argued that online planning is the more general term, as it compromises least commitment planning. Such approaches were presented e.g. by [20] and [27]. Interestingly methods based on least commitment strategies do not play an important role in current research, even if they are used in practical applications [9]. But for the purpose of this paper we exclude this field and use those terms interchangeable.

Technically there exist two major approaches how reactive planning can be implemented [10]:

- Plan repair: An existing plan is going to be adopted to a changed situation. Approaches are presented in [29, 2].
- Replanning: If an event occurs the existing plan is discarded and a new plan is computed from scratch [5].

Both approaches have their strengthes and weaknesses. Especially concerning the goals of reactive planning, mentioned above. From the perspective of complexity it has been shown that both face the same structure problems [15]. A comparison emphasizing stability can be found in [10].

It has to be mentioned that this terminology outlined here is not exclusiv. There are different naming schemes in use. For example in [33] the most general term used is rescheduling. The authors present an interesting framework for rescheduling approaches, that can be classified in their framework. Thereby they distinguish between dynamic scheduling and predictive-reactive scheduling. Dynamic scheduling is characterized where no plan exist a priori, like in least commitment strategies. Furthermore the authors mention that other names for dynamic scheduling are online scheduling or reactive scheduling.

2.2 Techniques for online planning

As previously outlined we can identify different techniques for online planning. Therefore we can identify different technologies for plan repair and replanning, even if there exist technologies that can be applied for both approaches, e.g. metaheuristics.

In the following we are going to discuss techniques that have been applied to plan repair. Approaches based on local search techniques apply plan modifications, like shifting dispatched operations in the future (right shift) or pull them backwards in time (left shift). A typical approach using those techniques is implemented in the OPIS system presented in [29].
Based on local search techniques, metaheuristics can be applied to plan repair as well. A counterargument therefore is the longer reaction time. But it has to be stated, that metaheuristics strive fast to good solutions, so that computation can be interrupted at a given time and results can be of good quality. A typical trend of plan's quality over time is sketched in figure 1. An early approach using tabu-search for instance is [7].

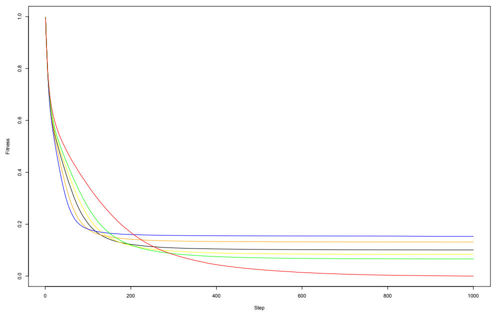


Fig. 1. Characteristic plan's quality trend over time, using different parameter sets

Other approaches, that has been applied to plan repair are multiagent systems. Using the characteristics that agents can adopt their behavior and their plans to their environment. Often orders and resources are represented by agents. Agents negotiate about the assignments, while typically orders want to be processed as quickly as possible while resources want to maximize their utilization. Examples of agents in dynamic environments in the manufacturing domain can be found in [4, 18, 25, 34]. Application in the transportation domain can be found in [8, 20] for example.
Of course hybride approaches are possible as well. Where techniques from multiagent systems and evolutionary approaches are combined, e.g. [30].

For the field of replanning different options are possible, and applied, as well. A complete replanning compromises the generation of a new plan for all non executed activities. In contrast in partial replanning only a subset of activities is rescheduled. Other parts of the plan remain fixed and needed resources are marked as blocked. After a new schedule has been computed, which can be done faster as the initial computation, because the number of activities to be scheduled have been reduced, the unchanged elements of the plan and the newly computed plan can be joined. The quite common technique to remove the orders that are

affected by an event, update those data and the integrate those events as new orders thus is a special case of partial replanning. According to [19] complete replanning leads to better plan's quality than partial replanning.
Of course looking at aspects, like plan stability, complete replanning is problematic, as the plan can change with each event completely. One can try to soften those changes by adding new constraints to the planning problem, that fixes parts of the plan in favor of stability. This decreases typically the ability to find feasible plans at all, because it is unknown to what degree similarity can be archived at all.

Another approach that is currently on its way into practical application is based on a less intelligent or optimizing way, but is a more interactive approach. Simulation techniques integrated in manufacturing control stations, typically incorporated in manufacturing execution systems (MES), allow the human operator to evaluate possible reaction options quickly. Sadly the aspect of interactivity is often ignored in research done in operational research and artificial intelligence. But it has to be mentioned that it is an important aspect to gain confidence in the system and its reaction to disturbing events.

Another discussion about applied methods for reactive planning with further references can be found e.g. in [19].

3 Challenges in online planning

In this section we outline a number of points that arise in planning in dynamic environments independently from the terminology and nearly independently from the application domain.

3.1 Optimizing without knowing where to go

A first fact that has to be admitted is that optimizing in a dynamic environment is not possible. The optimal solution can only be computed ex-post. Thus during the execution finding the optimal solution is comparable to guessing. This is of course a formulation that exaggerate the actual state. But it is in fact true that a lot of plan repair strategies, especially those who try to optimize the new generated plan, rely on the ceteris paribus assumption. That is that the existence of events in the future is neglected. This is for nearly all events (expect the last one) not true. And therefore the resulting plan is typically suboptimal. This makes a perspective shift necessary, especially from the classical perspective of operations research, here one typically wants to find the optimal solution.

3.2 What is dynamic and how can it be recognized?

In nearly all models for reactive planning the events are triggered by events, that change the environment, and may invalidate the current plan[3]. Those events are often abstractions from data collected in reality. Data collection is nowadays widely automated. Techniques based on GPS or RFID are used. For instance a bulk scan of RFID tagged products at the outgoing gate of a crossdocking station can only be interpreted as a certain number of products, leaving a certain place at a certain time. The conclusion that an order is late, is an abstraction of those information, that can be derived by data collected by the bulk scan and further context information. What is needed is a vertical data integration [26]. Information gathered by PDA (production data acquisition) systems or corresponding systems in the transportation domain is very fine grained. A hierarchy of different abstraction levels have to be derived regarding the context of the given production situation at hand.
As it becomes clear so far, the reactive planning systems have to be integrated in the existing IT structure of companies. Especially the integration with the aforementioned PDA systems and a sound abstraction of data collected by this system is a necessity to implement an applicable reactive planning system. Because otherwise deviations in the execution from a given plan could not be identified. On the other hand specialized triggers have to be realized that allow an integration within the existing ERP system. Because those systems are another source of dynamic information. Information concerning new, changed or canceled orders or resources are typically maintained in the ERP system.
Another very promising approach is complex event processing, which is designed to automated process a high number of events, modeled as an event-flow, with up to some 100.000 events per second, as it is generated by systems based on PDA or RFID techniques [14]. In complex event processing those raw, fine-grained events can be aggregated, efficiently.

3.3 React but not overreact

Assume we can surely identify events that change the environment. An event can either have the potential to influence the plan in a positive or negative way. Note that an event does not have to effect a plan at all. For instance a resource break down of a resource that is not scheduled in the current plan will not effect the plan at all; even if the environment has changed.

An example of a negative effect on an existing plan is the arrival of a new order. Typically it is encoded as a hard constraint that all existing orders have to be concerned in the plan. Thus this event has obviously negative effects to the plan, as it becomes invalid. Nevertheless, the new plan might be of lower quality, as well.

[3] This is not the only way to handle plan adaption in dynamic environments, of course. In their framework Vieira et al. [33] classify the approaches into event-driven or periodic. In a periodic approach the plan is regularly updated regarding a rolling time horizon.

An example of an event that can have positive effects on a plan is a withdraw of an existing order. There are at least three options how to react on this event.

- Ignore the event completely and do not change the plan, at all. Of course there are unnecessary actions within the plan, but commonly the plan will not be decreased[4].
- Delete the assignments needed to fulfill the withdrawn order. Thereby you can not decrease your plan. It will become eventually better, if the fulfillment of this order has evaluated negatively itself, i.e. the fulfillment of this order was delayed.
- Delete the assignments needed to fulfill the withdrawn order. And evaluate if other assignments in the existing plan can be moved to improve the plan's quality. Thereby trying to use the freed resources of the withdrawn order to improve the fulfillment of the remaining orders.

Actually it is an open question to what events an online planning system has to react. One would expect that online planning reacts when a plan becomes invalid. But a literature review shows that this is not common ground, so far. For example in [13] only the event of new customers is regarded, which is not the only event that can invalidate a plan. The set of regarded events in transportation planning was extended e.g. in [28] to at least five different events. The number of events that can occur is even higher in the job shop scheduling domain, for example Reheja et al. [22] list 17 different events that can occur in their model. It is arguable only to react to negative events, as the plan quality is typically decreased by those events. And the potentials to improve a plan, as a result of other events is not used.

Reacting to events means typically change the current plan, which can decrease the stability of a plan. It is only *can* here, because a plan adoption might save future adjustments of the plan, as a consequence of upcoming events in the future, which depends on the characteristics of future events and the event handling strategy.
Current discussions on plan stability can be found in [10, 19].

3.4 Robustness: way out or digging in?

Closely related to plan stability is the term of robustness or robust plans. Robustness is commonly seen as the means to achieve plan stability. A plan is called robust if it "is likely to remain valid under a wide variety of different types of disturbance" [7]. Robustness is typically achieved by allowing the potentials to incorporate upcoming events without changing the already planned activities. A typical technique therefore is to use buffers or slacks in the plan, reduce resource capacity or increase operation durations artificially in the original plan. Of course this means all decrease typically plan quality. Therefore a tradeoff

[4] Expect you have some contribution margin elements in your objective function, which is rather infrequent.

between plan's quality and robustness has to be made. Finding a tradeoff is complicated by the fact that robustness cannot easily be measured.

Whether a plan is robust or not depends heavily on the kinds and sequence of the occurring events. There can exist a sequence of events that can all be incorporate without changing the plan even one time, while another sequence of events might cause plan repair after each event, starting with exactly the same plan. Thus measuring robustness depends essentially on the events that are assumed to occur. As this is typically unknown different distributions of event sequences have to be explored. Thereby the sequence can be changed according to the

- different kinds of events that can occur,
- the percentage rate for each kind of event,
- the ordering of the events and
- the time when the events occur.

Moreover robustness and plan stability do not only depend on the sequence of events but on the current situation within the execution system, e.g. the shop floor, as well [24].
So the degrees of freedom while measuring robustness and consequently plan stability are comparable high and it becomes hard to find reasonable metrics. And even if there exist such a metric that indicates that a given plan is robust. There is always the potential that while executing this plan a lot of adoption becomes necessary caused by an event sequence that was assumed rather unlikely during the evaluation of the plan.
Speaking about robustness without indicating towards which events is misleading, too. As there most often exist a sequence of events that will result in an instable plan. What can be achieved is robustness against a certain type of event or defined event mixes.

3.5 Evaluation: Or why my system is always best

As it becomes clear so far, evaluating online scheduling systems is a hard task to accomplish. Thereby nearly every presented approach is evaluated. But most often this is done using a specialized scenario that mainly focus on specific characteristics of the presented method. With the result that most approaches claim to have some unique advantages in comparison to other existing methods.

Surveying existing papers about topics like online planning or reactive planning shows that only a few people deal with a more systematic approach to compare and evaluate existing approaches e.g. [19, 20, 24, 32]. Technically all those approaches base on simulation as primary methodology for evaluation of online planning systems (see also [1, 6, 19, 20, 32]). In doing so the planning system is used within a simulation environment that simulates the dynamic environment and rises the events. Thereby it has to be mentioned that even if the systematic is commonly agreed on, there exist no tool or system that is dominate in use. Even if there exist suggestions of testbeds [20, 32] those are not widespread either.

At least four objective reasons can be identified for the lack of work that characterize and evaluate online planning systems.

- Online planning has been identified as important. A consequence of the terminological mess, the different fields of research that are interested in this topic and the different application domains it is hard to survey relevant work, and maybe find appropriate existing methodologies.
- There exist no common ground on what online planning typically comprises. So which set of events should be handled, which functionalities an online planning tool should offer. This is at least one requirement to compare different approaches. As mentioned before the number of events that can be handled by different approaches vary from one to at least 17.
- A consequence of the aforementioned aspects is, that there exist no commonly accepted benchmark scenarios. And as outlined before a lot of approaches are not strive to be comparable or applicable to scenarios already existing in the literature, as they highlight to handle situation that are special in their scenario. Another challenging aspect is the high degree of freedoms that already have been mentioned by modifying a sequence of events. Expect using random functions there exist rarely systematic testing approaches for generating event sequences. Thereby those sequences may have some random elements but on the other hand allow exact replications of simulation runs that are needed to compare different approaches in identical situations.
- We have a lack of metrics. We have currently no adequate metrics that are commonly accepted either for
  - dynamic itself [13, 28, 11],
  - stability [19] or
  - robustness

  Metrics based on a competitive analysis [3] require a comparison to the optimal solution. But in most cases of job shop scheduling or transportation planning those optimal solutions are unknown and cannot be computed efficiently. Thus those analysis are only of theoretical interest.

4 Discussion

In this paper we have addressed challenges that currently arise in online transportation planning. We identified similarities between these planning problems and problems arising in dynamic scheduling domains. It has to be pointed out that planning in dynamic environment has a common ground, that have to be explored in more depth. So far research is in an early stage. More work seems to be investigated in finding of new methods than in comparing and evaluating existing techniques. As a reason for this we state that important fundamentals have to be investigated before a reasonable evaluation or comparison of online planning can be achieved.

Our hypothesis is that a common understanding of what online planning should comprise, common benchmarks and widely accepted metrics are needed.

This could lead to a more engineering oriented catalog and characterization of existing and upcoming online planning approaches. Thereby the evaluation of methods will be regarding different aspects. It is not enough, and in dynamic environment typically not feasible, to find that method a is superior to method b. Which method performs better typically depends on characteristics by the application domain and on characteristics of the changes within the environment. Therefore for a more engineering oriented approach it would be necessary to identify those characteristic and evaluate the abilities of existing methods according to those characteristics. A step towards that approach is for example [24].

Aspects discussed in this paper would lead to a more coherent, sound and although more engineering view of the field of online planning. Thereby different fields of research are involved investigating general challenges in online planning.

References

1. Christoph Aigner, Jürgen Dorn, and Mario Girsch. Cooperation between a simulation environment and a reactive scheduling system,. In *Proceedings of the Eighth International Conference on Manufacturing Engineering (CME 2000)*, Sydney Australia, 2000.
2. Marlene Arangú, Antonio Garrido, and Eva Onaindia. A general technique for plan repair. In *Proceedings of the 20th IEEE International Conference on Tools with Artificial Intelligence*, volume 1, pages 515–518, Dayton, Ohio,USA, 2008. IEEE Computer Society.
3. Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis.* Cambridge Univ. Press,, 1998.
4. Sergio Cavalieri, Marco Garetti, Marco Macchi, and Marco Taisch. An experimental benchmarking of two multi-agent architectures for production scheduling and control. *Computers in Industry*, 43:139 – 152, 2000.
5. William Cushing and Subbarao Kambhampati. Replanning: a new perspective. In *Poster Program. In Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS-05)*, Montery, 2005.
6. Jürgen Dorn. Evaluating reactive scheduling systems. In *IAT '04: Proceedings of the Intelligent Agent Technology, IEEE/WIC/ACM International Conference on (IAT'04)*, pages 458–461. IEEE Computer Society, 2004.
7. Jürgen Dorn, Roger Kerr, and Gabi Thalhammer. Reactive scheduling: Improving the robustness of schedules and restricting the effects of shop floor disturbances by fuzzy reasoning. *International Journal on Human-Computer Studies*, 42(6):687 – 704, 1995.
8. Klaus Fischer, Jorg P. Muller, Markus Pischel, and Darius Schier. A model for cooperative transportation scheduling. In Victor R. Lesser and Les Gasser, editors, *Proceedings of the First International Conference on Multiagent Systems (ICMAS)*, pages 109–116, San Francisco, California, USA, 1995. The MIT Press.
9. Truls Flatberg, Geir Hasle, Oddvar Kloster, Eivind J. Nilssen, and Atle Riise. Dynamic and stochastic vehicle routing in practice. In Vasileios Zeimpekis, Christos D. Tarantilis, George M. Giaglis, and Ioannis Minis, editors, *Dynamic Fleet Management: Concepts, Systems, Algorithms & Case Studies*, Operations Research / Computer Science Interfaces Series, pages 41 – 63. Springer, 2007.

10. Maria Fox, Alfonso Gerevini, Derek Long, and Ivan Serina. Plan stability: Replanning versus plan repair. In Derek Long, Stephen F. Smith, Daniel Borrajo, and Lee McCluskey, editors, *Proceedings of the Sixteenth International Conference on Automated Planning and Scheduling (ICAPS 2006)*, pages 212–221, Cumbria, UK, 2006. AAAI Press.
11. G.D.M. Frizelle, D. McFarlane, and L. Bongaerts. Disturbance measurement in manufacturing production systems. In *Advanced Summer Institute (ASI 98)*, Bremen, Germany, 1998.
12. Herwig Henseler. *Aktive Ablaufplanung mit Multi-Agenten*, volume 180 of *DISKI*. infix Verlag, 1998.
13. Allan Larsen. *The Dynamic Vehicle Routing Problem.* PhD thesis, Technical University of Denmark (DTU), 2001.
14. David Luckham. *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems.* Addison-Wesley Longman, Amsterdam, 2002.
15. Bernhard Nebel and Jana Koehler. Plan reuses versus plan generation: A theoretical and empirical analysis. *Artificial Intelligence*, 75(1-2):427–454, 1995.
16. Djamila Ouelhadj, Sanja Petrovic, Peter I. Cowling, and Amnon Meisels. Inter-agent cooperation and communication for agent-based robust dynamic scheduling in steel production. *Advanced Engineering Informatics*, 18(3):161–172, 2004.
17. Giselher Pankratz. Dynamic vehicle routing by means of a genetic algorithm. *International Journal of Physical Distribution & Logistics Management*, 35(5):362–383, 2005.
18. Y. Peng, Tim Finin, Yannis Labrou, R. Cost, B. Chu, J. Long, W. Tolone, and A. Boughannam. An agent-based approach for manufacturing integration - the ciimplex experience. *International Journal of Applied Artificial Intelligence*, 13(1-2):39 – 64, 1999.
19. András Pfeiffer, Botond Kádár, and László Monostri. Stability-oriented evaluation of rescheduling strategies, by using simulation. *Computers in Industry*, 58:630–643, 2007.
20. Martha E. Pollack. Planning in Dynamic Environments: The DIPART System. In A. Tate, editor, *Advanced Planning Technology: Technology Achievements of the ARPA/Rome Laboratory Planning Initiative*, pages 218 – 225. AAAI Press, 1996.
21. Warren B. Powell. Dynamic models of transportation operations. In A.G. de Kok and S.C. Graves, editors, *Handbooks in Operations Research and Management Science: Supply Chain Management: Design, Coordination and Operation*, volume 11 of *Handbools in Operations Research and Management Science*. Elsevier Publishing Company, 2003.
22. Amritpal Singh Raheja, K. Rama Bhupal Reddy, and Velusamy Subramaniam. A generic mechanism for repairing job shop schedules. *Innovation in Manufacturing Systems and Technology (IMST)*, 2003.
23. Andreas Richter. *Dynamische Tourenplanung: Modifikation von klassischen Heuristiken für das Dynamische Rundreiseproblem (DTSP) und das Dynamische Tourenplanungsproblem (DVRP) mit der Möglichkeit der Änderung des aktuellen Fahrzeugzuges.* PhD thesis, Technischen Universität Dresden, 2005.
24. I. Sabuncuoglu and M. Bayõz. Analysis of reactive scheduling problems in a job shop environment. *EJORS European Journal of Operational Research*, 126:567 – 586, 2000.
25. Tuomas Sandholm and Victor R. Lesser. On automated contracting in multi-enterprise manufacturing. In *Proceedings of Improving Manufacturing Performance*

in a Distributed Enterprise: Advanced Systems and Tools, pages 33–42, Edinburgh, Scotland, 1995.
26. Jürgen Sauer. Vertical data integration for reactive scheduling. *Künstliche Intelligenz*, 2009, to appear.
27. John Sauter and H. van Dyke Parunak. Ants in the supply chain. In *Workshop on Agent based Decision Support for Managing the Internet-Enabled Supply Chain, Agents 99,*, Seattle, 1999.
28. René Schumann, Thomas Timmermann, and Ingo J. Timm. Transportation planning in dynamic environments. In Bernhard Fleischmann, Karl-Heinz Borgwardt, Robert Klein, and Axel Tuma, editors, *Operations Research Proceedings 2008*, Operations Research Proceedings, pages 319 – 324, Augsburg, 2009. Springer.
29. Stephen F. Smith. Opis: A methodology and architecture for reactive scheduling. In Monte Zweben and Mark S. Fox, editors, *Intelligent Scheduling*, pages 29 – 66. Morgan Kaufmann Publishers, San Francisco, 1994.
30. Paul Valckenaers and Hendrik Van Brussel. Holonic manufacturing execution systems. *CIRP annals International Institution for Production Engineering Research annals*, 2006.
31. Paul Verstraete and Paul Valckenaers. Towards cooperating planning and manufacturing execution systems. In *INCOM*, 2006.
32. Paul Verstraete, Paul Valckenaers, Hendrik Van Brussel, Karuna Hadeli, and Bart Saint Germain. Multi-agent coordination and control testbed for planning and scheduling strategies. In *Fifth international joint conference on Autonomous agents and multiagent systems*, pages 1451 – 1452, Hakodate, Japan, 2006.
33. Guilherme E. Vieira, Jeffrey W. Herrmann, and Edward Lin. Rescheduling manufacturing systems: A framework of strategies, policies, and methods. *Journal of Scheduling*, 6(1):39–62, 2003.
34. Jan Wörner and Heinz Wörn. Benchmarking of multiagent systems in a production planning and control environment. In Stefan Kirn, Otthein Herzog, Peter Lockemann, and Otto Spaniol, editors, *Multiagent Engineering: Theory and Applications in Enterprises*, International Handbook on Information Systems, pages 115 –134. Springer, Berlin et al, 2006.

A Comparison of Metaheuristics on a Practial Staff Scheduling Problem

Maik Günther and Volker Nissen

Technical University of Ilmenau, Chair of Information Systems in Services,
D-98684 Ilmenau, Germany,
<maik.guenther@gmx.de, volker.nissen@tu-ilmenau.de>

Abstract. A practical scenario from logistics is used in this paper to compare different variants of particle swarm optimisation (PSO) and the evolution strategy (ES). Rapid, sub-daily planning with metaheuristics, which is the focus of our research, can significantly add to the improvement of staff scheduling in practice. PSO outperformes ES on this problem. The superior performance must be attributed to the operators and parameters of PSO since the coding of PSO and ES are identical. Repairing solutions to reduce the violation of soft constraints significantly improves the quality of results for both metaheuristics, although the runtime requirements are approximately doubled.

Key words: staff scheduling, sub-daily planning, metaheuristic, particle swarm optimization, evolution strategy

1 Introduction to the Problem of Staff Scheduling

Despite many advantages in automation logistics is still very labour intensive. At the same time logisticians are under considerable pressure. This is not only determined by legal regulations, stronger customer orientation and an increasingly tough international competition. Above all the current economic situation forces logisticians to take measures in order to remain competitive. On the one hand they try to hold the revenue. On the other hand they want to cut costs.

An important parameter for cost cuts is demand-oriented staff scheduling. According to current research employees spend up to 36% of their working time unproductively, depending on the branch [22]. It is working time in which the employees are present and being paid but are not making value-creating contribution. Major reasons include a lack of planning and controlling. The problem can be faced with demand-oriented staff scheduling. Staff scheduling ensures that the right number of employees with the right skills is available at the right time while considering various constraints. Beside the reduction of unproductive working time the key planning goals are increased productivity, reduction of personnel costs, prevention of overtime, motivation of employees with positive results for sales and service [24].

In practice, most often planning takes place based on prior experience or with the aid of spreadsheets. The application of a system for staff scheduling has not been very prevalent up to now [1]. It is obvious that the aforementioned goals of demand-oriented staff scheduling cannot be realised with these planning tools. Even with popular staff planning software employees are regularly scheduled for one workstation per day. However, in many branches, such as trade and logistics, the one-employee-one-station concept does not correspond to the actual requirements and sacrifices potential resources. Therefore, sub-daily planning should be an integral component of demand-oriented staff scheduling. This work compares two metaheuristics for solving the problem of *subdaily* staff scheduling with individual workstations.

In the following section we discuss work related to our own research before developing approaches based on PSO and ES for sub-daily staff scheduling in section 3. Section 4 describes the practical planning scenario and experimental setup. For the mathematical model of the problem reference is made to [19]. Empirical results are presented and discussed in section 5. The paper concludes with a short summary and some indications for future work.

2 Related Work

Ernst et al. offer a summary of about 700 papers related to the issue of staff scheduling in [8]. They identify certain categories of problems, such as the category *flexible demand*. This category is characterised by little available information on schedules and upcoming orders. A demand per time interval is given as well as a required qualification. Thus, the application problem discussed here can be classified in the group flexible demand schemes. It can additionally be classed under task assignment. *Task assignment* is used to generate assignments requiring certain qualifications and needing to be completed in a certain period of time, which are then distributed amongst the employees. The employees have already been assigned shifts.

As work related to our research Vanden Berghe [28] presents an interesting, approach to sub-daily planning. Here, demand is marked by sub-daily time periods, which allows the decoupling of staff demand from fixed shifts resulting in fewer idle times. However, scheduling ist not performed at the detailed level of individual workstations as in our research.

In [16] Schaerf and Meisels provide a universal definition of an employee timetabling problem. Both the concepts of shifts and of tasks are included, whereby a shift may include several tasks. Employees are assigned to the shifts and assume tasks for which they are qualified. Since the task is valid for the duration of a complete shift, no sub-daily changes of tasks (or rather workstations) are made. Blöchlinger [5] introduces timetabling blocks (TTBs) with individual lengths. In this model employees may be assigned to several sequential TTBs, by which subdaily time intervals could be represented within a shift. Blöchlinger's work also considers tasks – however, a task is always fixed to a TTB. Essentially, our problem of the logistics service provider represents a combination of [16] (assignment of staff to tasks) and [5] (sub-daily time intervals), but with the assignment periods (shifts) of the employees already being set.

Staff scheduling is a hard optimization problem. In [10] Garey and Johnson demonstrate that even simple versions of staff scheduling problems are NP-hard. Kragelund and Kabel [14] show the NP-hardness of the general employee timetabling problem. Moreover, Tien and Kamiyama prove in [27] that practical personnel scheduling problems are generally more complex than the TSP which is itself NP-hard. Thus, heuristic approaches appear justified for our application.

Apparently, there exists no off-the-shelf solution approach to the kind of detailed sub-daily staff planning problem considered here. A PSO approach and evolution strategy for this combinatorical optimzation problem are outlined in the following section. This work builds upon previous research [19]. We now give a brief general overview of particle swarm optimization and the evolution strategy. For more details, the reader ist referred to [13] [9] for standard-PSO and [3] [4] for standard-ES.

The basic principles of particle swarm optimization (PSO) were developed by Kennedy and Eberhart among others [12] [13]. Swarm members are assumed to be massless, collision-free particles that search for optima with the aid of a fitness function within a solution space. In this process each single particle together with its position embodies a solution to the problem [29]. While looking for the optimum, a particle does not simply orient itself using its own experience but also using the experience of its neighbours [9]. This means that the particles exchange information, which can then positively influence the development of the population in the social system as a whole [20].

Modifications of standard real-valued PSO exist for binary variables, where the speed of a particle is used as the probability for the change of the binary value [13]. This approach, however, has several limitations and was changed from binary to decimal variables in [30]. Another PSO-variant was developed for sequence planning tasks [26]. In 2007 Poli analysed the IEEE Xplore database for the thematic grouping of PSO applications [21]. Of approximately 1100 publications only one work is focused specifically on timetabling [7] which is related to our own application problem. In [7], the authors adjust PSO to the combinatorial domain. No longer is the position of a particle determined by its speed but rather by using permutation operators. In [6] university timetabling was also approached with PSO.

The evolution strategy (ES) was originally invented by Rechenberg and Schwefel [4] and soon applied to continuous parameter optimization problems. Like genetic algorithms the evolution strategy belongs to the class of evolutionary algorithms that form broadly applicable metaheuristics, based on an abstraction of the processes of natural evolution [2] [3]. There is some work on the evolution strategy in combinatorial and discrete optimization. Herdy [11] investigates discrete problems with some focus on neighbourhood sizes during mutation. Rudolph [23] develops an evolution strategy for integer programming by constructing a mutation distribution that fits this particular search space. Bäck [2] discusses mutation realized by random bit-flips in binary search spaces. Nissen [17] modifies the coding and the mutation operator of the evolution strategy to solve combinatorial quadratic assignment problems. Schindler et al. [25]

apply the evolution strategy to combinatorial tree problems by using a random key representation which represents trees with real numbers. Schwefel and Beyer [4] present permutation operators for the evolution strategy in combinatorial ordering problems. Li et al. [15] develop a mixed-integer variant of the evolution strategy that can optimize different types of decision variables, including continuous, normal discrete, and ordinal discrete values. Nissen and Gold [18] propose an evolution strategy for a combinatorial network design problem that successfully utilises a repair heuristic and domain-specific mutations. However, continuous parameter optimization is still the dominant field of application for the evolution strategy, as the operators of the standard form are particularly well adapted to this type of problem.

3 PSO Approach and Evolution Strategy

3.1 Problem Representation

The problem is represented as a two-dimensional matrix of employees and time periods, where the cells are filled with workstation assignments. With this representation each employee can only work at one workstation in a time period (hard constraint). To mark times in which an employee is not present due to his work-time model, a dummy workstation is introduced (in table 1: workstation 0). Assignment changes can only be made to non-dummy workstations, so that no absent employee is included (hard constraint). Each particle (PSO) and also each individual (ES) has an own matrix that determines its position and represents a solution to the application problem.

To lower the complexity the number of dimensions is reduced via a suitable depiction of time. Within the planned day, time is viewed with a time-discrete model. An event point (at which a new time interval begins) occurs when the allocation requirement for one or more workstations or employee availability change. With this method, however, the periods are not equally long any more, so that their lengths need to be stored.

Table 1. Assignment of workstations in a matrix.

employee	**period**						
	1	2	3	4	5	6	...
1	1	1	1	1	1	1	
2	0	2	2	2	2	2	
3	0	1	1	2	2	2	
...							

3.2 Repair Heuristic

Both scheduling metaheuristics outlined in this paper employ an identical repair heuristic to reduce the total error points of a solution. This repair heuristic does not generate identical particles or individuals. It corrects constraint violations in the following order based on error point size:

- Qualification: employees not qualified for the currently assigned workstation are given an apropriate assignment whilst ignoring under- or overstaffing.
- No demand: employees currently assigned to a workstation with zero demand are given a different assignment (if possible) whilst simultaneously considering their qualification.
- Understaffing: if workstations are understaffed employees are reassigned from other workstations with overstaffing (if possible) also considering their qualification. Thus, simultaneously the problem of overstaffing is reduced.

3.3 PSO for this Application

At the start of PSO the initialisation of the particle position creates valid assignments w.r.t. the hard constraints by using information from the company's current full-day staff schedule. Therefore, valuable foreknowledge is not wasted. Based on this plan, improved solutions can now be determined that include plausible workstation changes.

In each iteration the new particle position is determined by traversing all dimensions and executing one of the following actions with predefined probability. The probability distribution was heuristically determined in prior tests:

- No change: The workstation already assigned remains. (prob. p_1)
- Random workstation: A workstation is randomly determined and assigned. Only those assignments are made for which the employee is qualified. The probability function is uniformly distributed. (prob. p_2)
- pBest workstation: The corresponding workstation is assigned to the particle dimension from pBest, the best position found so far by the particle. Through this, the individual PSO component is taken into account. (prob. p_3)
- gBest workstation: The corresponding workstation is assigned to the particle dimension from gBest (or rather lBest if a gBest neighbourhood topology is not being used). gBest (lBest) represents the best position of all particles globally (in the local neighbourhood). The social behaviour of the swarm is controlled with these types of assignments. (prob. p_4)

By considering the best position of all particles, the experience of the swarm is included in the calculation of the new position. Premature convergence on a sub-optimal position can be avoided by using the lBest topology, in which a particle is only linked to its neighbours. The extent to which the swarm acts individually or socially is determined by the probability with which the workstation is assigned from pBest, gBest or lBest. The behaviour of the PSO-heuristic is relatively insensitive to changes of p_1, p_3, and p_4. The optimal value for p_2 depends on the problem size. Pre-tests revealed that a value of 0.3% for p_2 works best for the problem investigated here. The other probabilities were set at p_1=9.7%, p_3=30%, and p_4=60% with a gBest topology. Once created, a solution is repaired with the heuristic described in the previous section before it undergoes evaluation.

The characteristics of PSO have not been changed with these modifications. There are merely changes in the way to determine a new particle position, so that the calculation of the velocity is not needed. The current form of position determination makes it unnecessary to deal with dimension overruns. All other peculiarities of PSO regarding social or individual behaviour remain. Even all neighbourhood topologies can be used without restrictions. The neighbourhood can be defined by the particles index. In our implementation, PSO terminates after 400,000 inspected solutions. The following pseudocode presents an overview of the implemented PSO.

```
initialise the swarm
evaluate the particles of the swarm
determine pBest for each particle and gBest
loop
for i=1 to number of particles
   calculate new position // use the 4 alternative actions
   repair the particle
   evaluate the particle
   if f(new position)<f(pBest) then pBest=new position
   if f(pBest)<f(gBest) then gBest=pBest
next i
until termination
```

3.4 Evolution Strategy for this Application

The PSO results are compared to several variants of the evolution strategy. The ES population is initialized with valid solutions w.r.t the hard problem constraints. Again, information from the company's current full-day staff schedule is used. (μ, λ)-selection (comma-selection) as well as $(\mu + \lambda)$-selection (plus-selection) are used as well as different population sizes. The best solution found during an experimental run is always stored and updated in a "golden cage". It represents the final solution of the run. Following suggestions in the literature [3] [4], the ratio μ/λ is set to 1/5 - 1/7 during the practical experiments.

Ten alternative recombination variants were evaluated in a pre-test. The best performance was achieved with a rather simple form that is based on the classical one-point crossover. The recombination of parents to create an offspring solution works as follows: A common crossover point is determined at random for all employees (rows) of a solution and the associated parts of the parents are exchanged (see fig. 1).

Mutation is the main search operator employed in the evolution strategy. In standard-ES mutation is performed using normally-distributed random variables so that small changes in a solution are more frequent than large changes. In [19] we developed a search operator that adheres quite closely to this classical form of mutation and produced fairly good results.

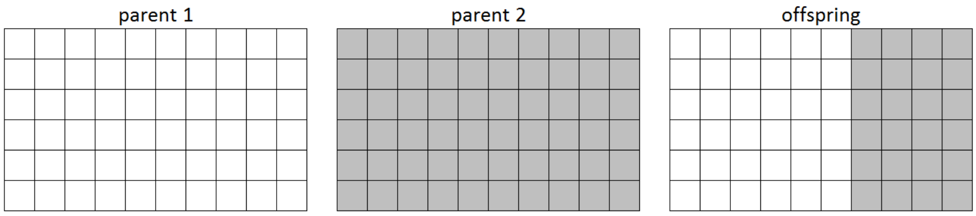


Fig. 1. Recombination operator employed.

In this paper, a different approach to mutation is employed that takes the characteristics of the discrete search space better into account. It is based on the work of Rudolph [23]. He developed an approach to construct a mutation distribution for unbounded integer search spaces. The concept of maximum entropy is used to select a specific distribution from numerous potential candidates. Rudolph tested his ideas on five nonlinear integer problems. Some adaptations were required for the staff scheduling problem, though. The search space in our problem domain is bounded and hard constraints must be considered. In short, the main differences to Rudolph's approach are as follows:

– dimension boundaries for the bounded search space are introduced,
– mutation produces only changes that consider employee availability,
– mutation respects necessary qualifications,
– mutation intensity is increased to account for the complex search space.

Before a solution is evaluated it is repaired using the same repair heuristic as was the case for PSO. The ES terminates when 400,000 solutions have been inspected to allow for a fair comparison with PSO. The following pseudocode presents an overview of the implemented ES.

01: initialise the population with μ individuals
02: repair the μ individuals
03: evaluate the μ individuals
04: loop
05: copy and recombine parents to generate λ offspring
06: mutate the λ offspring
07: repair the λ offspring
08: evaluate the λ offspring
09: select $((\mu + \lambda)$ or $(\mu, \lambda))$ best individuals as new generation
10: until termination

4 Test Problem and Experimental Setup

The present problem originates from a German logistician, which operates in a spatially limited area 7 days a week almost 24 hours a day. The tasks of employees concern logistic services e.g. loading and unloading or short distance transportation. The employees are quite flexible in terms of their working hours, which results in 13 different working-time models. There are strict regulations especially with regard to qualifications because the assignment of unqualified employees might lead to significant material damage and personnel injury. The employer regularly invests a lot of time and money in qualification measures so that many different employees can work at several different workstations. Currently, monthly staff scheduling is carried out manually within MS EXCEL$^{\text{TM}}$. Employees are assigned a working-time model and a fixed workstation each day. Several considerations are included, such as presence and absence, timesheet balances, qualifications and resting times etc.

The personnel demand for each workstation is subject to large variations during the day. However, employees are generally scheduled to work at the same workstation all day, causing large phases of over- and understaffing. This lowers the quality of service and the motivation of employees and leads to unnecessary personnel costs as well as downtime. Because of the complexity sub-daily workstation rotation is only rarely used in the planning. Usually, department managers intervene directly on site and reassign the employees manually. Obviously, demand-oriented staff scheduling cannot be realised with this approach.

The planning problem covers seven days (20 hours each), divided into 15-minute intervals. It includes 65 employees and, thus, an uncompressed total of 36,400 dimensions for the optimization problem to be solved. The general availability of the employees is known for each interval from the previous full-day planning. Nine different workstations need to be filled, with seven having qualification requirements. For the mathematical model of the problem reference is made to [19].

5 Results and Discussion

The results of the various scheduling approaches are shown in table 2. All test runs were conducted on a PC with an Intel 4 x 2.67 GHz processor and 4 GB of RAM. Thirty independent runs were conducted each time for each of the experiments to allow for statistical testing. The full-day manual staff schedule without sub-daily workstation changes results in 411,330 error points after an evaluation that included the penalties arising from the afore-mentioned constraints. All heuristics for sub-daily staff scheduling significantly outperform the manual full-day schedule in terms of total error points. This demonstrates the value of sub-daily scheduling as compared to today's standard staff scheduling approach which not only wastes resources but also demotivates personnel and deteriorates quality of service. Generally, the problems of understaffing and overstaffing for periods without demand are greatly reduced. On the other hand, all heuristics lead to more overstaffing in periods with demand > 0 as compared to the initial plan. This approach, however, is sensible because employees can still support each other instead of being idle when demand $= 0$.

Interestingly, the PSO heuristic provides the best results with a rather small swarm size of 10 particles, but also larger swarm sizes produce good results when

Table 2. Comparison (error points) of the different sub-daily scheduling heuristics with a termination criteria of 400,000 inspected solutions for PSO and ES, based on 30 independent runs each. Best results are bold and underlined. The suffix +R indicates the use of the repair heuristic. Results for PSO without repair are based on [19].

heuristic	error mean	error min	error standard deviation	number of job-changes	wrong qualifications in minutes	under-staffing in minutes	overstaffing in minutes demand >0	overstaffing in minutes demand =0
manual plan	411330	411330	-	0.0	1545	20130.0	14610.0	33795.0
PSO (20)	52162	51967	92.2	1666.8	0	7478.5	28488.0	7265.5
PSO (100)	52591	52400	107.7	1778.5	19.1	8136.8	27874.1	8537.7
PSO (200)	53727	53467	138.9	2220.3	0	7658.5	28017.0	7916.5
PSO (10)+R	<u>**51752**</u>	<u>**51736**</u>	9.2	1502.2	0	7365.0	28395.0	7245.0
PSO (20)+R	51781	51763	9.3	1531.4	0	7365.0	28395.0	7245.0
PSO (100)+R	51826	51811	9.0	1575.8	0	7365.0	28395.0	7245.0
PSO (200)+R	51841	51817	13.1	1591.2	0	7365.0	28395.0	7245.0
ES (10,50)	53648	53048	338.2	2149.4	0	7674.5	28075.2	7873.5
ES (10+50)	52864	52493	343.9	1919.2	0	7554.5	27422.3	7560.5
ES (30,200)	54471	53954	233.3	2451.1	0	7725.5	27706.5	8294.0
ES (30+200)	53824	53428	167.0	2351.1	0	7678.0	28111.0	7842.0
ES (10,50)+R	51870	51842	18.1	1620.3	0	7365.0	28395.0	7245.0
ES (10+50)+R	51843	51816	16.4	1592.5	0	7365.0	28395.0	7245.0
ES (30,200)+R	51855	51835	13.2	1604.5	0	7365.0	28395.0	7245.0
ES (30+200)+R	51846	51820	11.7	1596.3	0	7365.0	28395.0	7245.0

solution repair (+R in table 2) is employed. ES reacts similarly, with 10 parents performing better than 30 parents. Because of the uniform termination criterion of 400,000 fitness function calculations, a smaller population or swarm size means more iteration cycles. Many steps are required to arrive at a good schedule. Thus, it seems preferable to track changes for more iterations as compared to a higher diversity through larger population or swarm size.

Apparently, the plus-selection has a slight advantage over the comma-selection for the ES on this problem instance, but this should not be generalized. The mutation scheme based on maximum entropy provides better results for the ES than a more traditional approach based on rounded Gaussian mutations as given in [19]. This result underlines the great importance of adapting the mutation operator to fit the characteristics of the search space as good as possible.

Repairing solutions significantly improves the quality of results, but also doubles the runtime requirements of the heuristics from roughly 25 minutes in [19] to 50 minutes here for a single run of PSO and ES with 400,000 inspected solutions. This effort, however, is acceptable as there is sufficient time available for creating the schedule.

PSO(10) and ES(10+50), both with solution repair, provided the best mean error results in their respective groups. With 30 independent runs for each heuristic it is possible to test the statistical significance of the performance difference between both solution methods with a t-test (see table 3). A Levene-test revealed the homogeniety of variances (test level 5%) between both groups ($F = 3.55, p = 0.065$). The corresponding t-test with a 95% confidence interval confirms the better performance of PSO(10) with a very high statistical significance ($p < 0.001$ for H_0). The result remains the same, if heterogeniety of variances is assumed. A further test was conducted to compare the best parameterisation of PSO and ES without solution repair. Here PSO(20) and ES(10+50) were compared. The Levene-test confirmed the heterogeniety of variances (test level 5%) between both groups ($F = 13.74, p < 0.001$). Again, PSO outperforms the ES on a highly significant level ($p < 0.001$ for H_0). This success of PSO must be attributed to its operators since the coding of PSO and ES are identical.

Table 3. t-test results for pairwise comparison of heuristics.

H_1	T	df	significance H_0 (l-tailed)	mean difference	95% confidence intervall of differences	
					lower	upper
PSO(10)+R < ES(10+50)+R	-26.40	58.00	< 0.001	-90.67	-97.54	-83.79
PSO(20) < ES(10+50)	-10.79	33.15	< 0.001	-701.63	-833.88	-569.39

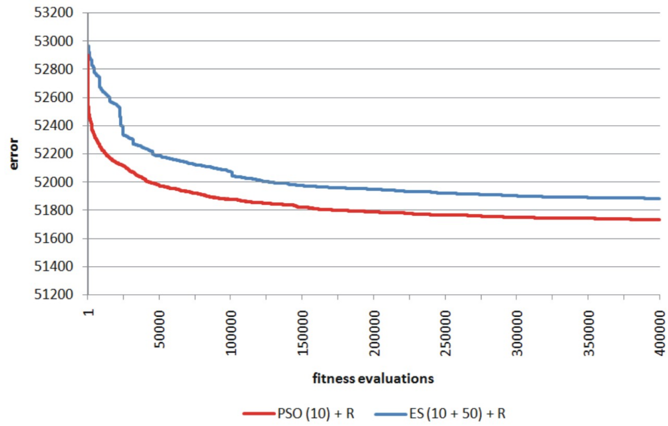


Fig. 2. Convergence chart for PSO(10) and ES(10+50), both with solution repair.

Fig. 2 shows the convergence behaviour of best variants from PSO and ES in comparison. Not only does PSO generate the better final solution, but it also demonstrates a more rapid convergence towards good solutions. This is generally a desirable characteristic, particularly when the available time for an optimization is rather limited (not the case here).

Extensive pre-tests were done for parameter setting of PSO and ES. Notwithstanding the fact that the PSO heuristic was able to provide better results for this problem, it also has one technical advantage over ES. The PSO outlined in this paper only requires the varying of two parameters (swarm size and p_2), which can both be very easily set. ES, on the other hand, offers more parameterisation possibilities (selection pressure, recombination scheme, plus or comma selection etc.), resulting in greater heuristic complexity from a user's perspective.

6 Conclusion and Future Work

Using an actual planning scenario, it was demonstrated that PSO and ES produce far better results than traditional full day scheduling. Sub-daily scheduling significantly increases the value contributions of individual staff members. Because PSO in its traditional form is not suitable for the planning problem at hand, the method was adapted to the combinatorial domain without sacrificing the basic PSO mechanism. ES also uses a mutation scheme based on maximum entropy that better fits the combinatorial domain than classical Gaussian mutations. PSO outperforms different variants of the ES on this problem. The superior performance must be attributed to the operators and parameters of PSO since the coding of PSO and ES are identical. Repairing solutions to reduce the violation of soft constraints significantly improved the quality of results for both metaheuristics, although the runtime requirements were approximately doubled. In future research, other metaheuristics will be tested against the current solution methods to further validate the promising results. Moreover, the approaches devoloped here are currently adapted to other practical domains.

References

1. ATOSS Software AG, FH Heidelberg (eds.) Standort Deutschland 2006. Zukunftssicherung durch intelligentes Personalmanagement. München, 2006.
2. T. Bäck. Evolutionary Algorithms in Theory and Practice. Oxford University Press, 1996.
3. T. Bäck, D.B. Fogel, Z. Michalewicz (eds.). Handbook of Evolutionary Computation. Institute of Physics Publishing, 1997.
4. H.G. Beyer, H.P. Schwefel. Evolution strategies: a comprehensive introduction. In: Natural Computing 1 (2002): 3-52.
5. I. Blöchlinger. Modeling Staff Scheduling Problems. A Tutorial. European Journal of Operational Research 158 (2004): 533-542.
6. O. Brodersen. Eignung schwarmintelligenter Verfahren für die betriebliche Entscheidungsunterstützung. Cuvillier, 2008.
7. S.C. Chu, Y.T. Chen, J.H. Ho. Timetable Scheduling Using Particle Swarm Optimization. In: Proceedings of the International Conference on Innovative Computing. Information and Control (ICICIC Beijing 2006) Vol. 3 (2006): 324-327.
8. A.T. Ernst, H. Jiang, M. Krishnamoorthy, B. Owens, D. Sier. An Annotated Bibliography of Personnel Scheduling and Rostering. In: Annals of OR 127 (2002): 21-144.

9. Y. Fukuyama. Fundamentals of Particle Swarm Optimization Techniques. In: K.Y. Lee, M.A. El-Sharkawi (eds.). Modern Heuristic Optimization Techniques with Applications to Power Systems. Wiley-IEEE Press, 2003, 24-51.
10. M.R. Garey, D.S. Johnson. Computers and Intractability. A Guide to the Theory of NP-Completeness, Freeman, 1979.
11. M. Herdy. Application of the 'Evolutionsstrategie' to Discrete Optimization Problems. In: H.P. Schwefel, R. Männer (eds). Parallel Problem Solving from Nature. Springer, 1990, 188-192.
12. J. Kennedy, R.C. Eberhart. Particle Swarm Optimization. In: Proc. of the IEEE Int. Conf. on Neural Networks. IEEE, 1995, 1942-1948.
13. J. Kennedy, R.C. Eberhart, Y. Shi. Swarm Intelligence. Kaufmann, 2001.
14. L. Kragelund, T. Kabel. Employee Timetabling. An Empirical Study, Master's Thesis, Department of Computer Science, University of Aarhus, Denmark, 1998.
15. R. Li, M.T.M. Emmerich, E.G.P. Bovenkamp, J. Eggermont, T. Bäck, J. Dijkstra, J.H.C. Reiber. Mixed Integer Evolution Strategies and Their Application to Intravascular Ultrasound Image Analysis. In: F. Rothlauf (ed.). Applications of Evolutionary Computation (LNCS 3907). Springer, 2006, 415-426.
16. A. Meisels, A. Schaerf. Modelling and Solving Employee Timetabling. In: Annals of Mathematics and Artificial Intelligence 39 (2003): 41-59.
17. V. Nissen. Solving the Quadratic Assignment Problem with Clues from Nature. IEEE Transactions on Neural Networks 5 (1994) 1: 66-72.
18. V. Nissen, S. Gold. Survivable Network Design with an Evolution Strategy. In: A. Yang, Y. Shan, L.T. Bui (eds.). Success in Evolutionary Computation. Studies in Computational Intelligence, Springer, 2008, 263-283.
19. V. Nissen, M. Günther. Staff Scheduling with Particle Swarm Optimization and Evolution Strategies. In: C. Cotta, P. Cowling (eds.). EvoCOP (LNCS 5482). Springer, 2009, 228-239.
20. K.E. Parsopoulos, M.N. Vrahatis. Recent Approaches to Global Optimization Problems through Particle Swarm Optimization. Nat. Comp. 1 (2002): 235-306.
21. R. Poli. An Analysis of Publications on Particle Swarm Optimization. Report CSM-469, Dep. of Computer Science, University of Essex, England, 2007.
22. Proudfoot Consulting: Produktivitätsbericht 2007. Company Report, 2007.
23. G. Rudolph. An Evolutionary Algorithm for Integer Programming. In: Y. Davidor, H.P. Schwefel, R. Männer (eds.). PPSN III (LNCS 866). Springer, 1994, 139-148.
24. B. Scherf. Wirtschaftliche Nutzenaspekte der Personaleinsatzplanung. In: M. Fank, B. Scherf, (eds.). Handbuch Personaleinsatzplanung. Datakontext, 2005, 55-83.
25. B. Schindler, F. Rothlauf, E. Pesch, E. Evolution strategies, Network Random Keys, and the One-Max Tree Problem. In: Applications of Evolutionary Computing: EvoWorkshops 2002 (LNCS 2279). Springer, 2002, 29-40.
26. M.F. Tasgetiren, M. Sevkli, Y.C. Liang, G. Gencyilmaz, G. Particle Swarm Optimization Algorithm for Single Machine total Weighted Tardiness Problem. In: Proceedings of the CEC 2004. IEEE, 2004, 1412-1419.
27. J. Tien, A. Kamiyama. On Manpower Scheduling Algorithms. In: SIAM Rev. 24 (1982) 3: 275-287.
28. G. Vanden Berghe. An Advanced Model and Novel Meta-heuristic Solution Methods to Personnel Scheduling in Healthcare. Thesis, University of Gent, 2002.
29. K. Veeramachaneni. Optimization Using Particle Swarm with Near Neighbor Interactions. In: GECCO-2003 (LNCS 2723). Springer, 2003, 110-121.
30. K. Veeramachaneni, L. Osadciw, G. Kamath, G. Probabilistically Driven Particle Swarms for Optimization of Multi-valued Discrete Problems: Design and Analysis. In: Proceedings of the IEEE SIS 2007, Honolulu, 2007, 141-149.

Integration of a Hierarchical Decision Making Framework into the Multi-Agent Simulation System LAMPSys

Christoph Mies and Robert Mertens

Fraunhofer Institute Intelligent Analysis and Information Systems IAIS
Schloss Birlinghoven, D-53754 Sankt Augustin, Germany
{christoph.mies,robert.mertens}@iais.fraunhofer.de

Abstract. LAMPSys is a Petri net based platform for simulating multi-agent scenarios. Throughout the last years, it has been used to successfully simulate complex scenarios in a number of different domains. However, in earlier versions of LAMPSys, agent behaviors had to be hard-coded. This fact makes modeling scenarios that involve complex agent behavior a rather tedious endeavor. This paper introduces an approach that integrates the concept of leader agents with LAMPSys' modeling language. Leader agents encapsulate decision making for a group of related agents in one central agent - thus modeling a decision structure that is often found in hierarchical organizations such as most business companies or the military. This strategy does reduce modeling effort by decomposing the overall task into less complex subtasks. It also models many decision structures in a highly realistic manner.

1 Introduction

In this paper we extend the existing agent-based simulation system LAMPSys with a framework for hierarchical decision making. LAMPSys is designed to simulate scenarios for decision support. The system itself is highly customizable and able to simulate scenarios from arbitrary domains. In order to be applied to different application areas, domain-dependent so-called functional agents have to be modeled. These agents are building blocks for the scenarios. In a traffic scenario functional agents would be motor-bikes, cars, pedestrians etc. as well as infrastructure elements such as road signs or traffic lights. Thus, a society of agents defines a given scenario.

The primary goal of LAMPSys is to work as decision support system. This goal is reached by simulating a number of candidate decisions in different scenarios. During these simulations, a lot of additional decisions have to be made autonomously. The simulation results help the user to select the decisions that on average performed best during the simulations. The required enormous number of simulation runs is usually performed on a cluster of computers. However, autonomous decision making is vital for a significant number of simulations.

The introduced framework is based on the concept of leader agents. These agents make decisions for a group of assigned subordinate agents. We follow the general definitions of agents provided in [1, chapter 2] and [2, chapter 1]. A negotiation with agents on the same hierarchical level is not yet integrated[1]. Since most business companies and the military are structured hierarchically, this purely hierarchical approach does not only reduce the modeling effort but also models the decision structures of many organizations in a realistic way.

The remainder of this paper is organized as follows: section 2 introduces the agent-based simulation system LAMPSys as well as its graph-based modeling language LAMPS. The next section gives an insight on how scenarios can be modeled with LAMPSys. Section 4 introduces the concept of leader agents and their command hierarchy. This structure naturally enables a decomposition of overall tasks into less complex tasks. At each level, the number of available actions for the leader agent decreases and a more efficient decision making becomes possible. Then the intended modeling according to the sense-think-act approach is discussed. Section 5 concludes the paper with a final discussion and gives an outlook on future work.

2 LAMPSys

In LAMPSys, processes and agents are modeled with a special modeling language called LAMPS (Language for the Agent-based Modeling of Processes and Scenarios). It extends the concept of colored and hierarchical Petri nets [4, 5]. The former introduced the concept of typed tokens, the latter structured so-called places (states) and actions hierarchically. Additionally, LAMPS introduces the concept of agents. The behavior of agents is modeled with LAMPS fragments. Each is inherently parallel due to its core of high-level Petri nets. Agents live in parallel, and each agent can execute several parallel actions. All actions whose conditions are true in a given cycle are executed. This is also the main difference to flow-charts, because there can be several token per place, and several places can be filled simultaneously. All basic constructs can be recursively encapsulated. For example, places can be combined to so-called *super-places*. An action can be recursively defined as a LAMPS fragment, as long as the interface (i. e. the incoming and outgoing places) of the action and the fragment are identical. This way, a process can be modeled and viewed on different levels of detail. Agents can also be recursively encapsulated. A group of agents can be aggregated into one agent (say, a group of computers into a cluster). LAMPSys is a simulation system that is able to execute LAMPS graphs. It is based on the Flip-Tick-Architecture [6] and uses the concept of a society of agents. The whole environment is modeled in terms of agents and the whole interaction is based on effects of actions executed by agents. The principle of autonomy of agents forbids the direct manipulation of internal data structures and behaviors of other agents. Consequently, all interactions between agents are handled via messages. The basic unit of execution is called a cycle. During one cycle, the agent reads its messages and triggers the appropriate actions, which might

[1] According to [2, chapter 2] and [3], negotiation is not required for multi-agent environments.

consist of writing messages to other agent. This is called the *tick* of an agent. During each tick a certain amount of time dt is simulated. Actions that last longer than dt are triggered during several consecutive ticks and perform only a part of the overall action during each tick. Thus, the time resolution of the simulation can vary from cycle to cycle. This is particularly valuable for increasing the time resolution in the computation of dynamic equations for fast moving objects. The whole simulation is performed in cycles. Each agent manipulates data during its *ticking*. After all agents have been ticked, the system *flips* them. During this *flipping*, the agents publish their data. Thus, at cycle t the agents read data produced in cycle $t - 1$. This process allows the parallel execution of agent actions without the danger of racing conditions. Please refer to [7, 8] for more details about LAMPSys.

3 Modeling Scenarios With LAMPSys

In LAMPSys everything that can change in a scenario is modeled in terms of agents, i. e. a society of agents defines a scenario. Since LAMPSys provides the framework only, all domain specific, i. e. functional, agents have to be modeled before interesting tasks can be simulated. Currently, there exist several domains where LAMPSys is used for simulation [9, 10]. In this paper, the domain of medical evacuation (see [7]) is analyzed in terms of demands of the simulation system. The domain as well as some requirements for the agents involved are explained in section 3.1. A taxonomy of different types of agents is given in section 3.2. The section introduces basic, auxiliary and decision maker agents. Afterward the approach for functional, i. e. domain dependent, communication, which maintains the interoperability of LAMPSys, is discussed. Finally the sense-think-act modeling approach of basic agents is introduced.

3.1 Domain of Medical Evacuation

The domain of medical evacuation is a logistics environment. At the beginning of the simulation some agents representing injured people are placed at random locations. A dispatcher agent can command some emergency transporters. Its task is to let these transporters rescue the injured by transporting them into hospitals. The transporters can either be helicopters or vans. All of them can have different average speeds and transport capacities. The most important difference between helicopters and vans is that the latter move on roads whereas the former ones can fly the direct way. These two types of transporters additionally have different activation and repairing times. The injured have different degrees of injuries, ranging from light to severely wounded. Additionally, they have a certain amount of health points that are reduced over time until the agent perishes. The number of health points as well as the reduction rate depends on the degree of the injury. When the injured have been delivered to the hospital they can be healed, i. e. all health points are restored. The dispatcher's primary target is the survival of all injured. The involved agents have to be capable of moving along roads (e. g. the vans), moving directly to target locations (e. g. the helicopters), communicating and causing effects (e. g. the hospitals must be able to heal the wounded). Additionally, they must be able to make intelligent decisions (dispatcher). The dispatcher agent can be considered as a leader agent (see section 4) since it commands other agents in order to reach its goal. In its basic version the domain of medical evacuation is fully-observable, deterministic and nearly markovian. The only violation of the markov assumption [1, chapter 15] is the stochastic occurrence of wounded. This can easily be solved by generating new decisions when new wounded occur. Additionally, no neutral or opponent agents exist. The wounded are considered as allied agents. Nevertheless, this domain can be extended such that it becomes partial-observable (e. g. if the wounded have to be actively searched for) or non-deterministic (e. g. if inter-agent communication is disrupted).

3.2 Types of Agents

A taxonomy of different types of agents is useful in order to give names to different kinds of agents. The following agent types exist:

- *Basic agents*: These are the most important agents from the functional point of view, because most parts of the domain are modeled by basic agents. They are able to execute orders (see section 3.3) and can perform non-trivial tasks like moving on roads or transporting wounded. Examples are helicopters, vans, hospitals etc.
- *Auxiliary agents*: They are also needed to model a domain. The difference to basic agents is that they perform very simple tasks that merely require decision making and cannot execute orders. Thus, they are relatively simple to model. In the medical evacuation domain, radio-equipment agents or channel-agents (see section 3.4) are auxiliary agents.
- *Decision maker agents*: These agents make complex decisions and their reasoning-ability is very important for the significance of the whole simulation. An example is the dispatcher agent which commands other agents. This agent type is extended to a framework in section 4. Thus, decision maker agents are substituted by leader agents.

This informal taxonomy of agent types simplifies the discussion in the remainder of this paper. Note that LAMPSys is not aware of that taxonomy and treats all agents in the same way.

3.3 Functional Communication

In order to decouple functional (domain-specific messages) and technical (data) communication, all functional communication is encoded in a domain-specific functional language. For the medical evacuation domain, BML [11] is used. BML is based on a formal grammar. It is designed such that it can easily be parsed by computers as well as been understood by humans. BML comprises three

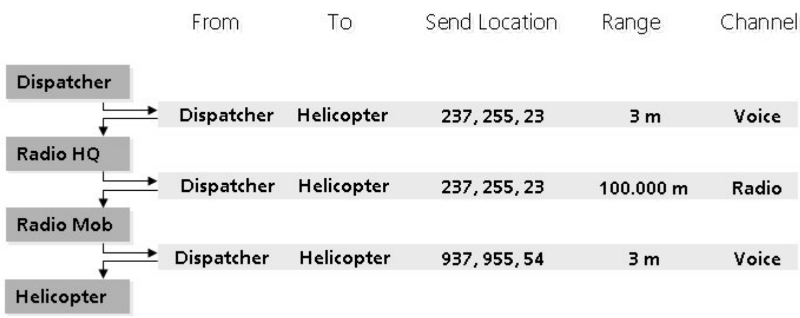


Fig. 1. Exemplary communication chain: The dispatcher wants to send a message to the helicopter which cannot hear it directly due to the small range of the voice-channel. But the dispatcher's radio equipment senses the message and sends it to the radio-channel. The helicopter's mobile radio equipment receives the message from there and sends it back to the voice-channel. Finally, the helicopter agent can receive it from there.

types of statements. It is able to express orders for other agents and reports about interesting events. Additionally it is also able to express effects. The BML statements are then compiled to conceptual graphs [12]. The main advantage is the interoperability of conceptual graphs. Many languages can be compiled to conceptual graphs and vice versa. Thus, the conceptual graph serves as main knowledge representation for many different domains that can all be simulated with LAMPSys. The signatures of services or procedures do not need to be changed. Additionally, it is straight-forward to perform reasoning on conceptual graphs [12]. The following section shows how simple reports being in an agent's short-term-memory can be extended by other reports in order to enrich the contained knowledge.

3.4 Sense, Think, Act for Basic Agents

The basic agents are modeled according to the sense-think-act approach. This sequential model requires that each agent reads and evaluates its sensors at the beginning of each cycle. Afterward, each agent makes its decisions and finally uses its effectors to execute actions in the environment. Many agent architectures are proposed in the literature, e. g. [1, ch. 2.4] and [2, ch. 1.3]. The basic agents can sense sounds, optical input and effects that are caused by other agents. The sound sensor is modeled very detailed in a physical manner. An auxiliary agent represents the voice-channel, from which the agent can sense the words that are spoken in its direct proximity. The radio equipments are also represented by auxiliary agents, that scan the voice sensor and transmit everything they hear to the radio-channel that is used for radio communication. This channel is again scanned by all radio equipments and then transmitted to the voice-channel again. Thus, the radio equipments work as translators between the voice- and the radio-channel. An example of a communication chain is depicted in figure 1. Arbitrary other communication channels, e. g. IT systems, can also be modeled using the same concept. Due to the autonomy of agents, every agent decides for itself how it reacts to a message. The same principle is pursued by the sensing of effects. All caused effects are communicated via a certain auxiliary agent, the effect-channel agent. Every basic agent scans this channel and decides whether it wants to take the effect or not. This is necessary since no agent is allowed to change the state of other agents. Thus, if one agent heals a wounded, the wounded decides if it becomes healthy or not. Taking this approach, causing and taking effects is modeled as communication. Arbitrary chains of effects can be built similarly to the case of the communication chain. If, for example, a helicopter's engine breaks down one effect could be that the helicopter crashes which again could cause the pilot to die. Thus, the primary effect of a broken engine triggers the secondary effect, i. e. the crashing helicopter. This event causes the pilot to die, the third effect. A slightly different approach is taken for the optical sensors, the visibility. Due to performance reasons, the visibility is calculated at a central agent. The technical visibility agent holds a voxel representation of the whole scenario. All basic agents participate in a publisher-subscriber communication. They publish all data that is needed for the visibility calculation, especially their voxel-shape as well as their position, and subscribe for the visibility messages calculated by the visibility agent. Basically, every agent receives a location-message for each other agent it can currently see. The current decision making performance of basic agents is quite poor. After reading the sensory input and fusing the new knowledge with the current one. This knowledge is composed of orders and reports. The second step concerns the execution of BML statements that are included in the agents' world model. Note that all basic agents are able to execute BML orders (see section 3.2). As mentioned above, conceptual graphs are used to represent the basic knowledge. An example for knowledge is the BML order `move on road dispatcher van to (100.0 200.0 0.0) reply with id1`. Its meaning is that the dispatcher-agent `dispatcher` orders the medical transporter `van` to move to the coordinates `(100.0, 200.0, 0.0)` and to send a reply-message with the label `id1` when the order's execution is finished. Let the answer of the agent `van` be `reply van dispatcher id1 successful` meaning that agent `van` has fulfilled its order. Then the agent `dispatcher` would merge the answer with its own order resulting in a larger conceptual graph. With this simple merge concept, arbitrary graphs with a huge amount of contextual information can be built. Thus, each graph represents information that belongs to the same context. All important knowledge concerning this context can be extracted from one single graph. In order to model the restricted mental capacity of basic agents only a certain number of graphs can be contained in the STM at the same time. If the maximal number of graphs in the STM is exceeded the oldest ones are discarded, i. e. forgotten. Additionally, all BML statements that have not been updated for more than a certain time are removed. Note that each merge with other graphs causes the graph to get a new time stamp and count as new graph. One advantage of the merged conceptual graphs can be seen here. After receiving `van`'s reply message, `dispatcher` automatically refreshes the knowledge about the given order. Thus, the whole context of the reply message is refreshed. The second phase of the current decision making of basic agents is the execution of BML orders. This execution can be compared with reactive skills of a layered robot control architecture as defined in [13, chapter 8]. Currently, any agent executes any order that it receives. If several contradicting orders occur, the last received is executed. Note that by now there exists no model for the identification of contradicting BML orders. Executing a BML order works as follows. A certain LAMPS graph `execute_bml_order` reads the orders. According to the type of order, e. g. `move on road`, the graph activates other LAMPS graphs by changing the corresponding attributes of the agent. This newly activated LAMPS graph, e. g. `move_on_road`, performs the action itself. Additionally `execute_bml_order` monitors the execution of the activated graph, e. g. `move_on_road`. If this LAMPS graphs stops, `execute_bml_order` generates a reply message, e. g. `reply van dispatcher id1 successful`. The activated LAMPS graph is modeled by hand, i. e. it is hard-coded. This is tractable for "simple" graphs like moving on a road, but gets very tedious and complicated if complex graphs like the dispatching of transporters are to be modeled. Afterward, the act phase begins. Here, the desired effects are caused. Basically, there are two kinds of effects of actions. On the one hand, effects may only cause the change of internal attributes. For example, if an agent is moving, it basically changes the internal attribute position. Note that a position change is published to the visibility agent as mentioned above. But in principle, a change of an internal attribute is not observable for other agents. On the other hand, an effect may involve communication. This approach is taken by speaking some orders for other agents into the voice-channel agent or by causing effects (e. g. healing) via the effect-channel agent.

4 Leader Agents

A *leader agent* commands some subordinate agents in order to reach its goal. The logistic dispatchers can be considered as leader agents although they do not yet work as described in this section. Basically, a leader agent calculates a plan for all its subordinates and monitors the plan's execution. Since a leader agent always simulates the "intelligence" of an agent, it must be plugged into a basic agent (see section 3.2). Metaphorically speaking, the logical leader agent, which models the intelligence, must be hosted in a basic agent that models the physics. Section 4.1 introduces the command hierarchy. This hierarchy defines the relation between leader agents and subordinate agents. The following three sections describe the sense-think-act modeling approach for leader agents. Especially section 4.3 is important since decision making is the core process of a leader agent.

4.1 Command Hierarchy

A precondition for leader agents is a command hierarchy which models which agent is commanded by which leader. Note that a recursive assignment is possible, i. e. a leader agent can command subordinate leader agents. This chain

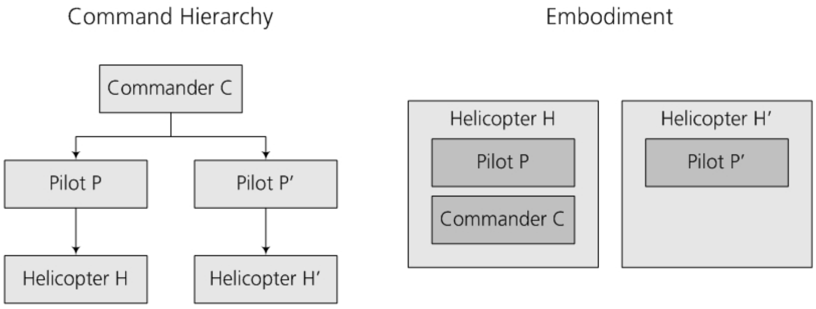


Fig. 2. The left part shows the command hierarchy: The leader agent C commands the elementary leader agents P and P' which again command the basic agents H and H', respectively. The embodiment of the leader agents is shown on the right: H embodies P as well as C and H' embodies P'.

ends with the *elementary leader agents* that command exactly one basic agent. The hierarchy must be modeled such that every agent is assigned to at most one leader agent in order to avoid competing orders. Note that it is not acceptable to simply execute the last given orders in case of conflicts if they come from different leader agents. Consider a leader agent C that commands two helicopters H and H'. Each of these helicopters consists of the helicopter itself (represented as basic agent) as well as the corresponding elementary leader agents P and P', respectively. Then each helicopter agent is commanded by its corresponding elementary leader agent which is itself commanded by C. Thus, C cannot command the basic helicopter agents directly. Note that C has to be embodied in a physical basic agent. It can be embodied in one of the helicopter agents denoted by H. In this case H embodies two leader agents (one commanding the other one) but is directly commanded by one, only. Figure 2 shows this exemplary situation. The task of the elementary leader agent is to command its embodying basic agent. The main advantage of sourcing out the basic agent's decision making to the elementary leader is a more modular modeling. The intelligence is encapsulated in the leader whereas the capabilities of, for example, a helicopter are modeled in the basic agent. Thus, the intelligence can be changed by simply changing the leader. It is also important that the command hierarchy is flexible during the simulation. Thus, it is possible that one agent moves from the command of one leader to the command of another one. This is especially important, because a basic agent can vanish during the simulation, e. g. a helicopter can crash and the leader agent C from the above example can die. Then a substitution must command the remaining helicopters. Functionally speaking, the commander changes, but technically speaking the command hierarchy changes and all assigned agents must be assigned to another leader agent. By this hierarchical approach, the overall complex task can easily be decomposed into partial tasks. These are then less complex and can again be decomposed into smaller ones. Finally, the elementary leader agent gives orders that are simple enough to be executable by its basic agent directly. Concerning the medical evacuation domain, the overall task is to evacuate all wounded. If n hospitals exists and each of them has its own vans and helicopters, the overall leader agent can command n leader agents that again command the elementary leader agents of the transporters. The embodiment of leader agents simplifies the modeling complexity since these agents serve as a "intelligence plug-in" for basic agents. The latter ones carry out tasks that require less tactical requirements whereas the leader agents' behavior has to be tuned in order to get suitable results. Thus, the basic agents need not to be adapted for every scenario.

4.2 Sense: World Model

The world model is the basis for all decisions that a leader agent can make. In each cycle, the world model is updated by sensory input. Currently, the sensory input consists of BML statements for facts. Additionally, the ¨felt¨ world can be represented by potential fields. Basically, friendly and hostile areas can be assigned peaks and troughs, respectively, in a potential field. This knowledge can, for example, be used for routing along roads. Another important point is the representation of the leader agents' target. This basically consists of the formulation of one primary target that has to be reached under certain constraints (secondary targets). In the domain of medical evacuation the target could be to rescue all wounded under the constraints of using as less fuel as possible, or of being as fast as possible. Planning requires models of the environment. In LAMPSys all non-static entities are represented as agents. Thus, only models about other agents are needed. From the leader agent's point of view the assigned agents can execute certain BML statements. Thus, they can be assigned *roles* describing exactly one capability. For example, an agent that can move along streets is able to execute the move-on-road-order and thus filling the move-on-road-role. Agents can fill different roles at the same time as long as these are not mutual exclusive. A van can transport a wounded while moving on roads. But it cannot load a wounded while driving. Two kinds of agents have to be distinguished: Models for subordinate and non-subordinate agents. The former ones are assumed to behave exactly as they are ordered by the leader agent itself. Thus, it knows which actions the assigned agents will execute in the future. The actions executed by the assigned agents can be considered as actions executed by the leader agent itself. The second case is more complex since the leader agent cannot know how these agents behave. The non-subordinate agents can again be clustered into three groups: Allied, neutral and opponent agents. The former ones are assumed to behave cooperatively. It could even be possible that these agents communicate their desired behavior in order to simplify the collaboration of the alliance. The neutral ones will not communicate their goals and desires. But they will not try to actively disturb the leader agents task. Finally, the opponent agents will try to harm the leader agent as well as its subordinates and actively try to prevent them from reaching their goals.

4.3 Think: Decision Making

The core of the leader agent concept is the decision making framework. The proposed framework should not be restricted to certain kinds of methods. Thus, a SOA approach might be useful. Currently, two types of dispatchers are available for the domain of medical evacuation. One is hard-coded and follows a greedy approach. The other one is optimized by a genetic algorithm (see [7]). However, the hierarchical task decomposition opens possibilities for integrating more sophisticated decision making methods. The fact that on each level of the hierarchy leader agents can only choose from a limited set of actions prunes search space considerably. This makes it possible to simulate leader agents' decision making with computationally expensive methods. One such method is automated planning. Planning is a suitable method, because the wide variety of available techniques ranging from domain-independent to heuristically guided systems opens many possibilities. Additionally, the supported range of target domains is wide. Classical planning techniques are available for restricted domains and there exist techniques that are able to cope with very demanding (e. g. highly uncertain) domains. An important question that arises when planning is applied concerns plan monitoring [14]. Its task is to decide when the currently executed plan should be stopped and a new one should be generated. If the plan is stopped because of execution flaws (e. g. due to unforeseen changes in the environment resulting from wrong or imprecise models) another question arises: Re-plan from scratch or repair the current plan? Although it has been proved by [15] that repairing is at least as hard as re-planning in general, many systems (e.g. a variant of LPG [16]) state that in certain domains repairing pays off in terms of computation time and plan similarity. The latter is also important if the agents have already committed themselves to certain tasks since repaired plans are often similar to the original ones. The monitoring can result in three outcomes: The first one is *Re-plan*, which initiates a re-plan from scratch. Secondly, a *Plan-repair* can be initiated. Another possible outcome is *Continue* denoting that the current plan is further executed. Further execution means that potentially new orders are given. Note that after a *Re-plan* or *Plan-repair* new orders might also be given to subordinate agents. The plan monitoring can easily be integrated into LAMPSys since the monitoring procedure is automatically called in each simulation cycle as part of executing the agent's behavior. In the current version, a *Re-plan* is initiated in case of execution flaws. With this approach, complex decision making is performed deliberatively and the same LAMPS graphs can be used in several domains. The planning domain as well as the planning problem generation have to be adapted for each application area. Note that plan monitoring has to consider incoming orders since these might require re-planning. The decision making of basic agents as described in section 3.4 remains unaffected except that only orders given by its elementary leader agent are executed.

4.4 Act: Give Orders

After a plan has been generated, the assigned subordinate agents have to execute it. Therefore, the plan has to be converted to BML commands for each assigned agent. These have to be transmitted to the corresponding agent. If a leader agent L gives orders to a subordinate agent S three cases exist. Firstly, S can be L's embodying basic agent. Then S can simply execute the orders. Secondly, S can be a leader agent embodied on the same basic agent as L is. Consequently, S can take the orders and execute them. Finally, S can be an agent that is not embodied on the same basic agent as L is. Then L's embodying agent has to communicate the order to S's embodying agent. In all three cases the BML commands are delivered from L to S using the appropriate communication mechanism (direct or via intermediate agents).

5 Conclusion and Future Work

In this paper, we introduced the concept of leader agents. We have integrated leader agents as a framework for automated decision making into the agent-based simulation system LAMPSys. The framework's most important characteristic is the hierarchical structure in which orders are given and executed. It supports a hierarchical task decomposition reducing the number of available actions at each level. Thus, a more efficient decision making is possible. Note that leader agents are not necessarily used to model human decision makers. It is our primary goal to provide decision support by performing many simulations with (nearly) optimal decisions. Currently, it is not intended to model social aspects of human communities. The concept of basic agents is still vital for LAMPSys, because it reduces the modeling effort in comparison to e. g. a stochastic action outcome generator. The reason therefore is that many events can occur during the simulation. Having modeled the agents' responses to these events, they can be used in all types of scenarios. This is an advantage, because having n stochastic components for n types of scenarios would require to adapt each component to all events. For example, weather can influence the speed of helicopters. This influence might be restricted locally. Thus, every helicopter has to calculate the weather's impact on its speed, which can perfectly be modeled with basic agents.

Since only the framework has been introduced, there remains a considerable amount of future work. First of all, we want to couple LAMPSys with several existing planning systems: A metric PDDL planning system [17], a decision theoretic (DT) planner, a hierarchical task network (HTN) planning system and a control knowledge based system. PDDL is interesting, because it is domain-independent and there exist many planning systems. DT planning systems are designed to cope with uncertainty and non-determinism. The last two planning approaches are domain-dependent and can be guided heuristically by the user. Most proposed techniques of [18] are represented by these planning approaches. It is important to evaluate the strengths and weaknesses of the systems. Then we can try to find rules or heuristics that identify situations in which certain systems are superior to others. Another important method is the automatic generation of domain and problem definitions (as defined in [18] for all approaches mentioned above) from the agent's knowledge. Thus, the planning system of each leader agent works on a context-dependent abstract world model. We expect that this automatic search space pruning leads to a significant improvement of the planning times. Since planning time is the limiting factor, we expect better plans in terms of solution quality to be achieved. A first approach for developing this meta-reasoning system might be to pursue the idea of [19]. Additionally, we want to adapt the idea of the Pandemonium [20]. Basically it should be possible to apply several planning systems at the same time and decide from situation to situation which system generates the plan.

References

1. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach (2nd Edition). Series in Artificial Intelligence. Prentice Hall (2002)
2. Weiss, G.: Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. The MIT Press, Berlin (1999)
3. Klügl, F.: Multiagentensimulation. Informatik Spektrum **29**(6) (2006) 421–417
4. Jensen, K.: Coloured Petri Nets: Basic concepts, analysis methods and practical use. Volume 1 of Monographs in Theoretical Computer Science. Springer (1992)
5. Vojnar, T.: Various kinds of petri nets in simulation and modelling. In Stefan, J., ed.: Proceedings of MOSIS '97. Volume 1. (1997) 227–232
6. Richter, G.: Flip-tick architecture: A cycle-oriented architecture for distributed problem solving. Technical report, GMD, Sankt Augustin, Germany (1999)
7. Mies, C., Wolters, B., Steffens, T.: LAMPSys - An experiment-platform for the optimization of processes and agent-behaviors. KI Zeitschrift (2) (2009) 42–45
8. Steffens, T., Zoeller, T., Huegelmeyer, P.: Agent-based modeling of processes and scenarios with high-level petri nets. Technical report, Fraunhofer IAIS (July 2006)
9. Huegelmeyer, P., Steffens, T., Zoeller, T.: Specifying and simulating modern warfare scenarios with ITSimBw. In Perrone, L.F., Wieland, F.P., Liu, J., Nicol, B.G.L.D.M., Fujimoto, R.M., eds.: Proceedings of WinterSim 2006, ACM (2006)
10. Usov, A., Beyel, C.: Simulating interdependent critical infrastructures with simcip. European CIIP Newsletter **4**(3) (2008)
11. Huegelmeyer, P., Schade, U., Zoeller, T.: Application of BML to Inter-Agent Communication in the ITSimBw Simulation Environment. In Biller, B., Hsieh, M.H., Shortle, J., Tew, J.D., Barton, R.R., eds.: Proceedings of WinterSim 2007. (2008)
12. Sowa, J.F.: Conceptual graphs. In: Handbook of Knowledge Representation, Elsevier (2008) 213–237
13. Siciliano, B., Khatib, O.: Springer Handbook of Robotics. Springer (2008)
14. Micalizio, R.: On-line monitoring of plan execution: A distributed approach. PhD thesis, Universit di Torino, Torino, Italy (2007)
15. Nebel, B., Koehler, J.: Plan Reuse versus Plan Generation: A Theoretical and Empirical Analysis. Technical Report RR-93-33, DFKI GmbH (1993)
16. Fox, M., Gerevini, A., Long, D., Serina, I.: Plan Stability: Replanning versus Plan Repair. In: ICAPS 2006, AAAI Press (2006)
17. Fox, M., Long, D.: PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains. Artificial Intelligence Research **20** (2003) 51–124
18. Ghallab, M., Nau, D., Traverso, P.: Automated Planning: Theory and Praxis. Morgan Kaufmann Publishers, Boston, MA, USA (May 2004)
19. Hartanto, R., Hertzberg, J.: Fusing DL Reasoning with HTN Planning. In: KI 2008: Advances in Artificial Intelligence. LNCS, Springer (September 2008) 62–69
20. Selfridge, O.: Pandemonium: A paradigm for learning. In: Mechanisation of Thought Processes, Her Majesty's Stationary Office (December 1959) 511–531

The 32nd Annual Conference on
Artificial Intelligence (KI-2009)
September 15-18, 2009, Paderborn, Germany

Workshop on

Relational Approaches to Knowledge Representation and Learning

http://www.fernuni-hagen.de/wbs/relkrl09.html

Gabriele Kern-Isberner & Christoph Beierle (Eds.)

KI 2009

AI And Automation

Workshop Organizers and Co-Chairs

Gabriele Kern-Isberner	Technische Universität Dortmund, Germany
Christoph Beierle	FernUniversität in Hagen, Germany

Program Committee

Salem Benferhat	Université d'Artois, Lens, France
Gerd Brewka	Universität Leipzig, Germany
James P. Delgrande	Simon Fraser University, Burnaby, BC, Canada
Jürgen Dix	TU Clausthal-Zellerfeld, Germany
Eduardo Fermé	Universidade da Madeira, Portugal
Andreas Herzig	Université Paul Sabatier, Toulouse, France
Pascal Hitzler	Universität Karlsruhe (TH), Germany
Antonis C. Kakas	University of Cyprus, Cyprus
Kristian Kersting	Fraunhofer IAIS, Universität Bonn, Germany
Thomas Lukasiewicz	University of Oxford, UK
Torsten Schaub	Universität Potsdam, Germany
Emil Weydert	University of Luxembourg, Luxembourg

Additional Reviewer

Maurício Reis

Preface

Christoph Beierle[1] and Gabriele Kern-Isberner[2]

[1] Dept. of Computer Science, FernUniversität in Hagen, 58084 Hagen, Germany
[2] Dept. of Computer Science, TU Dortmund, 44221 Dortmund, Germany

Knowledge representation encompasses a variety of methods and formalisms to encode and process all types of knowledge, belief, and information. It provides the theoretical foundation for rational and intelligent behaviour in real environments, focusing on topics like default logics and uncertain reasoning, belief change, ontologies, and argumentation, among many others. Moreover, in a thematical respect, knowledge representation is closely related to the areas of machine learning and knowledge discovery the methods of which allow the acquisition of useful information to build up knowledge bases.

Knowledge representation has made substantial progress over the last decade by devising sophisticated methods for inference and reasoning. Nevertheless, the connection to learning still holds undeveloped potential in methodological and technical respects which might be crucial for practical applications. Furthermore, the handling of relational information, i.e. the explicit representation of knowledge about objects and its linking to knowledge about classes, is still a challenge for many subareas of knowledge representation. Ontologies, logic programming and probabilistic relational models are just some important examples of areas of research that address both of these points.

The particular focus of the KI-2009 workshop *Relational Approaches to Knowledge Representation and Learning* (www.fernuni-hagen.de/wbs/relkrl09.html), which was organized by the *GI-Fachgrupppe Wissensrepräsentation und Schließen*, was to strengthen the connection between knowledge representation and learning by focusing on relational and first-order approaches to all areas of knowledge representation and learning.

The workshop started with an invited talk *Relevance, Conditionals, and Defeasible Reasoning* by James Delgrande. He investigates the notion of relevance in the context of defeasible reasoning, and presents an approach for incorporating irrelevant properties in a conditional knowledge base. It is argued that this approach exactly captures defeasible reasoning with commonsense normative conditionals.

The beliefs of an agent can be represented by a designated predicate in a selfreferential first-order language. However, such first-order theries often lead to pardoxes. In his paper *Log_A***B***: An Algebraic Logic of Belief*, Haythem Ismail develops a family of algebraic logics of beliefs that is almost as expressive as first-order theories, but at the same time weak enough to avoid pardoxes of self-reference.

Probabilistic logic is the general topic of the following three papers. With the origins of probabilistic logic based on propositional logic, the introduction of probabilistic graphical models, in particular the popular Markov and

Bayesian networks, enabled practical applications and spurred the research efforts in this area.

In his contribution *First-Order Probabilistic Conditional Logic - Introduction and Representation*, Jens Fisseler presents a first-order extension of a propositional probabilistic representation formalism, allowing in particular the representation of prababilistic *if-then* rules. The semantics employs the principle of maximum entropy, selecting a model that is as unbiased as possible. In order to tame the complexity of the resulting optimization problem to be solved, sufficient syntactic criteria for its simplification are developed.

Currently, the most prominent approaches to lifting propositional probabilistic logic to the first-order case, are Bayesian logic programs (BLP) and Markov logic networks (MLN). Both BLPs and MLNs as well as a new approach for using maximum entropy methods in a relational context are supported by the KReator toolbox that aims at providing a common and simple interface for working with different relational probabilistic approaches. This integrated development environment is presented by Marc Finthammer, Sebastian Loh, and Matthias Thimm in their contribution *Towards a Toolbox for Relational Probabilistic Knowledge Representation, Reasoning, and Learning.*

In his paper *Representing Statistical Information and Degrees of Belief in First-Order Probabilistic Conditional Logic*, Matthias Thimm proposes a formal semantics for first-order probabilistic conditionals that matches common sense and avoids ambiguities between statistical and subjective interpretations. Moreover, he shows how the principle of maximum entropy can also be applied in this framework, and proves formal properties of the resulting inference operator.

Finally, the last two papers deal with approaches based on logic programming. The paper *Reinforcement Learning for Golog Programs* by Daniel Beck and Gerhard Lakemeyer presents an approach of using the action language Golog to constrain the action state space to be explored in a reinforcement learning situation. The authors develop a Golog dialect using a semi-Markov Decision Process representation, and give a completely declarative specification of a learning Golog interpreter.

In general, conflicts may arise when pieces of information depend on each other, and so, Patrick Krümpelmann makes dependencies the atomic concept to study conflicts. In *Towards Dependency Semantics for Conflict Handling in Logic Programs*, he presents a formal framework for conflict resolution that is based on dependencies and involves consequences and preferences. As an application, he shows how this can be related to answer set semantics and the causal rejection principle.

We would like to thank all Program Committee members as well as the additional external reviewer Maurício Reis for detailed and high-quality reviews for all submitted papers. We are also grateful to the Gesellschaft für Informatik, the TU Dortmund, and the FernUniversität in Hagen for supporting this workshop.

August 2009 Gabriele Kern-Isberner and Christoph Beierle

Relevance, Conditionals, and Defeasible Reasoning

James P. Delgrande

School of Computing Science,
Simon Fraser University,
Burnaby, B.C.,
Canada V5A 1S6.
jim@cs.sfu.ca

This talk discusses the notion of relevance as it pertains to statements of normality and defeasible reasoning in Artificial Intelligence. The role of relevant properties in defeasible reasoning is first covered, along with a discussion of how relevance has been addressed in different approaches to nonmonotonic reasoning. Following this, an approach for incorporating irrelevant properties in a conditional knowledge base is presented; and a notion of defeasible reasoning is introduced, based on this approach.

In the approach, a closure operation is defined, so that from a theory of defeasible conditionals an extension is obtained wherein irrelevant properties are satisfactorily incorporated. The approach is shown to have desirable formal properties and handles various commonsense examples appropriately. It is also shown that this approach can be captured in an iterative definition. In conclusion, it is argued that defeasible reasoning with commonsense normative conditionals is exactly captured via a sufficiently strong logic of defeasible conditionals together with this means of handling relevance.

$\boldsymbol{Log_A}$B: An Algebraic Logic of Belief

Haythem O. Ismail

German University in Cairo
Department of Computer Science
haythem.ismail@guc.edu.eg

Belief is usually viewed as a relation between a believing agent and a believed entity, typically a proposition or a sentence. Logics of belief come in two main flavors: the modal and the syntactical. Modal approaches represent belief by a modal operator and employ some version of possible-worlds semantics. Syntactical theories employ self-referential first-order languages, where belief is represented by a (typically) dyadic predicate of agents and sentences of the language. The semantics is standard Tarskian semantics, but complications arise due to the need to employ theories of arithmetic or string manipulation. On one hand, first-order logics are more expressive and more well-understood than their modal rivals. On the other hand, a result by Thomason (following a similar result by Montague for the case of knowledge) shows that, assuming some desirable properties of belief, first-order doxastic theories are paradoxical, whereas modal ones are not.

Log_A**B** is a family of logics of belief. It holds a middle ground between the expressive, but prone to paradox, syntactical first-order theories and the often inconvenient, but safe, modal approaches. In this report, the syntax and semantics of Log_A**B** are presented. Log_A**B** is algebraic in the sense that it only contains terms, algebraically constructed from function symbols. No sentences are included in a Log_A**B** language. Instead, there are terms of a distinguished syntactic type that are taken to denote propositions. In the Log_A**B** ontology, propositions are structured in a Boolean lattice. This gives us, almost for free, all standard truth conditions, standard notions of consequence and validity, and an individuation of propositions that is neither too fine-grained, nor too coarse-grained, for a doxastic logic.

It turns out that recognizing propositions as first-class inhabitants of our ontology has the benefit of avoiding the doxastic paradoxes referred to above. In particular, Log_A**B** holds a middle ground between modal and first-order syntactical theories of belief. On one hand, it is almost as expressive as the first-order theories; on the other hand, it is weak just enough to avoid the paradoxes to which those theories are susceptible. A number of results are proved regarding paradoxical self-reference. They are shown to strengthen previous results, and to point to possible new approaches to circumventing paradoxes in syntactical theories of belief.

First-Order Probabilistic Conditional Logic
Introduction and Presentation

Jens Fisseler

Faculty of Mathematics and Computer Science
FernUniversität in Hagen
Hagen, Germany
jens.fisseler@fernuni-hagen.de

Knowledge representation and reasoning is one of the main research topics of artificial intelligence. As most real-world knowledge is uncertain rather than certain, knowledge representation formalisms should be able to deal with this uncertainty. One approach to represent and process uncertain knowledge is probability theory, which, with the introduction of *probabilistic graphical models*, has seen increasing research interest during the last two decades. Markov and Bayesian networks are two well-known classes of probabilistic graphical models, but only allow the representation of propositional probabilistic knowledge.

As many real-world knowledge representation tasks require the ability to represent uncertain knowledge about a varying number of objects and their (uncertain) relationships, several approaches for combining probabilistic graphical models and some subset of first-order logic have been developed. The best known of these formalisms are *probabilistic relational models* (*PRMs*), *Bayesian logic programs* (*BLPs*), and *Markov logic networks* (*MLNs*). Although their models are defined by "templates", specified by using some subset of first-order logic, for inference, these formalisms work at a propositional level: PRMs and BLPs induce a Bayesian network, whereas MLNs induce a Markov network. The formulas of PRMs and BLPs are parametrized with conditional probability functions, whereas a MLN consists of weighted formulas. Therefore, MLNs are not as easily comprehensible, and it is also difficult to specify them by hand, e.g. as background knowledge for learning. However, formalisms which are mapped to Bayesian networks have difficulties representing circular dependencies.

This paper introduces an alternative approach for combining a subset of first-order logic and probabilistic models which allows to specify models via probabilistic *if-then*-rules, so-called *conditionals*. Because these probabilistic conditionals do not specify a unique model, we propose to use the principle of maximum entropy as a model selection criterion, which yields a convex optimization problem, with one optimization variable for each ground instance of a first-order probabilistic conditional. As solving this optimization problem is computationally infeasible for all but the smallest sets of ground instances, we develop syntactic conditions which ensure that some or all ground instances of the same conditional "share" the same entropy-optimal parameter value, which greatly simplifies the entropy-optimization problem. The resulting formalism is similar to Markov logic, but allows for an adequate modeling of *if-then*-rules, as well as a quantization of its models via (conditional) probabilities.

Towards a Toolbox for Relational Probabilistic Knowledge Representation, Reasoning, and Learning

Marc Finthammer[1], Sebastian Loh[2], and Matthias Thimm[2]

[1] Department of Computer Science, FernUniversität in Hagen, Germany
[2] Department of Computer Science, Technische Universität Dortmund, Germany

Probabilistic inductive logic programming (or *statistical relational learning*) is a very active field in research at the intersection of logic, probability theory, and machine learning. This area investigates methods for representing probabilistic information in a relational context for both reasoning and learning. Many researchers developed extensions of propositional probabilistic models to the first-order case in order to take advantage of methods and algorithms already developed. Among these are the well-known Bayesian logic programs and Markov logic networks which extend respectively Bayes networks and Markov nets and are based on knowledge-based model construction techniques. Until now, a wide variety of different proposals have been made that support relational probabilistic knowledge representation. Due to this wide variety there are only few comprehensive comparisons between different proposals available. Although many approaches have been prototypically implemented an easy access to the use of these implementations in a general manner is not yet possible.

This paper presents KReator, a versatile and easy-to-use toolbox for statistical relational learning currently under development. KReator aims at providing a common and simple interface for representing, reasoning, and learning with different relational probabilistic approaches. Currently, the development of KReator is still in a very early stage but already supports Bayesian logic programs, Markov logic networks, and in particular a new approach for using maximum entropy methods in a relational context. KReator is an integrated development environment for representing, reasoning, and learning with relational probabilistic knowledge. KReator is written in Java and thus is designed using the object-oriented programming paradigm. Moreover, it is platform-independent and it facilitates several architectural and design patterns such as model-view control, abstract factories, and command patterns. Central aspects of the design of KReator are *modularity*, *extensibility*, *usability*, *reproducibility*, and its intended application in scientific research.

In this paper, we give an overview on the system architecture of KReator and describe its design. Moreover, we illustrate its usage and its functionalities. KReator is available under the GNU General Public License and can be obtained from `http://ls6-www.cs.uni-dortmund.de/kreator/`.

Representing Statistical Information and Degrees of Belief in First-Order Probabilistic Conditional Logic

Matthias Thimm

Information Engineering Group, Department of Computer Science,
Technische Universität Dortmund, Germany

Employing maximum entropy methods on probabilistic conditional logic has proven to be a useful approach for commonsense reasoning. Yet, the expressive power of this logic and similar formalisms is limited due to their foundations on propositional logic and in the past few years a lot of proposals have been made for probabilistic reasoning in relational settings. For example, two of the most prominent approaches for extending propositional approaches to the relational case are Bayesian logic programs and Markov logic networks. While Bayesian logic programs extend Bayes nets using a logic programming language Markov logic networks extend Markov nets using a restricted form of first-order logic. Both frameworks use knowledge-based model construction techniques to reduce the problem of probabilistic reasoning in a relational context to probabilistic reasoning in a propositional context. In both frameworks—and also in most other approaches—this is done by appropriately grounding the parts of the knowledge base that are needed for answering a particular query and treating this grounded parts as a propositional knowledge base. While most approaches to relational probabilistic reasoning employ graphical models of probabilistic reasoning, in this paper we take another direction by lifting probabilistic conditional logic to the first-order case and applying maximum entropy methods for reasoning.

In probabilistic conditional logic knowledge is captured using conditionals of the form $(B \mid A)[\alpha]$ with some formulas A, B and $\alpha \in [0,1]$. A probabilistic conditional of this form partially describes an (unknown) probability distribution P^* by stating that $P^*(B \mid A) = \alpha$ holds. In contrast to Bayes nets probabilistic conditional logic does not demand to fully describe a probability distribution but only to state constraints using conditionals on it. We take a specific focus on representing relational probabilistic knowledge by differentiating between different intuitions on relational probabilistic conditionals, namely between statistical interpretations and interpretations on degrees of belief. We introduce the syntax of a relational probabilistic conditional logic and propose a specific semantics that captures these different intuitions on the interpretation of relational conditionals. We develop a list of desirable properties on an inference procedure for this logic and propose a specific inference procedure that fulfills these properties. We furthermore discuss related work and give some hints on future research.

Reinforcment Learning for Golog Programs

Daniel Beck and Gerhard Lakemeyer

Department of Computer Science
RWTH Aachen University
Aachen, Germany
{dbeck|gerhard}@cs.rwth-aachen.de

In classical reinforcement learning (RL) and Markov Decision Processes (MDPs), we are given a set of states, actions which stochastically take us from a given state into one of a number of states, and a reward function over states. The goal of learning is to find the optimal policy, which tells us for each state which action to select to maximize our expected reward. In principle this is well understood with methods such as Q-learning solving the problem. However, for most practical applications the huge state and action space is a concern, as explicit representations usually are not viable computationally. To address this problem, state abstraction mechanisms have been explored, including FOMDPs, which employ first-order logic to characterize a possibly infinite state space using a finite set of formulas.

In this paper, we take this idea further by also constraining the action space using programs written in the action language Golog. Roughly, instead of a state and a set of primitive actions to choose from, we are given a formula describing the current state and a program we need to follow. In the extreme case, when the program is completely deterministic, there is nothing to learn, as the program tells us exactly what the next action is. However, in general the program allows for non-deterministic choices, and here we again need to learn what choices are the best ones in terms of maximizing expected rewards. As we will see, the idea of Q-learning can be adapted to this setting.

More precisely, we present a method to compute, for a given reward function and Golog program, first-order state formulas describing the possible states before the program is executed. Roughly, these formulas specify sets of states which are equivalent in the sense that the expected rewards are identical when following a policy which is compliant with the program. Moreover, only those properties of the states which are relevant to the expected reward are reflected in those state formulas.

Then, we construct a joint semi-MDP (SMDP) over a state space which is made up of tuples consisting of a subprogram of the given program which starts off with a non-deterministic choice and a corresponding state formula.

We give the semantics for our new Golog dialect QGolog which incorporates reinforcement learning techniques to learn the optimal decisions for the choice points of a program by means of executing it and observing the outcomes. We do so in a completely declarative manner, which we feel is more transparent and better lends itself to formal analysis.

Towards Dependency Semantics for Conflict Handling in Logic Programs

Patrick Krümpelmann

Information Engineering
Technische Universität Dortmund
Germany
patrick.kruempelmann@udo.edu

In this paper, we present a dependency based framework for the definition of semantics for non conflict free non-monotonic belief bases. Here, we utilize extended logic programs for the representation of belief bases. The intention of this work is to lay a foundation for the definition and analysis of methods for handling conflicts in such belief bases and to give first results.

We define a general framework of dependencies for extended logic programs and sequences of programs which we will then use to analyse and compare common approaches as well as to find improvements of these. This framework is intended to be based on logic programming, but to detach from the syntactic approach towards a semantic view of conflict handling in the dynamics of logic programming. It provides dependency based semantics for sequences of logic programs. Dependencies are generated by means of rules, and in particular by the default assumptions of these. A dependency relation is generated by a set of rules. Based on this dependency relation, methods for handling conflicting information are defined by determining sets of dependencies which cause the conflict. These can then be used to resolve the conflict by defining conflict free subsets of the dependency relation and extensions for these. We present a powerful set of tools for the definition of these methods which lead to different semantics. These semantics can be used to acquaint more in-depth insights into the detection and elimination of conflicts as it gives a formal definition of these methods. We construct the framework in a modular way by making key definitions variable. Hereby, several instantiations of this framework can be defined for means of comparison, analysis and for different scenarios of application.

In this work, we start elaborating on the properties and possibilities of this kind of semantics and show similarities and differences with a big class of current approaches based on the causal rejection of rules. Moreover, we show that this can lead to deeper insights and to the definition of improved ways of treating dynamics of beliefs in diverse settings. This formal representation can, and is intended to, find new approaches to conflict handling and provides the means for general and powerful operations, incorporating many aspects of dynamics in non-monotonic logics. First results in this direction are presented in this paper by defining different instantiations of the presented framework and demonstrating the differences in behaviour on benchmark examples. We show how the defined semantics coincide with answer set semantics on conflict free bases and poses desirable properties for conflicting belief bases.

The 32nd Annual Conference on
Artificial Intelligence (KI-2009)
September 15-18, 2009, Paderborn, Germany

Workshop on

Self-X in Mechatronics and other Engineering Applications

http://wwwhni.uni-paderborn.de/self-x-in-engineering

Benjamin Klöpper (Ed.)

KI 2009

AI And Automation

Workshop Organizers and Co-Chairs

Benjamin Klöpper	Universität Oldenburg, Germany kloepper@hni.upb.de

Program Committee

Jürgen Belz	Hella KGaA Hueck & Co., Lippstadt, Germany
Stefan Edelkamp	University of Bremen, TZI, Germany
Lothar Hotz	University of Hamburg, Germany
Thorsten Krebs	University of Hamburg, Germany
Jrgen Sauer	University of Oldenburg, Germany
Bernd Schattenberg	University of Ulm, Germany
René Schumann	University of Frankfurt, Germany

Preface

Benjamin Klöpper and Wilhelm Dangelmaier

Heinz Nixdorf Institute
University of Paderborn
Germany
{kloepper|whd}@hni.upb.de

Classical engineering applications such as mechanical, electrical or production engineering considerably benefit by the extensive application of information technology. Consequently, a new aim in engineering disciplines is the development of intelligent systems. Commonly, this term is applied to systems which are able to carry out tasks for their owner or users in a rather efficient and autonomous way. Obviously, autonomous behavior is closely related to self-x abilities. The term self-x is applied if a technical system is able to observe and manipulate its behavior in way which usually requires human intervention. Important examples of such abilities are self-diagnosis, self-repair or -healing, self-coordination or self-optimization. Methods used for the implementation often originate from computational intelligence and related computer science areas. The workshop on "Self-X in Engineering Application" is devoted to the efforts to implement these methods in engineering applications in order to achieve the vision of intelligent and autonomous systems. The idea of the workshop originates in the Collaborative Research Centre 614 "Self-optimizing concepts and structures in mechanical engineering" (CRC 614), which works on the definition of a new school of development of technical systems. Thus, the focus workshop is not only on the implementation of smart algorithms, but also on the great challenge to envision and design systems with self-x properties.

Benjamin Klöpper
Wilhelm Dangelmaier

Self-Monitoring and Control for Embedded Systems using Hybrid Constraint Automata

Paul Maier and Martin Sachenbacher

Technische Universitt Mnchen
Germany
{maierpa|sachenba}@in.tum.de

Many of today's mechatronic systems - such as automobiles, automated factories or chemical plants - are a complex mixture of hardware components and embed-ded control software, showing both continuous (vehicle dynamics, robot motion) and discrete (software) behavior. The problems of estimating the internal dis-crete/continuous state and automatically devising control actions as intelligent reaction are at the heart of self-monitoring and self-control capabilities for such systems. In this paper, we address these problems with a new integrated approach, which combines concepts, techniques and formalisms from AI (constraint optimi-zation, hidden markov model reasoning), fault diagnosis in hybrid systems (sto-chastic abstraction of continuous behavior), and hybrid systems verification (hybr-id automata, reachability analysis). Preliminary experiments with an industrial filling station scenario show promising results, but also indicate current limitations.

Self-optimising flexible assembly systems

Christian Brecher[1], Nicolas Pyschny[1], Peter Loosen[2], Max Funck[2], Valentin Morasch[2], Robert Schmitt[3] and Alberto Pavim[4]

[1] Fraunhofer Institute for Production Technology IPT, Aachen
[2] Fraunhofer Institute for Laser Technology ILT, Aachen
[3] Laboratory for Machine Tools WZL, RWTH Aachen
[4] Nicolas.Pyschny@ipt.fraunhofer.de,Max.Funck@ilt.fraunhofer.de, A.Pavim@wzl.rwth-aachen.de

In turbulent environments, today's production systems have to exhibit an in-creased flexibility and mutability to deal with dynamically changing conditions, objectives and an increasing number of product variants. For automated systems, especially assembly systems, an increased flexibility usually induces a significantly higher complexity, whereby the efforts for planning and programming, but also setups and reconfiguration expand. In this paper a definition and concepts of self-optimising assembly systems are presented to describe a possible means to reduce the planning efforts in complex production systems. The concept of self-optimisation in assembly systems will be derived from a theoretical approach and will be transferred to a specific application scenario - the automated assembly of a miniaturised solid state laser - where the challenges of unpredictable influences from e.g. component tolerances can be overcome by help of self-optimisation.

Solution Patterns for the Development of Self-Optimizing Systems

Roman Dumitrescu, Jürgen Gausemeier, Wilhelm Dangelmaier and Benjamin Klöpper

Heinz Nixdorf Institute
University of Paderborn
Germany
roman.dumitrescu@hni.uni-paderborn.de

Machines are everywhere. They produce, they transport. Machines facilitate work and assist. The increasing penetration of mechanical engineering by information technology enables considerable benefits. The arising new discipline is referred to by the term mechatronics, which expresses the close integration of mechanics, electronics, control engineering and software engineering in order to improve the behavior of a technical system. The conceivable development of information and communication technology will enable mechatronic systems with inherent partial intelligence. We refer to this by using the term "self-optimization". Self-optimizing systems react autonomously and flexibly on changing operational conditions. They are able to learn and to optimize their behavior during operation. The design of such systems is an interdisciplinary task. Mechanical, electrical, control and software engineers are involved as well as experts from mathematical optimization and artificial intelligence. During the development phase "conceptual design" developers from various domains are looking for solution patterns to reuse once proven ways of realizing technical functions. As a consequence a domain-spanning specification of solution patterns applied in mechatronic systems is necessary. In this contribution we present a specification for the domain-spanning modeling of solution patterns for self-optimizing systems, called Active Patten for Self-Optimization (APSO), as well as a tool-based approach for the domain-spanning use of APSO. By a hybrid planning architecture one specific active pattern will be exemplified.

Buildings Blocks and Prototypical Implementation of a Hybrid Planning Architecture

Phillip Adelt[1] and Benjamin Klöpper[2]

[1] C-Lab
Heinz Nixdorf Institute
[2] University of Paderborn
Germany
padelt@c-lab.de

Self-optimizing systems are a new class of technical systems. Their ability to ad-just themselves to changing environments and changing requirements enables a superior systems performance. Planning is an important mean to perform the process of self-optimization. Unfortunately, planning employ discrete models while. This results in unavoidable deviations during the execution of a plan. We introduce a hybrid planning approach which enables the mechatronic systems to react appropriately to these deviations.

Author Index