

Joscha Bach
Stefan Edelkamp (Eds.)

LNAI 7006

KI 2011: Advances in Artificial Intelligence

34th Annual German Conference on AI
Berlin, Germany, October 2011
Proceedings



Springer

Lecture Notes in Artificial Intelligence 7006

Subseries of Lecture Notes in Computer Science

LNAI Series Editors

Randy Goebel

University of Alberta, Edmonton, Canada

Yuzuru Tanaka

Hokkaido University, Sapporo, Japan

Wolfgang Wahlster

DFKI and Saarland University, Saarbrücken, Germany

LNAI Founding Series Editor

Joerg Siekmann

DFKI and Saarland University, Saarbrücken, Germany

Joscha Bach Stefan Edelkamp (Eds.)

KI 2011:Advances in Artificial Intelligence

34th Annual German Conference on AI
Berlin, Germany, October 4-7, 2011
Proceedings

Series Editors

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

Volume Editors

Joscha Bach
Humboldt-University of Berlin
Berlin School of Mind and Brain
Unter den Linden 6, 10099 Berlin, Germany
E-mail: joscha.bach@hu-berlin.de

Stefan Edelkamp
University of Bremen
Faculty 3, Mathematics and Computer Science
P.O. Box 33 04 40, 28334 Bremen, Germany
E-mail: edelkamp@tzi.de

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-24454-4 e-ISBN 978-3-642-24455-1
DOI 10.1007/978-3-642-24455-1
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011936978

CR Subject Classification (1998): I.2, H.4, F.1, H.2.8, I.2.6, H.5.2

LNCS Sublibrary: SL 7 – Artificial Intelligence

© Springer-Verlag Berlin Heidelberg 2011

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

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

Printed on acid-free paper

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

Preface

The yearly German Conference on Artificial Intelligence is the premier forum for German research in artificial intelligence, and attracts numerous international guests, too. KI 2011, the 34th event of the series, reflected a long-standing tradition, and continued to mirror the trends and developments of the science. The 2011 conference took place in Berlin during October 4–7, in co-location with INFORMATIK 2011, the 41st Annual Meeting of the Gesellschaft für Informatik, and MATES 2011, the 9th German Conference on Multi-Agent System Technologies.

Since its inception, artificial intelligence has been at the vanguard of computer science, and today, its applications and methods have become so widespread and ubiquitous that most people simply take them for granted. Its contributions have so thoroughly permeated the fabric of our digital lives that they have become almost invisible. Artificial intelligence has become a mainstay, and is integrated everywhere from entertainment electronics and Internet technologies to transport and industry applications, from finance to agriculture, from art to electronic reading, from cognitive science to medicine. KI 2011 reflected this by its focus on advances “Towards a Smart World—Evolving Technologies in Artificial Intelligence.”

This volume contains the technical papers of KI 2011. For the technical program, we received 81 submissions, of which the Program Committee accepted 32 after a rigorous review (25 of these as full papers, and 7 as short papers, limited to five pages). The volume also includes three programmatic contributions corresponding to invited talks of KI 2011: “Everything You Always Wanted to Know About Planning (But Were Afraid to Ask)” by Jörg Hoffmann, “Why We Need Evolutionary Semantics” by Luc Steels, and “General Game Playing in AI Research and Education” by Michael Thielscher. A fourth invited talk by Sven Koenig concerned “Making Good Decisions: Case Studies in Planning and Coordination.”

On the first day of the conference, PhD students presented and discussed their work in a doctoral consortium, which ended in a panel discussion on methods, practices and philosophy of artificial intelligence, held by Bernhard Nebel, University of Freiburg; Franz Baader, TU Dresden; Ulrich Frank, University of Duisburg-Essen; and Ingo J. Timm, University of Trier.

The 1st German Open in General Game Playing (GO-GGP) was organized by the competition chairs Peter Kissmann and Tim Federholzner.

In addition to the main program, KI 2011 featured seven workshops with many additional research presentations:

- 5th Workshop on Emotion and Computing—Current Research and Future Impact. Chair: Dirk Reichardt.
- 6th Workshop on Behavior Monitoring and Interpretation (BMI 2011). Chairs: Björn Gottfried and Hamid Aghajan.
- Third Workshop on Dynamics of Knowledge and Belief—Evolving Knowledge in Theory and Applications. Chairs: Gabriele Kern-Isbner and Christoph Beierle.
- 26th Workshop on Planning Scheduling, Configuration Design (PuK 2011). Chairs: Jürgen Sauer, Stefan Edelkamp and Bernd Schattenberg.
- First International Workshop on Algorithmic Intelligence. Chairs: Carsten Elfers, Rune Jensen, Hartmut Messerschmidt and Rasmus Pagh.
- Workshop on Visibility in Information Spaces and in Geographic Environments. Chairs: Andreas Henrich, Christoph Schlieder and Ute Schmid.
- Workshop on Context-Aware Intelligent Assistance (CAIA 2011). Chairs: Stefan Mandl, Bernd Ludwig and Florian Michahelles.

Giorgio Grisetti, Andreas Nüchter and Alexander Kleiner offered a full-day tutorial on “SLAM to the Rescue: A Hands-On Tutorial on Using State-of-the-Art SLAM Algorithms in Harsh Environments.” Various robotics researchers from Berlin universities provided demonstrations of their work, especially Verena Hafner, Manfred Hild, Clemens Eppner, Daniel Seifert, and their teams.

The organization of a conference like this one is only possible with the support of many individuals. First of all, the organizers wish to thank the authors for their contributions. We had a very strong and competent Program Committee consisting of 75 members, which ensured that each submission underwent several thorough and timely reviews.

The KI 2011 conference team included:

- Co-location: Doris Fähndrich (TU Berlin)
- Workshop Chair: Bernd Schattenberg (University of Ulm)
- Tutorial Chair: Sebastian Kupferschmid (University of Freiburg)
- Doctorial Consortium Chair: René Schumann (University of Applied Sciences Western Switzerland)
- Industry Liaisons: Roman Englert (T-LABs)

We extend our thanks to these and all other people and institutions that made this happening a success, especially the Deutsche Forschungsgemeinschaft (DFG), the Gesellschaft für Informatik (GI), the Technologie-Zentrum Informatik in Bremen (TZI), the Technical University of Berlin, Springer, and the Elsevier *Journal for Artificial Intelligence* (AIJ).

Organization

General Chair

Stefan Edelkamp Universität Bremen

Local Chair

Joscha Bach Humboldt-Universität zu Berlin

Workshop Chair

Bernd Schattenberg Universität Ulm

Tutorial Chair

Sebastian Kupferschmid Universität Freiburg

Doctorial Consortium Chair

René Schumann NII, Tokyo

Program Committee

| | |
|----------------------|--|
| Klaus-Dieter Althoff | University of Hildesheim, Germany |
| Tamim Asfour | Karlsruhe Institute of Technology, Germany |
| Amit Banerjee | University of Nevada, Reno, USA |
| Sven Behnke | University of Bonn, Germany |
| Maren Bennewitz | University of Freiburg, Germany |
| Ralph Bergmann | University of Trier, Germany |
| Marc Cavazza | University of Teesside, UK |
| Daniel Cernea | University of Kaiserslautern, Germany |
| Eliseo Clementini | University of L'Aquila, Italy |
| Cristobal Curio | Max Planck Institute for Biological Cybernetics, Germany |
| Kerstin Dautenhahn | University of Hertfordshire, UK |
| Frank Dylla | University of Bremen, Germany |
| Dominik Endres | University of Tübingen, Germany |
| Florian Eyben | Munich University of Technology, Germany |

VIII Organization

| | |
|------------------------|---|
| Udo Frese | University of Bremen, Germany |
| Stefan Funke | University of Stuttgart, Germany |
| Johannes Fürnkranz | TU Darmstadt, Germany |
| Christopher Geib | University of Edinburgh, UK |
| Bjoern Gottfried | University of Bremen, Germany |
| Horst-Michael Gross | Ilmenau University of Technology, Germany |
| Jens-Steffen Gutmann | Evolution Robotics, Pasadena, USA |
| Martin Günther | University of Osnabrück, Germany |
| Fred Hamker | Chemnitz University of Technology, Germany |
| Malte Helmert | University of Freiburg, Germany |
| Dominik Henrich | University of Bayreuth, Germany |
| Joachim Hertzberg | University of Osnabrück, Germany |
| Otthein Herzog | University of Bremen and Jacobs University Bremen, Germany |
| Jörg Hoffmann | INRIA, France |
| Gabriele Kern-Isberner | Dortmund University of Technology, Germany |
| Peter Kissmann | University of Bremen, Germany |
| Alexander Kleiner | University of Freiburg, Germany |
| Roman Kontchakov | Birkbeck College, UK |
| Oliver Kramer | University of Oldenburg, Germany |
| Ralf Krestel | Leibniz Universität Hannover, Germany |
| Rudolf Kruse | Otto von Guericke University Magdeburg, Germany |
| Torsten Kröger | Stanford University, USA |
| Bogdan Kwolek | Rzeszow University of Technology, Poland |
| Kai-Uwe Kühnberger | University of Osnabrück, Germany |
| Gerhard Lakemeyer | RWTH Aachen University, Germany |
| Tobias Lang | Freie Universität Berlin, Germany |
| Hagen Langer | University of Bremen, Germany |
| Volker Lohweg | inIT - Institute Industrial IT, Germany |
| Benedikt Löwe | Universiteit van Amsterdam, The Netherlands |
| Katja Markert | University of Leeds, UK |
| Robert Mattmüller | University of Freiburg, Germany |
| Bärbel Mertsching | GET Lab, Paderborn University, Germany |
| Hartmut Messerschmidt | University of Bremen, Germany |
| Bernd Michaelis | Otto von Guericke University Magdeburg, Germany |
| Ralf Möller | Hamburg University of Technology, Germany |
| Oliver Niggemann | inIT - Institute Industrial IT, Germany |
| Justus Piater | University of Innsbruck, Austria |
| Felix Putze | Karlsruhe Institute of Technology, Germany |
| Jochen Renz | Australian National University, Australia |
| Gerhard Rigoll | Munich University of Technology, Germany |
| Alessandro Saffiotti | Örebro University, Sweden |

| | |
|---------------------|---|
| Jürgen Sauer | University of Oldenburg, Germany |
| Bernd Schattenberg | Ulm University, Germany |
| Malte Schilling | International Computer Science Institute, Berkeley, USA |
| Ute Schmid | University of Bamberg, Germany |
| Lutz Schröder | DFKI Bremen and University of Bremen, Germany |
| Carsten Schuermann | IT University of Copenhagen, Denmark |
| René Schumann | National Institute of Informatics, Japan |
| Jan-Georg Smaus | University of Freiburg, Germany |
| Luciano Spinello | University of Freiburg, Germany |
| Steffen Staab | University of Koblenz-Landau, Germany |
| Cyrill Stachniss | University of Freiburg, Germany |
| Rainer Stiefelhagen | Karlsruhe Institute of Technology, Germany |
| Ingo J. Timm | University of Trier, Germany |
| Rudolph Triebel | University of Oxford, UK |
| Toby Walsh | ICTA and UNSW, Australia |
| Thomas Wiemann | University of Osnabrück, Germany |
| Dirk Wollherr | Munich University of Technology, Germany |
| Diedrich Wolter | University of Bremen, Germany |
| Stefan Wölfle | University of Freiburg, Germany |

Additional Reviewers

| | | |
|-----------------------|----------------------|------------------------|
| Alhalah, Ziad | IJsselmuiden, Joris | Ruß, Georg |
| Althoff, Daniel | Janssen, Frederik | Ryll, Markus |
| Bach, Kerstin | Jung, Jean Christoph | Schauerte, Boris |
| Banerjee, Amit | Maier, Daniel | Schiffer, Stefan |
| Bengtson, Jesper | Martiny, Karsten | Stanczyk, Bartłomiej |
| Besold, Tarek Richard | Maye, Jerome | Steinbrecher, Matthias |
| Bienvenu, Meghyn | Moewes, Christian | Stommel, Martin |
| Breidt, Martin | Müller, Steffen | Storandt, Sabine |
| Eisner, Jochen | Newo, Rgis | Stricker, Ronny |
| Elfers, Carsten | Oezcep, Oezguer | Szedmak, Sandor |
| Eyerich, Patrick | Paris, Jeff | Volkhardt, Michael |
| Ferrein, Alexander | Passenberg, Benjamin | Wandelt, Sebastian |
| Goncharov, Sergey | Paulheim, Heiko | Ziegelmann, Mark |
| Held, Pascal | Pfeifer, Niki | |
| Hornung, Armin | Rohrmüller, Florian | |

Table of Contents

| | |
|---|-----|
| Everything You Always Wanted to Know about <i>Planning</i> (But Were Afraid to Ask) | 1 |
| <i>Jörg Hoffmann</i> | |
| Why We Need Evolutionary Semantics | 14 |
| <i>Luc Steels</i> | |
| General Game Playing in AI Research and Education | 26 |
| <i>Michael Thielscher</i> | |
| Conversational Agents in a Virtual World | 38 |
| <i>Peter Adolphs, Anton Benz, Núria Bertomeu Castelló, Xiwen Cheng, Tina Klüwer, Manfred Krifka, Alexandra Strekalova, Hans Uszkoreit, and Feiyu Xu</i> | |
| Dependency Graphs as a Generic Interface between Parsers and Relation Extraction Rule Learning | 50 |
| <i>Peter Adolphs, Feiyu Xu, Hong Li, and Hans Uszkoreit</i> | |
| Evaluation and Comparison Criteria for Approaches to Probabilistic Relational Knowledge Representation | 63 |
| <i>Christoph Beierle, Marc Finthammer, Gabriele Kern-Isbner, and Matthias Thimm</i> | |
| Segmentation of Action Streams Human Observers vs. Bayesian Binning | 75 |
| <i>Dominik Endres, Andrea Christensen, Lars Omlor, and Martin A. Giese</i> | |
| Speedy Local Search for Semi-Supervised Regularized Least-Squares | 87 |
| <i>Fabian Gieseke, Oliver Kramer, Antti Airola, and Tapio Pahikkala</i> | |
| Model-Based Object Recognition from 3D Laser Data | 99 |
| <i>Martin Günther, Thomas Wiemann, Sven Albrecht, and Joachim Hertzberg</i> | |
| Swarm Intelligence for Medical Volume Segmentation: The Contribution of Self-reproduction | 111 |
| <i>Robert Haase, Hans-Joachim Böhme, Daniel Zips, and Nasreddin Abolmaali</i> | |
| Efficient Sequential Clamping for Lifted Message Passing | 122 |
| <i>Fabian Hadji, Babak Ahmadi, and Kristian Kersting</i> | |

| | |
|---|-----|
| BetterRelations: Using a Game to Rate Linked Data Triples | 134 |
| <i>Jörn Hees, Thomas Roth-Berghofer, Ralf Biedert, Benjamin Adrian, and Andreas Dengel</i> | |
| Generic Performance Metrics for Continuous Activity Recognition | 139 |
| <i>Albert Hein and Thomas Kirste</i> | |
| Bayesian Logic Networks and the Search for Samples with Backward Simulation and Abstract Constraint Learning | 144 |
| <i>Dominik Jain, Klaus von Gleissenthall, and Michael Beetz</i> | |
| Transformation Rules for First-Order Probabilistic Conditional Logic Yielding Parametric Uniformity | 157 |
| <i>Ruth Janning and Christoph Beierle</i> | |
| Variance Scaling for EDAs Revisited | 169 |
| <i>Oliver Kramer and Fabian Gieseke</i> | |
| Hierarchically Structured Energy Markets as Novel Smart Grid Control Approach | 179 |
| <i>Jörg Lässig, Benjamin Satzger, and Oliver Kramer</i> | |
| Compiling AI Engineering Models for Probabilistic Inference | 191 |
| <i>Paul Maier, Dominik Jain, and Martin Sachenbacher</i> | |
| Smooth Conditional Transition Paths in Dynamical Gaussian Networks | 204 |
| <i>Michał Matuszak, Jacek Miękisz, and Tomasz Schreiber</i> | |
| HTN-Style Planning in Relational POMDPs Using First-Order FSCs ... | 216 |
| <i>Felix Müller and Susanne Biundo</i> | |
| Gates for Handling Occlusion in Bayesian Models of Images: An Initial Study | 228 |
| <i>Daniel Oberhoff, Dominik Endres, Martin A. Giese, and Marina Kolesnik</i> | |
| TGA-Based Controllers for Flexible Plan Execution | 233 |
| <i>Andrea Orlandini, Alberto Finzi, Amedeo Cesta, and Simone Fratini</i> | |
| A Metric to Evaluate a Cluster by Eliminating Effect of Complement Cluster | 246 |
| <i>Hamid Parvin, Behrouz Minaei, and Sajad Parvin</i> | |
| Predicting Numbers: An AI Approach to Solving Number Series | 255 |
| <i>Marco Ragni and Andreas Klein</i> | |
| Prediction of Classifier Training Time Including Parameter Optimization | 260 |
| <i>Matthias Reif, Faisal Shafait, and Andreas Dengel</i> | |

| | |
|---|-----|
| Human-Machine Corpus Analysis for Generation and Interaction with Spoken Dialog Systems | 272 |
| <i>Roland Roller, Tatjana Scheffler, and Norbert Reithinger</i> | |
| Comparison of Laser-Based Person Tracking at Feet and Upper-Body Height | 277 |
| <i>Konrad Schenk, Markus Eisenbach, Alexander Kolarow, and Horst-Michael Gross</i> | |
| Refinements of Restricted Higher-Order Anti-Unification for Heuristic-Driven Theory Projection | 289 |
| <i>Martin Schmidt, Helmar Gust, Kai-Uwe Kühnberger, and Ulf Krumnack</i> | |
| Linkless Normal Form for <i>ALC</i> Concepts and TBoxes | 301 |
| <i>Claudia Schon</i> | |
| Shape Retrieval with Qualitative Relations: The Influence of Part-Order and Approximation Precision on Retrieval Performance and Computational Effort | 313 |
| <i>Arne Schuldt</i> | |
| Classification of Semantic Concepts to Support the Analysis of the Inter-cultural Visual Repertoires of TV News Reviews | 325 |
| <i>Martin Stommel, Martina Duemcke, and Otthein Herzog</i> | |
| Shaking Hands in Latent Space: Modeling Emotional Interactions with Gaussian Process Latent Variable Models | 330 |
| <i>Nick Taubert, Dominik Endres, Andrea Christensen, and Martin A. Giese</i> | |
| Value-Difference Based Exploration: Adaptive Control between Epsilon-Greedy and Softmax | 335 |
| <i>Michel Tokic and Günther Palm</i> | |
| Calculating Meeting Points for Multi User Pedestrian Navigation Systems | 347 |
| <i>Bjoern Zenker and Alexander Muench</i> | |
| Algorithmic Debugging to Support Cognitive Diagnosis in Tutoring Systems | 357 |
| <i>Claus Zinn</i> | |
| Author Index | 369 |

Everything You Always Wanted to Know about *Planning*

(But Were Afraid to Ask)

Jörg Hoffmann

INRIA, Nancy, France
joerg.hoffmann@inria.fr

Abstract. Domain-independent planning is one of the long-standing sub-areas of Artificial Intelligence (AI), aiming at approaching human problem-solving flexibility. The area has long had an affinity towards playful illustrative examples, imprinting it on the mind of many a student as an area concerned with the rearrangement of blocks, and with the order in which to put on socks and shoes (not to mention the disposal of bombs in toilets). Working on the assumption that this “student” is you – the readers in earlier stages of their careers – I herein aim to answer three questions that you surely desired to ask back then already: *What is it good for? Does it work? Is it interesting to do research in?* Answering the latter two questions in the affirmative (of course!), I outline some of the major developments of the last decade, revolutionizing the ability of planning to scale up, and the understanding of the enabling technology. Answering the first question, I point out that modern planning proves to be quite useful for solving practical problems - including, perhaps, yours.

Disclaimer. *This exposition is but a little teaser to stimulate your appetite. It's far from a comprehensive summary of the field. The choice of topics and literature is a willful sample according to my personal interests, and of course it over-represents my own contribution. The language and style are sloppy. On the positive side, the paper is entertaining and easy to read (or so I hope).*

1 Planning? What's that?

Planning is the problem of selecting a goal-leading course of actions based on a high-level description of the world. One could fill books (and that's what people have done [2]) with the different variants of what exactly this means. Herein, we will make do with the most canonical definition:

Definition 1. A *planning task* is a 4-tuple $\Pi = (\mathcal{V}, \mathcal{A}, s_0, s_*)$ where:

- (i) $\mathcal{V} = \{v_1, \dots, v_n\}$ is a set of finite-domain **state variables**.
- (ii) \mathcal{A} is a set of **actions** a , where each a is a pair $(\text{pre}_a, \text{eff}_a)$ of partial variable assignments called **preconditions** and **effects**.
- (iii) s_0 is a complete variable assignment called the **initial state**, and s_* is a partial variable assignment called the **goal**.

I omit the straightforward formal semantics associated with this syntax. Suffice it to say that the task is associated with its **state space**, the directed graph of all **states** (complete variable assignments), with an arc from s to s' iff there exists $a \in \mathcal{A}$ so that pre_a complies with s , and changing s according to eff_a yields s' (note that eff_a over-writes previous variable values; we don't distinguish "adds" and "deletes"). A **plan** is a path leading from s_0 to a state complying with s_* . The plan is **optimal** if it is a shortest such path.

From a computer science perspective, planning is just one formalism succinctly describing large transition systems, similar to automata networks or Turing machines. Trivially, planning is hard (**PSPACE**-complete in our case here). What makes planning special is its purpose in AI. We'll get to that in a moment. A particularly special aspect of planning are the illustrative examples (and benchmarks!) that have long dominated the field. Fig. 1 gives two of the most emblematic scenarios students are confronted with when first hearing of this beautiful area.

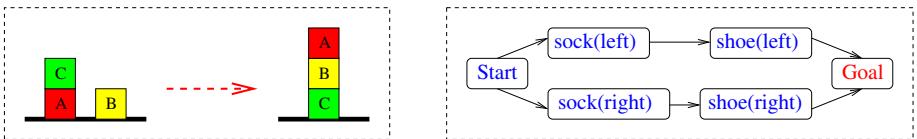


Fig. 1. This is planning (?)

Fig. 1 (left) actually is the only scientific object in the planning area that has ever been adorned with its inventor's name ("The Sussman Anomaly"). Isn't that depressing? I wouldn't insist on having a "Hoffmann's Theorem", but surely *someone* did more interesting planning stuff than that, sometime. Anyway, Fig. 1 (right) is perhaps even more counterproductive. Few people will appreciate the importance of AI technology in putting on their socks and shoes. As for the widely used benchmark called "Bomb in the toilet" (inner workings omitted for the sake of sanity), I guess the best that can be said about it is that it is not associated with an emblematic illustration. So, without any further ado:

2 What is it good for?

Back in the day of GOFAI¹, planning got started having in mind to approach human problem solving flexibility. One may argue whether or not, to accomplish this, it makes sense to assume mathematically precise world models as the planning input. But, in the spirit of the 21st century, let's just forget philosophy and be rock-bottom pragmatic: How can we earn money with planning?

What characterizes planning research is the attempt to create *one* planning solver that will perform sufficiently well on *all* possible domains (inputs). That will never work out (the problem is hard), but there's been tremendous algorithmic progress in the last decade. This will pay off for you if either:

¹ "Good Old-Fashioned AI", cf. <http://en.wikipedia.org/wiki/GOFAI>

- (A) **Your problem is subject to frequent change.** If you implement your own solver, you'll have to keep adapting it. Using planning, it suffices to change the declarative planning model.
- (B) **It would be costly to implement your own solver.** Unless your problem is quite easy, making a solver will cost time+money. Writing the planning model is typically much less effort.

In other words, *planning is a cost-effective method for software engineering*. It's a model-based approach. The planning model serves as a high-level programming language decoupling the problem from its solution.

Scenario (A) is a classical argument made in favor of planning, and has long been the reason for its investigation in the context of space travel, where “adapting the solver” can be problematic due to circumstance (watched Apollo 13, anyone?). In the meantime, industrial applications exist also down on this earth. For example, at Xerox a planning architecture is successfully being employed to flexibly control highly configurable printing systems [36]. At SAP, planning fits seamlessly into a major model-driven software engineering effort aimed at effective change management [24].

Scenario (B) is a tad unconventional, but is quite real and may indeed be the “killer app” for planning technology: *planning is a quick hack to get things up and running*. Rapid prototyping, in other words. Planning people don't usually think of themselves in these terms, and this point was driven home to me only quite recently, in a conversation with Alexander Koller who has been using planning for natural language sentence generation [27,28]. When I asked him why he doesn't develop a specific solver that could be more effective – a typical if not emblematic planning question – his answer was: “well, that's not the research problem I'm interested in; the planner works reasonably well, and I don't want to spend the time working out an alternative”.

Shortly afterward, I actually found myself being better friends with the development department than with the research department of a company I was working with. “If pigs could fly”, you might think right now; but it's true. The respective punchlines were “oh, but we could come up with something that works much better than the planner” (research department) vs. “yeah, maybe, but you don't know when and my product deadline is next month” (development department). The company – Core Security Technologies, <http://www.coresecurity.com/> – now employs a variant of my Metric-FF planner [19] in their product, serving to intelligently select possible attacks in regular security checks against a client network [29]. Note that, both for Alexander and Core Security Technologies, the “quick hack” actually turned into a long-term solution!

Generality is of course not a unique virtue of planning. SAT and CP, for example, are competing model-based approaches. Planning has potential advantages in modeling, since planning models are very high-level and can thus be more human-readable and easier to modify. In terms of solving, the approaches are complementary. Generally speaking, one can expect constraint-based methods to have the edge in combinatorial optimization. But for obtaining reasonable

solutions quickly, in particular in applications where finding a feasible solution is already hard, a planner might be the better choice. Let me outline why.

3 Does it work?

The curse of planning is dimensionality, aka the state explosion problem. The major news from the last decade is that we now have techniques able to tackle this problem fairly well, judging at least from performance in an ever-growing set of established benchmarks from the International Planning Competition. We now have some 40 domains and well over 1000 instances, all encoded in the same common language PDDL [31][10].

Like SAT, planning experienced a major scalability breakthrough in the last decade. Where in SAT this is largely thanks to clause learning techniques, in planning it's largely thanks to heuristic search. The reader who has followed the planning literature just a little bit will be familiar with this development. Let me say a few words for the benefit of the reader that didn't; everybody else may skip to below Fig. 2. A major player here is this simple definition:²

Definition 2. Let $\Pi = (\mathcal{V}, \mathcal{A}, s_0, s_*)$ be a planning task. An action sequence $\langle a_1, \dots, a_n \rangle$ is a **relaxed plan** for Π iff, with $s_0^+ = \{(v, c) \mid s_0(v) = c\}$ and $s_i^+ = s_{i-1}^+ \cup \text{eff}_{a_i}$, we have that $s_* \subseteq s_n^+$ and that $\text{pre}_{a_i} \subseteq s_{i-1}^+$ for $1 \leq i \leq n$. The relaxed plan is **optimal** if n is minimal; the **relaxed plan heuristic** is then $h^+(\Pi) = n$.

Many planners use this kind of heuristic in a forward state space search, where for every state s visited during search on planning task $\Pi = (\mathcal{V}, \mathcal{A}, s_0, s_*)$, the planner evaluates the relaxed plan heuristic for the “local” task $\Pi_s = (\mathcal{V}, \mathcal{A}, s, s_*)$. Hence a relaxed planning task is solved in every individual search state.

The relaxed plan heuristic assumes that, whenever a variable changes its value, it obtains the new value *and* retains the old one. In other words, we never overwrite previous values; what we achieved once will remain true forever. That is, of course, not the case in the real world, or in any planning application known to man. Just to illustrate, the relaxed plan heuristic for the n -discs Towers-of-Hanoi problem is n , under-estimating the actual plan length by an exponential number of steps because it effectively ignores all the interactions between different disc moves. Despite this, it turns out that h^+ delivers excellent search guidance for many planning domains.

We'll get into a little more detail on this below (Section 4.2); to give an intuition why h^+ could be informative, perhaps a few examples are helpful. If the planning domain is graph distance – finding a shortest path in a graph – then h^+ is exact (because shortest paths “never walk back”). The same goes for (discrete) path finding in the presence of obstacles. In the sliding-tiles puzzle, h^+ strictly dominates Manhattan distance. In TSP, relaxed plans are equivalent to

² In a logics-based representation where action effects are positive or negative, this definition is equivalent to ignoring the negative effects.

the minimum spanning tree approximation. If a planning task contains some of these kinds of structure, then these reasonable approximations will be captured.

Obviously, h^+ is lower-bounding (admissible) and consistent. It first appeared in the literature in 1994, as a footnote of an investigation of planning tractability [6] (describing it as an uninteresting sub-class). Its use for search guidance was first proposed 2 years later [305], and was proliferated during the last decade to become *the* most successful technique for finding plans effectively.

Computing h^+ itself is actually hard, but upper bounds can be computed easily [2][23], essentially by generating *some* (not necessarily optimal) relaxed plan. These bounds tend to be close to h^+ in many benchmarks [20], but do not provide any theoretical guarantees so these heuristic functions are in general not admissible. What we can use them for is satisficing (greedy) search for plans. They also provide us with natural action pruning techniques, simply by giving a preference to those actions that are used by the relaxed plan [23][32].

Virtually every winner of the satisficing tracks at the planning competitions since the year 2000 makes use of the described techniques in one way or another, within one or another instance of heuristic search (e.g., [23][11][33]). To give an impression of the scale of the performance boost, Fig. 2 compares the state of the art prior to the year 2000 – which was based essentially on Graphplan [1] – to my FF planner [23] that dominated the 2000 competition.

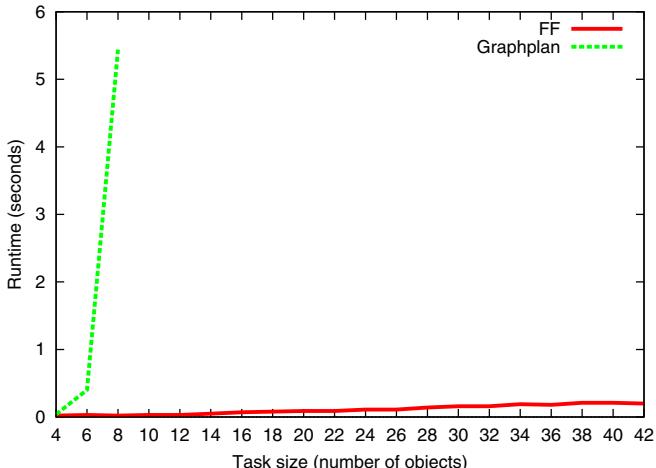


Fig. 2. Good-bye, Graphplan!

The benchmark domain underlying Fig. 2 is extremely simple (robot transports n objects from A to B), but does not exaggerate the point. Whereas previously our best planners exploded with tiny size parameters already, we could suddenly scale up to instances many many times larger, even obtain near-linear runtime behavior as in Fig. 2. Current heuristic search planners comfortably

solve instances with astronomically large state spaces (10^{100} is not unheard of), so long as their search guidance mechanisms point them “the right direction”³.

What does this mean for the comparison to the competing model-based approaches – SAT and CP – I mentioned above? *Should you personally consider to use planning to solve your problem? Or are you better off with something else?*

I certainly wouldn’t claim to have a definite answer, but let me try to give a guideline. As outlined, the effective planners mentioned above do not give any hard guarantees on solution quality. So if a guarantee is essential to you, don’t go there. Otherwise, if your problem is naturally formulated in terms of planning, then it is definitely worth a try. Which approach will be more effective computationally depends on “the kind of structure” your problem has. Constraint reasoning is probably better at playing through tightly interconnected options, like solving a puzzle, finding a clever schedule, or finding a complicated erroneous behavior in Model Checking. But such reasoning will be hopeless in a domain more like the one from Fig. 2, where guided greediness is key to finding large feasible solutions. As an intuitive if imprecise picture, a DPLL solver may spend ages determining that n pigeons need at least n holes; heuristic planners will just go ahead and assign them.

Not wanting this paper to drag on forever, I’ll only briefly touch upon two additional recent and important developments. First, optimal planning (with additive action costs functions) has undergone a major boost in the last years, mainly thanks to a new admissible heuristic called LM-cut [14] that under-estimates h^+ . Combining admissible with non-admissible estimators yields powerful bounded-suboptimal search algorithms (see e.g. [33]) so, contradicting what I said above, this may be your way to go if quality guarantees are of essence.

A very recent development is that SAT-based planning – which has been more in the Graphplan ballpark up to now – seems to be about to catch up to heuristic search. Dramatic speed-ups are obtained through a combination of clever encodings, SAT call scheduling, and planning-specific branching heuristics [35,34]. In Jussi Rintanen’s experiments, these planners manage to solve as many, and more, benchmark instances as the best heuristic search planners. They weren’t as competitive in the 2011 planning competition, though. But certainly there is the potential to form another powerful tool for applying planning⁴.

Summing up, planning has made dramatic scalability progress, and might well be useful for solving some of your problems (whatever they are). So perhaps you’ll reconsider the impression you got as a student when looking at (your equivalent of) Fig. 1, and you’ll ask yourself:

³ In difference to FF, Graphplan gives an optimality guarantee. But it’s a useless one (an algorithm artifact rather than a realistic optimization criterion). Anyhow, after having to wait for a couple of millennia, presumably the plan will be kinda useless.

⁴ If you’re curious about the self-contradiction with respect to the discussion of SAT/CP above: (a) the world is too complicated for a 12-page LNCS-style paper; (b) planning-specific greediness in the branching heuristic is a key element of these new SAT planners. Thus perhaps, to some extent, they are able to combine the virtues of heuristic search planning with those of constraint reasoning.

4 Is it interesting to do research in?

Why, of course it is!

The malicious reader might be tempted at this point to put forward arguments of a psychological nature, connecting this statement and the fact that I've been working in planning for about 15 years now. I'd be happy to meet in a bar sometime to discuss this, but perhaps I can convince you here already.

One thing that makes planning interesting is the wide open space of promising algorithmic possibilities. Whereas in SAT, DPLL has been ruling the house since ages and people are busy finding new ways to push around the bits in unit propagation⁵, planning has always been characterized by a profusion of very different algorithms. The reader might argue now that (a) heuristic search has been ruling the house in the planning competition since 10 years, which (b) is just as boring as DPLL. I concede (a), except for reminding the reader of the aforementioned modern competitive SAT-based planners [35,34]. But I beg to differ on (b) – it's not as boring as DPLL.

There is a wide open algorithmic space as to *how to automatically generate heuristic functions*. Arguably, this question boils down to “how to abstract the problem” since heuristic estimates are universally generated by considering some simplification (“abstraction”/“relaxation”) of the planning task at hand. But the land of abstractions is indeed one of unlimited possibilities. That is particularly true of planning in difference to, e.g., Verification where abstractions also are paramount. In Verification, an abstraction must, as far as the to-be-verified property is concerned, be identical to the original system. In heuristic search planning, we can abstract in whichever way we want, and as much as we want, and we will still obtain potentially useful estimates.

Is, then, the life of a researcher in heuristic search planning characterized by the following pseudo-code?

```

while ( not retired ) do
    think up some new heuristic  $h^{\text{foo-bar}}$ 
    run it on the benchmarks
endwhile

```

Fig. 3. The life of a planning researcher?

The answer to that one is “NO!”. Far beyond just improving performance on benchmarks, the *understanding* of heuristics is where heuristic search planning really turns into a natural science. Dramatic progress has been made, in that science, during the last years. For example, Bonet and Geffner [3] proved (and exploited) connections between h^+ , logic programming, and knowledge

⁵ My sincere apologies to colleagues from the SAT area for this polemical formulation.

The comparison to SAT in the present context is owed to the fact that everybody (including myself) thinks that SAT is really cool.

compilation; Katz and Domshlak [26] proved that optimal cost-partitionings⁶ can be computed in polynomial time; Bonet and Helmert [4] proved that h^+ is equivalent to maximal hitting sets over action landmarks (sets of actions at least one of which takes part in every plan). Let me give just a little bit more detail on two other recent results, addressing what are probably the two most fundamental questions when trying to understand heuristics.

4.1 How do heuristics relate to each other?

An arsenal of heuristic functions is, naturally, not merely a list h_1, \dots, h_N of names and associated methods. There are algorithmic relationships. In planning, specifically, the heuristics proposed so far can be classified as: *critical-paths heuristics*, which relate to Graphplan [13]; *relaxation heuristics* relating to h^+ [2][23]; *abstraction heuristics*, like pattern databases, homomorphically mapping the state space to a smaller abstract state space [9][15]; and *landmark heuristics*, based on counting yet un-achieved landmarks [33][25]. The relationships implied by this classification are obvious, and most of them are simply the historical effect of people building on each other's work. However, Helmert and Domshlak [14] recently showed how we can make connections *across* these classes, uncovering for example that certain apparently unrelated heuristics are, in fact, equivalent.

Helmert and Domshlak [14] introduce a *compilation framework for classes of heuristics*. Given two classes A and B of admissible heuristics (e.g., A =critical-paths heuristics vs. B =landmark heuristics), A can be compiled into B if, for any given state s and heuristic $h^A \in A$, one can in time polynomial in the size of the planning task construct a heuristic $h^B \in B$ so that $h^A(s) \leq h^B(s)$. That is,

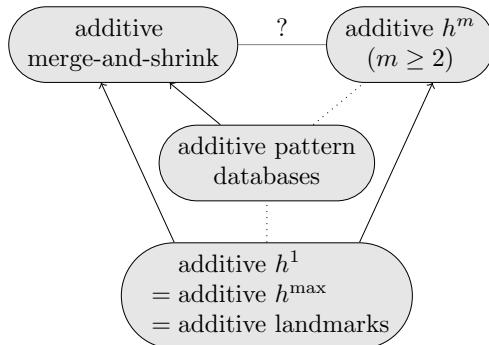


Fig. 4. Helmert and Domshlak's [14] compilation results. Arrows indicate compilability, dotted lines indicate that compilation is not possible. Merge-and-shrink heuristics cannot be compiled into h^m heuristics; the converse question is open.

⁶ Given an ensemble of abstractions, a cost partitioning is a function that distributes the cost of each action across these abstractions, so that the sum of all abstraction heuristics is still a lower bound. The cost partitioning is optimal in a given state s if that sum, in s , is maximal.

we can always at least simulate A through B , with only polynomial overhead. Fig. 4 provides their results overview for illustration.

Additivity in Fig. 4 refers to the use of heuristic ensembles and cost partitionings. The h^+ heuristic is not included because it cannot be compilable into any heuristic (else we could compute h^+ in polynomial time). Major discoveries here are that landmarks are incomparable with pattern databases but can be compiled into merge-and-shrink (a generalization of pattern databases), and that landmarks are equivalent to h^{max} which is a relaxation heuristic. The latter result is easy for compilation into h^{max} , but is quite non-trivial for the other direction. The proof construction – constructing landmarks in a way so that the eventual heuristic value will at least dominate h^{max} – has been implemented. In fact, it’s the aforementioned LM-cut heuristic that has revolutionized the scalability of optimal planning!

4.2 Under what circumstances does a given heuristic work well?

Any heuristic gives good search guidance only on some sub-class of planning tasks. What we would like to understand is what that class is. Ideally, we would like a machine to implement this “understanding”, so that the planning technology could test automatically whether or not any one heuristic is likely to work well. In other words, we want a fortune-teller for search performance. Anybody experienced in search knows that this is so difficult one can just as well look into a crystal ball. Surprisingly, it turns out that for the h^+ heuristic – the most influential heuristic by far in planning – a suitable crystal ball actually exists.

The “crystal ball” is called TorchLight [16,22]. Its history started in the year 2000 when I tried to understand, manually and on a per-benchmark-domain basis, where and in what ways h^+ is informative. The final outcome of that was a “planning benchmark domain taxonomy”, dividing them into classes differing with respect to the topology of the search space surface under h^+ [17,18,21]. Most strikingly, in many of the domains, *no local minima exist at all*.

Thus a very basic crystal ball should be able to divine whether or not there are local minima under h^+ . In 2001, having attempted this in vain for several months, I gave up. In 2009 – while explaining to someone why this is never gonna work – I finally realized that *causal graphs* can do the trick. The vertices in this graph are the state variables, and there is an arc (x, y) iff moving y sometimes involves a condition on x . As it turns out, if the causal graph is acyclic and all variable transitions are invertible, then there are no local minima under h^+ . This sufficient condition is easily testable, and can be significantly generalized. Voilà our crystal ball! Fig. 5 overviews its performance.

The table structure of Fig. 5 corresponds to my planning domain taxonomy. Leaving out the details, a domain’s topology is the “easier” the nearer it is to the bottom left; domains without local minima are (highlighted in blue and) marked with a “*”. The numbers in brackets give TorchLight’s estimation of the domain’s difficulty, namely the fraction of sampled states proved to not be on a local minimum. Thus 0=“very hard”, 100=“very easy”. This estimate is, of course, not perfect. But it correlates well with planner performance [22], and is

| | | | |
|--|--|------------------------------|---|
| BlocksArm [30] Depots [82] Driverlog [100] | Pipes–Tank [40] Pipes–NoTank [76] PSR [50] | Rovers [100] Opt–Tele [7] | Mystery [39] Mprime [49] Freecell [55] Airport [0] |
| Hanoi* [0] BlocksNoArm* [57] Transport* [100] | Grid* [80] | | |
| Elevators* [100] Logistics* [100] Ferry* [100] Gripper* [100] | Tyeworld* [100] Satellite [100] Zenotravel [95] Miconic–STR* [100] Movie* [100] Simple–Tsp* [100] | Din–Phil [24] | |

Fig. 5. Overview of TorchLight domain analysis results

indeed more refined than my own hand-made analysis. All I could give you is a “local minima exist? yes/no”. So, don’t ask me, ask TorchLight!

5 And now, what?

I was kinda hoping that at this point you might feel tempted to play around a little bit with an actual planner. For your convenience, I put a little starter package at <http://www.loria.fr/~hoffmanj/PlanningForDummies.zip>. It contains the FF source code as well as 3 simple benchmark domains with instance generators. Linux executables and some example planning tasks are included. I was so nice to also add a README file. If you get seriously interested, you should have a look through the planning competition web pages <http://ipc.icsaps-conference.org/>, and through the page of Fast Downward <http://www.fast-downward.org/> which is quickly becoming the main implementation basis for new planning techniques. Do send me email if you have questions.

Scientifically, I’d like to close this paper by striking a blow for research on supporting *modeling* for planning. This is an active research area. The planning competition has a separate track of events for knowledge engineering (latest edition: <http://kti.mff.cuni.cz/~bartak/ICKEPS2009/>). Several research groups are developing modeling environments, e.g. itSIMPLE [39] <http://code.google.com/p/itsimple/> and GIPO [37] <http://scom.hud.ac.uk/planform/gipo/>. Learning techniques support the automatic extraction of domain models from various kinds of data [87]. Still, in my humble opinion, this issue is not nearly given sufficient attention. The vast majority of planning researchers are concerned with what’s going on inside their planners, and worry little (at all?) about where these planners will actually get their PDDL input from – in practice, not in the planning competition!

I was no different until quite recently. Then I worked on 3 different applications, with Alexander Koller [27], at SAP [24], with Core Security Technologies [29]. In each and every one of these, it was easy to obtain good planning performance by simple modifications of FF. The real issue was designing the PDDL.

Contrary to common lore, planning is not “automatic”. Yes, it suffices to describe the domain. But that is not a push-button operation.

Remember what I pointed out previously: *planning is a quick hack to get things up and running*. I believe that our technology has great potential as a method for rapid prototyping. We are very far indeed from fully exploiting that potential. Search algorithms like branch-and-bound are widely known, and any practitioner (with a CS background) having a search problem to solve is likely to go ahead and build on those. People simply don’t know that, perhaps, they could just describe their problem to a planner and be done. We need to get this knowledge out there, and we need to provide users with the tools needed to conveniently access our technology. Advancing the outreach of planning in this way is our major challenge for the coming decade – apart, of course, from keeping ourselves happy by proving interesting theorems about heuristic functions.

Acknowledgments. I would like to thank Stefan Edelkamp for inviting me to write this paper. I am grateful to all my colleagues from the planning community, and I thank them specifically for their work as described and cited herein. I hope we’ll still be friends after you read this.

References

1. Blum, A.L., Furst, M.L.: Fast planning through planning graph analysis. *Artificial Intelligence* 90(1-2), 279–298 (1997)
2. Bonet, B., Geffner, H.: Planning as heuristic search. *Artificial Intelligence* 129(1–2), 5–33 (2001)
3. Bonet, B., Geffner, H.: Heuristics for planning with penalties and rewards formulated in logic and computed through circuits. *Artificial Intelligence* 172(12-13), 1579–1604 (2008)
4. Bonet, B., Helmert, M.: Strengthening landmark heuristics via hitting sets. In: *Proceedings of the 19th European Conference on Artificial Intelligence* (2010)
5. Bonet, B., Loerincs, G., Geffner, H.: A robust and fast action selection mechanism for planning. In: *Proceedings of the 14th National Conference of the American Association for Artificial Intelligence* (1997)
6. Bylander, T.: The computational complexity of propositional STRIPS planning. *Artificial Intelligence* 69(1–2), 165–204 (1994)
7. Castillo, L.A., Morales, L., González-Ferrer, A., Fernández-Olivares, J., Borrajo, D., Onaindia, E.: Automatic generation of temporal planning domains for e-learning problems. *Journal of Scheduling* 13(4), 347–362 (2010)
8. Cresswell, S., McCluskey, T.L., West, M.M.: Acquisition of object-centred domain models from planning examples. In: *Proceedings of the 19th International Conference on Automated Planning and Scheduling* (2009)
9. Edelkamp, S.: Planning with pattern databases. In: *Recent Advances in AI Planning. 6th European Conference on Planning* (2001)
10. Fox, M., Long, D.: PDDL2.1: An extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research* 20, 61–124 (2003)
11. Gerevini, A., Saetti, A., Serina, I.: Planning through stochastic local search and temporal action graphs. *Journal of Artificial Intelligence Research* 20, 239–290 (2003)

12. Ghallab, M., Nau, D., Traverso, P.: *Automated Planning: Theory and Practice*. Morgan Kaufmann, San Francisco (2004)
13. Haslum, P., Geffner, H.: Admissible heuristics for optimal planning. In: Proceedings of the 5th International Conference on Artificial Intelligence Planning Systems (2000)
14. Helmert, M., Domshlak, C.: Landmarks, critical paths and abstractions: What's the difference anyway? In: Proceedings of the 19th International Conference on Automated Planning and Scheduling (2009)
15. Helmert, M., Haslum, P., Hoffmann, J.: Flexible abstraction heuristics for optimal sequential planning. In: Proceedings of the 17th International Conference on Automated Planning and Scheduling (2007)
16. Hoffmann, J.: Where ignoring delete lists works, part II: Causal graphs. In: Proceedings of the 21st International Conference on Automated Planning and Scheduling (2011)
17. Hoffmann, J.: Local search topology in planning benchmarks: An empirical analysis. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence (2001)
18. Hoffmann, J.: Local search topology in planning benchmarks: A theoretical analysis. In: Proceedings of the 6th International Conference on Artificial Intelligence Planning and Scheduling (2002)
19. Hoffmann, J.: The Metric-FF planning system: Translating "ignoring delete lists" to numeric state variables. *Journal of Artificial Intelligence Research* 20, 291–341 (2003)
20. Hoffmann, J.: Utilizing Problem Structure in Planning: A Local Search Approach. LNCS (LNAI), vol. 2854. Springer, Heidelberg (2003)
21. Hoffmann, J.: Where 'ignoring delete lists' works: Local search topology in planning benchmarks. *Journal of Artificial Intelligence Research* 24, 685–758 (2005)
22. Hoffmann, J.: Analyzing search topology without running any search: On the connection between causal graphs and h^+ . *Journal of Artificial Intelligence Research* 41, 155–229 (2011)
23. Hoffmann, J., Nebel, B.: The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research* 14, 253–302 (2001)
24. Hoffmann, J., Weber, I., Kraft, F.M.: SAP speaks PDDL. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence (2010)
25. Karpas, E., Domshlak, C.: Cost-optimal planning with landmarks. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence (2009)
26. Katz, M., Domshlak, C.: Optimal additive composition of abstraction-based admissible heuristics. In: Proceedings of the 18th International Conference on Automated Planning and Scheduling (2008)
27. Koller, A., Hoffmann, J.: Waking up a sleeping rabbit: On natural-language sentence generation with FF. In: Proceedings of the 20th International Conference on Automated Planning and Scheduling (2010)
28. Koller, A., Petrick, R.: Experiences with planning for natural language generation. *Computational Intelligence* 27(1), 23–40 (2011)
29. Lucangeli, J., Sarraute, C., Richarte, G.: Attack planning in the real world. In: Proceedings of the 2nd Workshop on Intelligent Security (2010)
30. McDermott, D.: A heuristic estimator for means-ends analysis in planning. In: Proceedings of the 3rd International Conference on Artificial Intelligence Planning Systems (1996)
31. McDermott, D., et al.: The PDDL Planning Domain Definition Language. In: The AIPS 1998 Planning Competition Committee (1998)

32. Richter, S., Helmert, M.: Preferred operators and deferred evaluation in satisficing planning. In: Proceedings of the 19th International Conference on Automated Planning and Scheduling (2009)
33. Richter, S., Westphal, M.: The LAMA planner: Guiding cost-based anytime planning with landmarks. *Journal of Artificial Intelligence Research* 39, 127–177 (2010)
34. Rintanen, J.: Heuristics for planning with SAT. In: Cohen, D. (ed.) CP 2010. LNCS, vol. 6308, pp. 414–428. Springer, Heidelberg (2010)
35. Rintanen, J., Heljanko, K., Niemelä, I.: Planning as satisfiability: parallel plans and algorithms for plan search. *Artificial Intelligence* 170(12-13), 1031–1080 (2006)
36. Ruml, W., Do, M.B., Zhou, R., Fromherz, M.P.J.: On-line planning and scheduling: An application to controlling modular printers. *Journal of Artificial Intelligence Research* 40, 415–468 (2011)
37. Simpson, R.M., Kitchin, D.E., McCluskey, T.L.: Planning domain definition using GIPO. *Knowledge Engineering Review* 22(2), 117–134 (2007)
38. Thayer, J., Ruml, W.: Bounded suboptimal search: A direct approach using inadmissible estimates. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (2011)
39. Vaquero, T.S., Romero, V., Tonidandel, F., Silva, J.R.: itSIMPLE 2.0: an integrated tool for designing planning domains. In: Proceedings of the 17th International Conference on Automated Planning and Scheduling (2007)

Why We Need Evolutionary Semantics

Luc Steels^{1,2}

¹ ICREA, IBE(UPF-CSIC) - Barcelona

² Sony Computer Science Laboratory - Paris

steels@arti.vub.ac.be

Abstract. One of the key components for achieving flexible, robust, adaptive and open-ended language-based communication between humans and robots - or between robots and robots - is rich deep semantics. AI has a long tradition of work in the representation of knowledge, most of it within the logical tradition. This tradition assumes that an autonomous agent is able to derive formal descriptions of the world which can then be the basis of logical inference and natural language understanding or production. This paper outlines some difficulties with this logical stance and reports alternative research on the development of an ‘embodied cognitive semantics’ that is grounded in the world through a robot’s sensori-motor system and is evolutionary in the sense that the conceptual frameworks underlying language are assumed to be adapted by agents in the course of dialogs and thus undergo constant change.

1 Introduction

Human language like communication with robots remains today a very distant goal. A few decades ago the problem was almost entirely on the side of robots. There were not enough physical robots to work with and the scarce robots that were available were unreliable, difficult to control and had only weak sensing capabilities. Also the computing power and electronics available for sensing and motor control had strict limitations. This situation has changed significantly the past few years. There are now thousands of powerful robots in the world and their capacities in terms of embodiment, sensori-motor potential and computing power, are quite sufficient for high level tasks. The intense activity around the Robocup and the new developments towards standardized components for robotics, such as ROS, are illustrative of this trend and it bodes well for future research. On the other hand, research on natural language processing appears not ready to exploit these new robotic capabilities. After promising work with systems like Shrdlu [19] or Shakey [9] in the early seventies, the ARPA speech understanding projects in the eighties [5], and the Verbmobil project in the nineties [18], the quest for artificial systems that could understand language and produce themselves goal-directed communication slowed down and research in computational linguistics became dominated by statistical language processing.

There is no doubt that statistical language approach has been very successful and is of practical use. Statistical language processing relies on a large corpus of

example sentences (the larger the better) and on general purpose machine learning algorithms. It basically attempts to develop language models that predict the probability of a word occurring in a sentence given the previous words. This approach stands in contrast to the one explored in earlier deep natural language processing research which used sophisticated grammars based on linguistic theory and procedural semantics for the precise interpretation of meaning in terms of world models derived from sensing and actuating. Parsers try to extract rich grammatical structures of sentences before interpreting them and producers used sophisticated planning techniques to determine what to say and then map meaning into words and grammatical constructions.

The main reasons why statistical language processing became more popular are as follows:

1. Human languages are unlike programming languages in the sense that sentences are rarely fully grammatical. Often only partial fragments are communicated and errors in meaning, grammar use, word choice, or pronunciation are very common due to the speed with which utterances need to be produced. Consequently parsers that rely on sentences being grammatical easily break down on real input. Statistical language processing handles this problem by being rather shallow in terms of the syntactic structures that are extracted, sometimes even relying only on sequential structure instead of hierarchy [3]. Often these shallow structures are enough for tasks that are needed by search engines.
2. Grammars of human languages are extraordinarily complicated. It therefore became clear quite early in language processing research that it would be extremely hard to design grammars and lexicons by hand. Some form of automatic language learning is essential, and the most effective way to do so at the moment is to use statistical machine learning techniques.

But what if the goal is to use language for interacting with complex devices such as robots? Shallow parsing is not sufficient because the rich grammatical structures underlying sentences are there to help listeners grasp meaning. If we ignore them we deprive ourselves of an important source of information. Lack of semantics or shallow semantics is too risky because it may lead to actions by the robot which are inappropriate or outright dangerous. Language production must rely on careful planning of meaning and this meaning needs to be the basis of sentence formulation as opposed to retrieving from memory sentence fragments that have tended to occur in similar circumstances. Most importantly it is also crucial that meaning gets grounded in the context through the sensori-motor apparatus of the robot, and unless we have corpora that contain vast amounts of data on grounded interactions it is not possible to apply statistical machine learning techniques.

This paper is part of a research effort that has deep grounded language understanding and production again as the main target. We have been carrying out experiments in which humanoid robots play language games about real world scenes that they experience through cameras and sensori-motor embodiment (see figure [1] from experiments in spatial language as discussed in [12]).

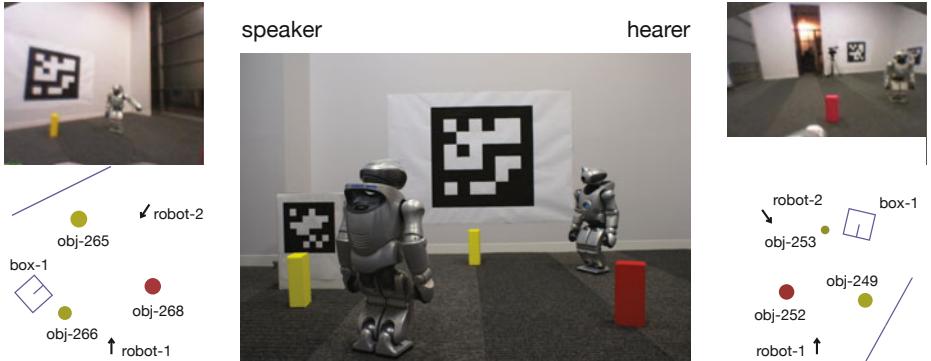


Fig. 1. Example of experimental set-up in language game experiments. They always involve two robots which share some reality. The images on the left and right show (top) what each robot sees and (bottom) the world model they derive from vision. The robots have a cooperative goal, for example one robot tries to draw attention to another robot, and they need to conceptualize reality and formulate utterances to achieve this goal. In this example, the robot might for example say: "the yellow block right of you".

This requires work on two fronts: language processing and semantics. Language processing requires novel solutions to reach the flexibility and open-endedness that remains problematic for deep parsing systems. How we approach this problem in our group is discussed in other papers (see e.g. [16]). Here I focus only on the semantics side.

Work on (grounded) semantics in AI has mostly been logic-based, more specifically within the tradition of logical empiricism, which was imported into AI through the early influence of John McCarthy [8]. This tradition has its root in the work of early logicians like Frege and Russell, and the research program of the later Vienese Circle, which included Carnap, Reichenbach, (early) Wittgenstein and others. In the hands of AI researchers, these theoretical proposals have been operationalised in a brilliant way and it has lead to a wealth of applications in problem solving, expert systems, semantic web, common sense, etc. The recent Computers and Thought IJCAI address of Kowalski [6] shows that this logical framework still forms one of the core approaches within AI.

The logical approach was originally designed as a normative framework for streamlining rational deliberative thinking and for giving some guarantee that inference steps are correct. It therefore focuses on the connection between propositions and how truthvalues are preserved across propositions. The framework of logic appears entirely adequate for this task. The question here is whether this framework is adequate for acting as the core of a grounded language processing system.

The logic-based approach simply makes the assumption that the 'Language of Thought' is "a simplified and canonical form of unambiguous sentences in natural language" [6], p.2. It is canonical in two ways: in terms of the concepts that are

being used, which are considered to be universal and shared, and in terms of the way these concepts are used to construct more complex propositions.

Using a canonical representation has at first sight many advantages, because natural language expressions which appear equivalent in terms of meaning can be mapped to the same representation and other processes, such as visual perception, can construct representations using the same form and the same conceptual content. However, deep issues come up concerning the question how propositions are to be expressed, how they are grounded in reality through a sensori-motor embodiment, and how they relate to natural language.

The rest of this paper discusses these issues in more detail (section 2) and how we should go about building AI systems to support human-like language communication in the light of this criticism (section 3).

2 The Nature of Conceptualisation

The main problem with the logical stance is that it trivialises conceptualisation. Conceptualisation is the process whereby language users categorize the world in order to talk about it. For example, for the sentence "the box left of the table" the speaker has categorized the two objects involved in terms of classes ("box" and "table"), and introduced a spatial relation "left of" between them. Apparently in this context it is clear which unique box and table are intended because the article "the" signals that they are 'definite'. There is also implicitly a perspective on the scene, because left-of is relative to the point of view of the speaker with respect to the table and the hearer.

2.1 Conceptualization Relies on Cognitive Operations

It is well known today that mapping concepts to reality requires a vast amount of signal processing and pattern recognition. These activities can be organized in terms of *cognitive operations* which perform segmentation, feature extraction, dimensionality reduction, transformation, classification, set operations, etc. A typical example of a cognitive operation is the application of the kind of classifiers acquired by Support Vector Machines, which are based on a representation of the examples as points in space and hyperplanes that represent the largest separation between classes. An example of transformation is the computation of location of objects to transform a perceived scene from a viewer-centered coordinate system to that of another agent or object in the scene, which is crucial to conceive or understand sentences such as "the box left of the table from your perspective" or "the pub left of the townhall" (assuming implicitly that you stand in front of the townhall). Set operations include grouping elements into sets, picking elements out of sets, computing unions or intersections, etc.

Whereas in the early stages of AI these cognitive operations were thought to be straightforward and yielding a clear outcome that then would result in the set of propositions making up a world model, it is now known that the matter is not that simple.

1. Cognitive operations can be highly complex and therefore they need to be actively invoked, for example it is not computationally feasible to perform geometric perspective transformations for every object in the scene or to categorize all objects in all possible ways, or to compute all possible groupings of elements into sets. Understanding a natural language sentence requires an active process of carrying out cognitive operations (rather than matching sentence meaning to a prior body of propositions) and producing a sentence requires planning the cognitive operations that the hearer should carry out.
2. Cognitive operations seldom yield clear-cut results. There are usually several alternative solutions with a degree of fit. This is why approaches such as fuzzy semantics have been developed. Which alternative is ultimately chosen in language communication will depend on many additional factors, in particular their relevance for the overall communicative goals of dialog partners and what is already available from prior discourse.
3. If the features and categories available for conceptualization are being learned, then the outcome of a cognitive operation will depend heavily on the data that has been seen so far, and this outcome will keep changing as learning proceeds. This means for example that an object classified in one way early in the learning process may become classified in a different way later, and this implies in turn that conceptualizations of a particular scene may change as a result of learning. This is another indication that we cannot simply assume that there are stored canonical representations of scenes in terms of propositions based on static concepts.

2.2 Conceptualization Is Strongly Context-Dependent

The way reality is conceptualized for language depends partly on the concepts and compositional combinations that speakers and listeners can handle. But it depends also very strongly on the context in which the communication takes place. This is for example very obvious in spatial language. If we conceptualize how the spatial location of an object is to be communicated, we need to take into account the position of speaker and listener with respect to various landmarks that we might use. Some of the landmarks (or the object itself) may not be visible to the listener, and a particular spatial relation may be different depending on the point of view (e.g. "left of" can be reversed if the listener is located opposite of the speaker). This context-dependence makes it very difficult to present in a canonical way the facts about a particular scene.

2.3 Human Languages Use Inferential Coding

Human language expressions often leave out aspects of meaning which are in fact critical for proper understanding (such as the perspective in the phrase "the box left of the table"). This is possible because human languages (in contrast to programming languages) are inferential coding systems that assume intelligence from the part of the listener as well as a shared context which does not need to

be described [11]. This implies that the listener must be able to fill in large parts of the cognitive operations that the speaker assumes, in addition to the ones that are explicitly evoked by the utterance itself. The meaning directly conveyed by the utterance is the tip of the iceberg of what is required to fully grasp the intention of the speaker.

Human languages are also unlike programming languages in the sense that they do not specify which cognitive operations the listener has to perform but just provide the arguments for cognitive operations. For example, the speaker simply says "box" to mean 'categorise the objects in reality in terms of whether they are a box and retain those that are members of this class'. Very often the same concept can be used in many different ways, as in :

1. He slows down.
2. You should take the slow train.
3. This train is slow.
4. The slower train nevertheless arrived earlier.
5. Take the slow one.

All these expressions use the same concept 'slow', but they implicitly invoke different cognitive operations. For example, in case 4, trains have to be ordered in terms of the time they take to reach their destination. In case 1 slow is used to describe an action in terms of change in the speed with which the action is going on. The action itself is not classified. In case 3, the speed of the train is compared to the speed that is normally expected.

These examples show clearly that a lot of the features and categorisations needed to understand an utterance are indirectly invoked by the speaker. The listener must not only reconstruct the meaning explicitly conveyed in the utterance but also fill in all the contextual details that are required to interpret this meaning in the present context.

2.4 Conceptualization Is Language and Culture Specific

Research into cognitive linguistics of the past decade has unmistakably shown that there are significant differences in the way that different languages and cultures conceptualize reality (see e.g. [17]). Some languages (and some speakers of a certain language) will prefer one way over another and these preferences are engrained in the language and thus culturally transmitted. Those claiming that there is a canonical way usually use their own language as the measure of other cultures and languages, and cannot imagine that the world can be conceptualized in different ways. But the - by now abundant - data of cultural influence of language on conceptualization cannot be ignored.

Let's take a domain which at first sight is a good candidate for being universally shared, namely color. Names for the basic hues (like red, green, blue, etc.) have been most intensely studied and although there was for some time a consensus that color terms are based on universal color foci, it is now clear that there are not only important individual differences in color naming between individuals of the same language group, but also significant differences in

color boundaries between different languages [10], p. 442. These cultural differences become even more important when non-basic colors are considered or more complex color expressions (such as 'a whiter shade of pale'). Profound cultural differences show up in all of the other domains whose semantics has recently been studied, such as for example the domain of space. [2]

Given that concepts are learned and that there are so many alternative approaches possible, it would be very odd if every individual shared the same canonical way for internally representing reality. In fact, it could only be explained by assuming that there is a strong innate bias to categorize the world. But this raises the issue where these strong biases come from, particularly for the many concepts that have only become relevant quite recently and are without much effort picked up by the majority of the population. A much more plausible explanation is that concepts are culturally evolving and become shared by the members of a population through tacit agreement as a side effect of communication. So our challenge as AI researchers is to work out how this might be possible.

3 An Evolutionary Approach

AI has drawn and contributed a lot to many disciplines such as decision theory, logic, linguistics, psychology and of course computer science. However, it has been less influenced by biology, and particularly by evolutionary theory. For biologists all aspects of living systems are considered to undergo constant change, either on a small time-scale in the form of adaptation and development, or on a longer time scale in terms of evolution. The main point of this paper is that we need to take a similar point of view with respect to intelligent processing in general and language in particular, and that we should be borrowing a lot more concepts from biology for achieving artificial intelligence.

Concretely, this means in the present context that we should view language as a complex adaptive system that undergoes cultural evolution at all levels [14]. Language users not only invent and align their sound systems, modes of interaction, and lexical and grammatical expressions but also the way they conceptualize reality. The linguistic system is not static because language users invent new conceptualizations and new modes of expression and reuse them in future communications. Which of these survive in future discourse is based on the communicative needs and environments that language users encounter and the need to dampen cognitive effort as much as possible, in other words the selectionist forces are cultural rather than biological (survival). Importantly, cultural evolution not only takes place at the level of surface forms (e.g. the use of specific words) but also at the level of semantics. The remainder of this section briefly sketches some of the directions that we have taken to explore this point of view further.

3.1 Language Games

The observations in the previous sections have shown that we cannot investigate semantics without context and without the cooperative goals that motivate

communication. We therefore structure the investigation in terms of *language games*. A language game is a turn-taking interaction between at least two autonomous agents (human-robot or robot-robot) drawn from a population. Each agent can either be speaker or hearer which implies that they are capable of both understanding and producing language. There is a common shared goal, for example execute an action or pay attention to an object in the scene. The speaker conceptualizes reality for achieving this goal within the present context, transforms the conceptualization into an utterance, and the hearer has to parse the utterance and reconstruct the meaning. Non-verbal communication, for example pointing gestures, form an integral part of a language game and are often needed as a complementary source of information particularly in language learning.

In our experiments, agents either started with scaffolded partial inventories for concepts and language, inspired by human languages, and then play language games and expand and adjust these inventories in order to be successful in the game. Occasionally we start from experiments where no initial inventories are given and agents invent concepts and expressions and coordinate them in the population, clearly showing that cultural evolution can give rise to an emergent communication system that is adaptive. There is no need to have innate concepts to make language possible. An example of a result is shown in figure 2 (from [1]) which shows an experiment in the emergence of a color lexicon and color categories without initial scaffolding.

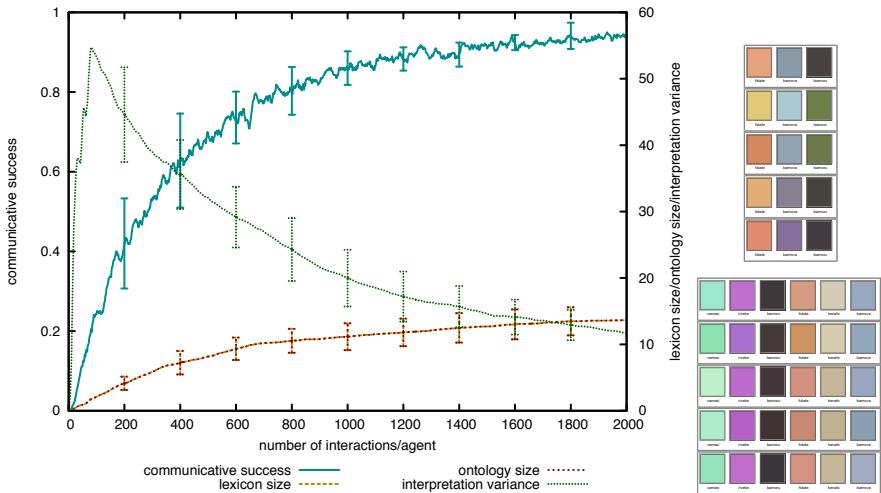


Fig. 2. Example experiments where a population of 5 agents bootstraps a color lexicon from scratch. The graph on the left shows how communicative success (meaning success in the language game) moves up to 95 %. Lexicon and ontology size (i.e. inventory of color concepts) grows steadily to be adequate for the game. The interpretation variance steadily decreases. On the right are two stages in the emergence of the color categories. The top is at the initial stage with less concepts and less coherence.

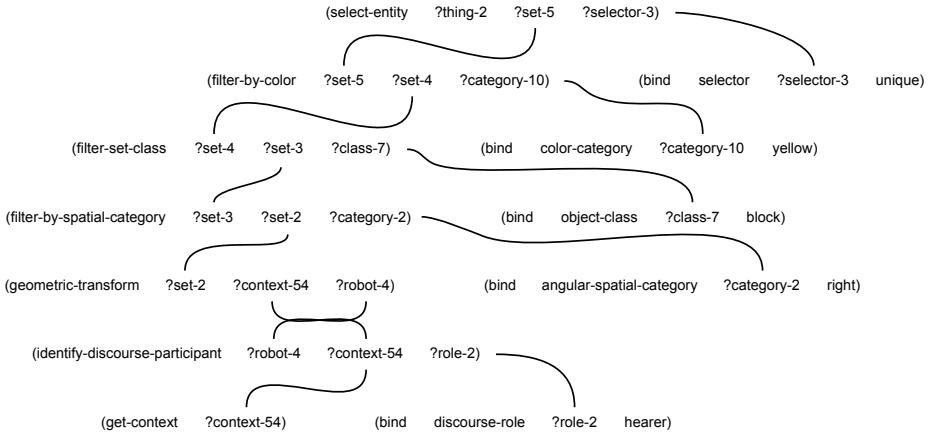


Fig. 3. Example of an IRL network. Select-entity, filter-by-spatial-category, etc. are names of cognitive operations. The variables (indicated with a question-mark, such as ?set-5 or ?selector-3) are slots that are filled in a constraint propagation process.

3.2 Internal Representations

To handle the meaning that is expressed by utterances, we have designed in our group a system called IRL (Incremental Recruitment Language) [14], [15], [7] which comes with an (open) library of cognitive operations that can be linked in networks as shown in figure 3. The networks operate using a data flow principle as in constraint languages, i.e. each cognitive operation can be used in multiple directions and as soon as enough information is available, the operation is executed and results propagate in the network.

Each cognitive operation uses an inventory of conceptual building blocks (semantic entities) that can be used to perform the cognitive operation on the contextual input. For example, a classifier will need an inventory of possible classes. This inventory is dynamic as it is expanded by learning processes to cope with more data or more communicative situations.

Such IRL-networks are used as the meaning of utterances and lexical and grammatical processes translate the networks into surface forms or reconstruct the meaning through a parsing process. Often the networks are incomplete, in which case the planning system that is used to come up with a network in language production is re-invoked by the hearer to fill in missing parts.

3.3 Diagnostics and Repairs

We have found that a meta-level architecture is very useful to organize the way in which agents cope with robustness, flexibility, continuous adaptation and evolution (see Figure 4). This architecture uses diagnostics to monitor the outcome of ongoing object level processes and repairs to expand the inventory

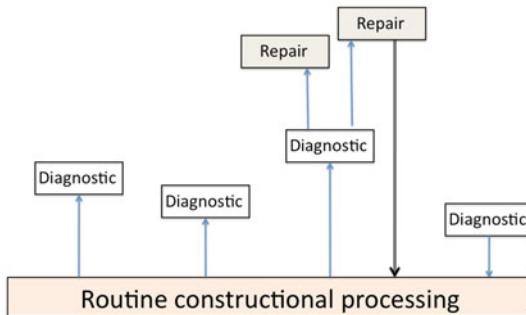


Fig. 4. The routine application of constructions during parsing and production is augmented with meta-level processes performing diagnosis and possibly repairing problems by extending or aligning the inventory of the speaker or the hearer

of semantic entities used by cognitive operations or to re-organize networks to better cope with the needs of communication.

For example, one diagnostic associated with a classifier would notice that the classifier is unable to distinguish two objects in the scene and this could possibly trigger a repair action in which the inventory of classifiers is expanded with a new classifier. Another diagnostic for classifiers would trigger when the listener observes that an object classified in one way by the speaker is classified in another way by himself, and this then could trigger a repair action in the form of an adjustment of the inventory.

The planning of networks is a search process like any planning process, and standard techniques such as chunking and storing of obtained solutions for future references are used to avoid search and progressively bootstrap the system to handle more complex conceptualizations. This progressively yields an inventory of standardized conceptualization strategies which have also standard forms of expression in language and therefore become culturally shared and transmitted.

3.4 Alignment

Research on natural dialog carried out by psychologists such as Simon Garrod [4] has abundantly shown that partners in dialog align themselves at all levels, not only phonetic, lexical and grammatical but also at the conceptual level. They adjust their concept inventories and conceptualization strategies on the basis of the outcome of a communication. For example, partners in dialog may adjust the prototypes of some of their color concepts so that they become more similar and hence that they have a higher chance for mutual understanding (see figure 5 from [1]). Alignment can easily be operationalized as part of the meta-level architecture discussed in the previous subsection, and when the right alignment operations are used, it operates very effectively.

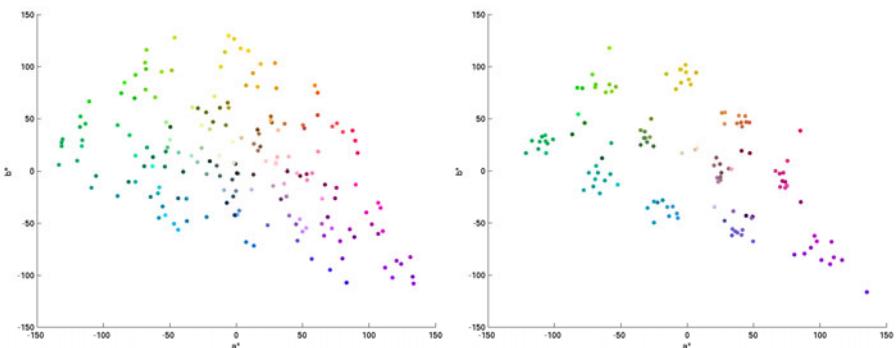


Fig. 5. Example experiment in the emergence of color categories through language games within a population of 10 agents. The left shows the prototypes of all agents with alignment and the right without alignment. Without alignment, the categories are scattered more or less randomly over the color space. With alignment, the color categories cluster around certain regions of the color space.

4 Conclusions

Handling grounded deep semantics for human-like language understanding and production requires a reconsideration of some of the foundations of AI, particularly with respect to the logical approach which has informed much of past research. This paper emphasized that we need to view language and meaning as a complex adaptive system that is continuously undergoing change, as it is shaped and reshaped by language users in order to satisfy their needs within the ecological settings they are confronted with. This means that we need to take an evolutionary perspective on semantics instead of assuming that the building blocks of meanings and their usage in language communication is static and a priori shared.

References

1. Bleys, J.: Language Strategies for Color. In: Steels, L. (ed.) *Experiments in Cultural Language Evolution*. John Benjamins Pub. Co., Amsterdam (2011)
2. Evans, N., Levinson, S.: The myth of language universals: Language diversity and its importance for cognitive science. *Behavioral and Brain Sciences* 32(5), 429–492 (2009)
3. Frank, S., Bod, R.: Insensitivity of the human sentence-processing system to hierarchical structure. *Psychological Science* (to appear, 2011)
4. Garrod, S., Anderson, A.: Saying What You Mean in Dialogue: A Study in Conceptual and Semantic Coordination. *Cognition* 27, 181–218 (1987)
5. Klatt, D.: Review of the ARPA speech understanding project. In: *Readings in Speech Recognition*. Morgan Kaufmann Publishers Inc., San Francisco (1990)
6. Kowalski, R.: Artificial Intelligence and Human Thinking. In: IJCAI 2011. Computers and Thought Lecture (2011), <http://www.doc.ic.ac.uk/~rak/papers/IJCAI2011.pdf>

7. Spranger, M., Pauw, S., Loetzsch, M., Steels, L.: A Formalism for Embodied Cognitive Semantics. In: Hild, M., Steels, L. (eds.) *A Whole-Systems Approach to Language Grounding in Robots*. Springer, Berlin (2011)
8. McCarthy, J.: Programs with Common Sense. In: *Mechanisation of Thought Processes*. In: Proceedings of the Symposium of the National Physics Laboratory, London, U.K, pp. 77–84 (1959)
9. Nilsson, N.: *Shakey The Robot*, Technical Note 323. AI Center, SRI International (1984)
10. Regier, T., Kay, P.: Language, thought, and color: Whorf was half right. *Trends in Cognitive Sciences* 13, 439–446 (2009)
11. Sperber, D., Wilson, D.: *Relevance*. Blackwell, Oxford (1986)
12. Spranger, M.: The co-evolution of basic spatial terms and categories. In: Steels, L. (ed.) *Experiments in Cultural Language Evolution*. John Benjamins Pub. Co., Amsterdam (2011)
13. Steels, L.: The emergence of grammar in communicating autonomous robotic agents. In: Horn, W. (ed.) *ECAI 2000: Proceedings of the 14th European Conference on Artificial Life*, pp. 764–769. IOS Publishing, Amsterdam (2000)
14. Steels, L.: Language as a Complex Adaptive System. In: Deb, K., Rudolph, G., Lutton, E., Merelo, J.J., Schoenauer, M., Schwefel, H.-P., Yao, X. (eds.) *PPSN 2000, Part VI. LNCS*, vol. 1917, pp. 17–26. Springer, Heidelberg (2000)
15. Steels, L.: The Recruitment Theory of Language Origins. In: Lyon, C., Nehaniv, C., Cangelosi, A. (eds.) *The Emergence of Communication and Language*, pp. 129–151. Springer, Heidelberg (2007)
16. Steels, L., Van Trijp, R.: How to Make Construction Grammars Fluid and Robust. In: Steels, L. (ed.) *Design Patterns in Fluid Construction Grammar*. John Benjamins Pub. Co., Amsterdam (2011)
17. Talmy, L.: *Toward a Cognitive Semantics: Concept Structuring Systems (Language, Speech, and Communication)*. The MIT Press, Cambridge (2000)
18. Wahlster, W.: *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, Berlin (2000)
19. Winograd, T.: *Understanding Natural Language*. Academic Press, London (1972)

General Game Playing in AI Research and Education

Michael Thielscher

School of Computer Science and Engineering,
The University of New South Wales
`mit@cse.unsw.edu.au`

Abstract. Introduced in 2005 as a new AI Challenge and Competition, general game playing has quickly evolved into an established research area. More recently it is also gaining popularity as a useful addition to AI curricula at universities around the world. The first part of this paper will survey the research landscape of general game playing, which covers a broad range of classic AI topics, including knowledge representation, search, planning and learning. The second part will argue that general game playing provides a unique approach to teaching a number of different topics such as problem solving by search, logic, logic programming and planning. The inherent competitive aspect also can be used as a great motivator for students to design and implement their own AI systems.

1 Introduction

General game playing is the attempt to create a new generation of AI systems that are able to understand the rules of arbitrary games and then learn to play these games without human intervention. Unlike specialised systems like the chess program Deep Blue, a general game player cannot rely on algorithms designed in advance for specific games. Such a system rather requires a form of general intelligence that enables it to autonomously adapt to new and possibly radically different environments. General game-playing systems are a quintessential example of a new generation of software that end users can customise for their own specific tasks. This makes general game playing an interesting and challenging problem for AI, involving many fundamental issues such as reasoning, learning, planning and decision making. Consequently, general game playing can, and in fact should, be of interest to researchers in a variety of AI disciplines beyond conventional computer game playing. At the same time and for the same reasons, general game playing provides a new anchor for AI education as a unique—and attractive for students—framework for teaching multiple basic AI topics, such as problem solving by search, propositional and first-order logic, logic programming and planning.

This paper attempts to give an overview of what has been achieved, and what lies ahead, some five years after the introduction of general game playing to the broad AI community through the inauguration of the annual AAAI General Game Playing Competition in 2005 [7]. Initiated and organised by

Michael Gensereth and members of his Stanford Logic Group (and endowed with the attractive purse of \$10,000), the competition quickly caught the interest of a number of researchers, the author included, from different backgrounds. Yet the idea itself, that is, to build a system that can learn to play a whole array of games, has been around for over 40 years, going back to the French AI pioneer Jacque Pitrat, who wrote the first ever computer program that, in principle, could learn to play arbitrary chess-like board games by being given their rules [20]. Later general game-playing programs include [19][12], but it required the aforementioned AAAI competition to spark broad interest in this problem as an AI Grand Challenge. Some five years later, an impressive number of research groups on general game playing have been established world-wide, including the German universities in Berlin, Bremen and Potsdam. Most of these groups develop their own player, but also there is an increasing number of researchers who are interested in specific aspects of general game playing, which does not require them to build a full-fledged, competitive game-playing system. They all contribute to a fast growing body of literature on general game playing. Further indications that general game playing is a maturing research field are the establishment of a series of biennial IJCAI workshops on this topic in 2009¹, the inclusion of “general game playing” among the standard keywords at the AAAI Conference; and the first ever special issue of a journal on this topic².

The first part of this paper will survey the research landscape of general game playing. We will see that a surprisingly broad range of classic AI fields have a role to play in general game playing. In each case we will show how existing approaches, methods and systems are contributing to the foundations for general game playing, to the improvement of the quality of existing general game-playing systems, and to the development of new methodologies. But even if your interest as an AI researcher does not lie in general game-playing systems themselves, they can be used as a non-trivial application for a broad range of more theoretically motivated AI research. We will also report on current research trends, identify some of the most pressing open questions, and look at the possibilities to gradually broadening today’s concept of general game playing to involve even more aspects of AI.

In the second part of this paper, we will give an overview of how general game playing has entered AI education, either in form of an advanced AI graduate course, with lectures and tutorials but where the special focus lies on practical work; or as part of a general introductory course to AI. Examples for the former can be found in the curricula at Stanford University, Reykjavík University, Bremen University and TU Dresden, where it was held by the author for four consecutive years starting in winter 2006/7. Examples for the latter include the general introduction to AI for undergraduate students at the University of New South Wales to which the author contributed in Spring 2011. We will show why general game playing provides an excellent angle for teaching a variety of basic

¹ For GIGA’09 held in Pasadena see www.ru.is/faculty/yngvi/GIGA09; for GIGA’11 held in Barcelona see www.aiide.org/giga11.

² *KI—Künstliche Intelligenz*, volume 25, Springer Verlag 2011.

AI methods that also is a great motivator for students to design and implement their own AI systems. Our overview will include a survey of freely available teaching aides including slides, tutorial questions and programming tools, for the benefit of potential instructors.

2 The Research Landscape of General Game Playing

An outstanding characteristic of general game playing as an AI research topic is to involve a broad range of sub-disciplines with a focus on symbolic AI (as opposed to, say, RoboCup or the DARPA Grand Challenge for driverless vehicles). For this reason, general game playing has all the potential to become a rich source for interesting research problems in many different areas. As we will survey the research landscape, we will encounter several cases in which general game playing has been successfully used as an attractive—and challenging—application to demonstrate the viability of existing theories, methods, and systems. We will also see examples where the concept of general game playing has generated new research problems and solutions. Most importantly, it will become clear that it is not at all necessary to actually build a full-fledged, competitive player in order to make an original and significant contribution to this research challenge. Yet another characteristic of general game playing research is to often concern the combination and integration of two or more theories and methods, which naturally leads to collaborations involving different AI sub-disciplines.

In the relatively short time span since the first AAAI competition in 2005, at least four traditional AI disciplines have proved to be core aspects of research in general game playing:

1. Knowledge Representation
2. Search
3. Planning
4. Learning

In the remainder of this section, we will discuss in turn the role that each of these areas plays for general game playing: what interesting research problems they give rise to, which methods have been successfully applied, and what challenges lie ahead.

2.1 Knowledge Representation and Reasoning

General game playing requires a formal, symbolic language in which the rules of arbitrary games can be described to a system. The general *Game Description Language* (GDL) has been developed for that purpose [7]. It can be seen as a specific high-level action description language in the tradition of AI Planning languages like STRIPS or PDDL: game worlds are described with the help of individual features (e.g., the position of a piece on a game board), and moves are specified by their preconditions and effects. For example, the standard laws for chess include rules saying that a player whose turn it is can castle with the

kingside rook under certain conditions, and as a result if white performs this action his king will move to square (g,1). In actual GDL, this looks something like the following, where GDL-specific keywords are printed in *italic* and variables are indicated by a leading “?”.

```

1  (<= (legal ?player kingside_castle)
2    (true (turn ?player))
3    (can_castle_to_kingside ?player))
4  (<= (next (cell ?g ?1 ?white_king)))
5    (does white kingside_castle))
```

This would be accompanied by an axiomatisation of the domain-specific predicate *can_castle_to_kingside*(*x*) according to the standard rules for castling. Game rules like these can be considered as a formal representation of knowledge for general game-playing systems. They are the source of a variety of interesting and challenging reasoning problems.

Reasoning about actions. The use of GDL as the input language for general game-playing systems immediately raises research questions such as: How to transform GDL descriptions in order to improve the efficiency of logical reasoning? How to apply existing techniques for reasoning about actions to GDL? How to extend or modify GDL for other applications of general intelligence? These problems are being extensively studied, and recent results include mappings of GDL into Propositional Automata [27], C++ code [34], and Binary Decision Diagrams [11]. GDL has been successfully embedded into a structure-rewriting formalism [9], the functional programming language OCAML [21], and the Situation Calculus [25]. Extensions of basic GDL have been developed to capture imperfect-information games [32] and coalition games [29]. However, a number of challenging problems remain, including

1. the development of a transformation of GDL into an efficient encoding that scales to large games for which full-fledged grounding (i.e., propositionalisation) of the rules is impossible in practice;
2. generalised reasoning methods for randomised, imperfect-information games;
3. the design of general problem description languages for other domains to provide the foundation for general-purpose agents, general-purpose robots, or for any other modern equivalent of the classic General Problem Solver in other AI disciplines.

Automated theorem proving. A key to successful general game playing is the ability of a system to use the right strategies and heuristics when playing a new game. In many cases this requires a system to analyse the game rules with the goal to establish game-specific properties that are not explicitly given but follow implicitly. Since games are specified in a logical language, this amounts to a theorem proving task. Successful approaches include methods for extracting general properties like symmetries [22], factorability [4,26], or state invariants [33],

and methods for identifying prevailing concepts such as mobility, boards, and pieces [13][3]. Answer Set Solvers have been successfully deployed as general-purpose theorem provers in this context [24][31][18]. Among the open problems are:

1. the identification of other common properties that a general game-playing system should determine in order to improve game-play;
2. the development and application of new non-propositional theorem proving techniques for games whose GDL rules cannot be fully grounded in practice.

Other KRR techniques. The logic-based formalisation of game rules as the basis for general game playing implies that there are many as yet unexplored possibilities for other areas in Knowledge Representation to contribute to the general game playing effort. Some of a potentially large number of examples are

1. *Reasoning About Uncertainty* applied to games with randomised moves and where players have imperfect information;
2. *Belief Revision* employed, for example, to represent and revise beliefs about the opponents, as a way to build and maintain opponent models;
3. *Spatial Reasoning* to identify and reason about spatial relations in games.

2.2 Search

Once a general game-playing system is capable of computing legal moves and position updates from the game rules, it can search through the space of possible ways in which the game can proceed. Of course only very simple games allow a complete brute-force search in practice. Hence, one of the fundamental challenges for general game playing is to develop intelligent search techniques without knowing the particulars of the game to which they will be applied. The early general game-playing systems applied standard iterative deepening search in combination with automatically constructed heuristic evaluation function [3][8][13][19][23]. More recently, the use of Monte Carlo simulations has become mainstream in general game playing as it provides a way to achieve good performance without the need to generate any game-specific knowledge [2][15][18][28]. The state-of-the-art search methods in existing general game-playing systems give rise to a number of interesting challenge problems.

Monte Carlo Tree Search. Any effective use of random playouts in general game playing requires effective search-control methods. The standard algorithm is UCT (for: upper confidence bound applied to trees) [2], originally introduced to computer game playing by Kocsis and Szepesvari in 2006. A key to improving the performance of simulation-based general game playing is the ability to learn game-specific knowledge to effectively guide the random UCT simulations. Some such techniques have been described in [6][10]. Among the open and challenging problems are the following.

1. the use of high-level game concepts such as those generated and used in heuristic evaluation function-based players for search control in Monte Carlo simulations;
2. the automation of the decision about the appropriate search technique and enhancement at runtime;
3. the generalisation of simulation-based methods for general game playing to imperfect-information games, where a player may be highly uncertain about the current game state.

Informed search. Informed search uses problem-specific knowledge to overcome the inefficiency of blind search. General game playing raises the challenge that any such knowledge needs to be automatically extracted. Existing methods for computing symmetries [22] and factors [426] can be used to cut down the search space. A recent method for computing general distance measures from the mere rules of a game can be used to automatically generate admissible search heuristics [16], but interesting problems remain, including

1. the use of other types of automatically extracted game-specific knowledge for effective search control;
2. the application of further informed search methods to general game playing, for example, local search algorithms.

2.3 Planning

Planning is closely related to general game playing, as both are instances of general problem solving, where the specifics of a problem are unknown until runtime. The game description language GDL can in fact be seen a special-purpose action description language that follows the tradition of planning languages. The latter, however, describe a problem from the perspective of a single agent, even in case of adversarial planning, and GDL generalises this to the presence of other agents that have their own actions and goals. Reasoning about the intentions of the other players is the basis for Opponent Modelling—one of the crucial aspects in which general game playing goes beyond planning. A further difference is that planning is mainly concerned with solving a problem upfront, whereas much of general game playing is concerned with finding a good course of actions without being able to see all the way to the end of a game.

Still, many aspects of planning are very relevant for general game playing, and already some results have been successfully transferred: In [5] it is shown how symbolic planning technology can be applied to explore the search space for both single-player and two-player games. Planning as Answer Set Programming has been successfully adapted to general game playing in [31]. The abovementioned automatic identification of symmetries [22] and decompositions [426] and their use for game tree search are generalisations of similar results for planning, too. Some interesting open aspects at the boundary of planning and general game playing are the following.

1. the development of an automatic translations of GDL into an existing planning language such as PDDL, which is challenging because GDL does not describe negative effects of actions explicitly (and instead appeals to the principle of negation-by-failure [14]);
2. the adaptation of successful planning heuristics, such as relaxation, to general game playing, which is challenging for similar reasons;
3. the generalisation of other planning techniques, e.g. hierarchical planning;
4. the converse application of techniques developed in general game playing, such as opponent modelling, for adversarial planning.

2.4 Learning

The very idea of general game playing is to build systems that automatically learn to master arbitrary new games. It is therefore justified to consider learning to be *the* core research area for this AI Challenge. Thus it may be somehow surprising that general AI learning methods are not (yet) dominating the state of the art in general game playing. The main reason seems to be the sheer difficulty of the problem for general-purpose learning algorithms, which is why existing, successful applications of learning techniques in general game playing focus on improving individual aspects of specific approaches: In [1], transfer learning has been applied to generalise knowledge from one game for other games. Neural networks have been used in [17] to improve automatically generated evaluation functions. In [6], statistical and reinforcement learning has been employed to provide search guidance in Monte Carlo simulations by biasing the random selection of moves. A variant of temporal difference learning has been developed in [10] to automatically construct features that are correlated with the outcome of a match. Also decision trees have been applied to evaluate states [30].

Despite these first attempts to employ learning techniques to improve the quality of various approaches, the application of learning in general game playing is still in its infancy. An obstacle may be that the current format of competitions is not favourable for many general learning methods, for two reasons: a comparably short amount of time to learn a new game (typically in the order of 10 minutes) and the fact that games are rarely repeated, if at all. General game playing as a principle, however, is more general than suggested by the special format of contemporary competitions. It can certainly be expected that learning will play an increasingly important role when systems have both more time—ideally using life-long learning—and more data to learn from besides random playouts, including games they lost and games played by masters.

2.5 Further Relevant Research Areas

In addition to the four AI disciplines that today build the core of general game playing research, other areas are likely to eventually become relevant as the scope of general game playing is broadened.

Decision making. Current state-of-the-art research in general game playing focuses on games that are deterministic and in which players have complete knowledge of the state. As we progress towards the general case of randomised, imperfect-information games, aspects of general decision making will become increasingly important, including utility functions, Markov decision processes, and game-theoretic solutions.

Natural language processing. Most end users of general game-playing systems will expect a better way than using GDL to input the rules of their own invented games. This could be achieved with the help of some controlled form of natural language for describing games. In most cases, this will include a process of disambiguation through a dialogue between the player and its user. This generalisation seems particularly attractive because it could be met in a mostly modular fashion, where an existing general game player is coupled with a pre-processing natural language interface.

Game-playing robots. General game-playing software provides a great opportunity to make a relatively simple robotic system act smart. A robot arm capable of moving pieces on a board, coupled with a state-of-the-art player, can in principle learn to play arbitrary games with these pieces. Beyond this, interesting challenges are posed by game-playing robots that learn to recognise and manipulate new pieces, or mobile robots that can solve a whole array of new tasks formulated as games.

3 General Game Playing in AI Education

The fact that it addresses so many different aspects of AI makes general game playing a valuable tool for AI education, too, as it allows to teach a variety of methods on a single subject. A further advantage of general game playing is to naturally lead to practical assignments, where students can experiment with various techniques to address an interesting and challenging problem. Plus, there is the competitive aspect when students' players are pitted against each other for evaluation, which works as a great motivator. With today's available software tools and online resources for teaching general game playing, this is also much easier to organise than, say, a full-fledged robotics laboratory.

In this second part of the paper we give an overview of how general game playing can help with teaching the following core topics of AI:

1. Problem Solving (search, constraint satisfaction, game playing)
2. Knowledge and Reasoning (logic, inference, knowledge representation)
3. Planning

This will be complemented by an overview of teaching resources that are available for download, including slides, example tutorial assignments and software support for teaching general game playing.

3.1 Problem Solving

Problems solving by searching a solution space is often the first topic in an introductory AI course. Since general game playing is, in a certain sense, general problem solving, all of the following standard methods can be very nicely motivated and illustrated in this setting.

- *Problem formulation*, that is, modelling a problem as a single-player game defining a state search space;
- *Uninformed search strategies* like breadth-first, depth-first, iterative deepening etc., and methods for avoiding repeated states;
- *Informed search* using heuristic functions;
- *Constraint satisfaction* as a special kind of general problem formulation;
- *Adversarial search* using the minimax algorithm and alpha-beta pruning.

In fact, many of the standard examples used in textbooks to illustrate these methods are classic games, such as the 15-puzzle for state-based search, the 8-queens problem for constraint solving, and tic-tac-toe for minimax and alpha-beta pruning.

3.2 Knowledge and Reasoning

General game playing requires to axiomatise the rules of games, which makes it an ideal case study for learning how to use symbolic logic to represent knowledge. Through the problem of describing a game formally, students learn to choose appropriate symbols for representing the various elements of a game (such as state features and moves) and to formulate in (propositional or first-order) logic laws that are given in natural language. A basic general game-paying system must be able to reason about games described in GDL in order to determine its legal moves and their effects etc., which can be used to motivate studying various inference methods. In this way, general game playing allows to address all of the following topics:

- *Propositional logic*, using simple games with a small state space;
- *Resolution* to draw inferences from game rules;
- *First-order logic* for games with rules that are naturally described using variables and quantification;
- *Logic programming* as the foundation for a basic general game player.

3.3 Planning

We have already argued earlier, in Section 2.3, why general game playing is closely related to planning and in fact can be seen as a generalisation thereof. Hence, all aspects of planning can naturally be taught under the umbrella of general game playing, too:

- *Classic Planning* for solving single-player games;
- *Continuous Planning* for playing single-player games;
- *Conditional Planning* for nondeterministic and partially observable games;
- *Multiagent- and Adversarial Planning* for cooperative/competitive games.

3.4 Teaching Resources

The author wants to stress that none of the above is meant to suggest conveying to students the impression that general game playing is the only reason for studying problem solving, knowledge representation or planning. Rather, the idea is to provide a domain to which students can repeatedly return in order to practice a new theory and experiment with a new method. Thus there are two principled ways to use general game playing in AI education, both of which the author and a number of other instructors at various universities have successfully used in the past:

1. A specialised graduate course on general game playing, the ultimate goal of which is for students to develop their own players and where, along the way, they learn to apply the various relevant AI methods. Experiences with such a course at Stanford University and TU Dresden over several years suggest that the right size for such a course is about 15–30 participants with groups of 2–4 (ideally, 3) students designing one player.
2. Integrating general game playing into a general introductory AI course, both on the undergraduate and the graduate level.

For either type of course, the website www.general-game-playing.de is a rich source for prospective instructors of free supporting material:

1. Complete sets of slides from several universities worldwide for full-fledged general game playing courses and for parts of introductory AI courses.
2. Sample tutorial questions and assignments.
3. Basic players in several programming languages available for download that students can be provided with as a starting point.
4. Software to run a student competition in class.

Any general introductory course to AI can in fact profit from the available tools even without reference to general game playing: Instructors can use the general game control software to let students experiment with any concrete game or agent domain, like e.g. the famous Wumpus World, for which a formalisation in GDL exists.

The author welcomes any additional material or pointers to new courses on general game playing.

4 Conclusion

General game playing is an exciting, still young but on the verge of maturing topic, which touches upon a broad range of aspects of artificial intelligence. In this paper we surveyed the research landscape of general game playing in an attempt to show its many facets and that it provides a rich source of interesting and challenging problems for many an AI researcher. We also showed that general game playing provides a unique approach to teaching a number of different topics in AI. Students who got exposed to the idea of a general game-playing AI system have repeatedly described it as “cool,” and the author is inclined to agree.

Acknowledgement. The author is the recipient of an Australian Research Council Future Fellowship (project number FT 0991348).

References

1. Banerjee, B., Stone, P.: General game learning using knowledge transfer. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Hyderabad, pp. 672–677 (2007)
2. Björnsson, Y., Finnsson, H.: CADIAPLAYER: A simulation-based general game player. *IEEE Transactions on Computational Intelligence and AI in Games* 1(1), 4–15 (2009)
3. Clune, J.: Heuristic evaluation functions for general game playing. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 1134–1139 (2007)
4. Cox, E., Schkufza, E., Madsen, R., Genesereth, M.: Factoring general games using propositional automata. In: Proceedings of the IJCAI Workshop on General Intelligence in Game-Playing Agents (GIGA), pp. 13–20 (2009)
5. Edelkamp, S., Kissmann, P.: Symbolic exploration for general game playing in PDDL. In: Workshop on Planning and Games at the International Conference on Automated Planning and Scheduling, ICAPS (2007)
6. Finnsson, H., Björnsson, Y.: Learning simulation control in general game-playing agents. In: Proceedings of the AAAI Conference on Artificial Intelligence, Atlanta, pp. 954–959 (2010)
7. Genesereth, M., Love, N., Pell, B.: General game playing: Overview of the AAAI competition. *AI Magazine* 26(2), 62–72 (2005)
8. Kaiser, D.: Automatic feature extraction for autonomous general game playing agents. In: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), Honolulu (2007)
9. Kaiser, L., Stafniak, L.: First-order logic with counting for general game playing. In: Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco (2011)
10. Kirci, M., Sturtevant, N., Schaeffer, J.: A GGP feature learning algorithm. *KI—Künstliche Intelligenz* 25, 35–42 (2011)
11. Kissmann, P., Edelkamp, S.: Gamer, a general game playing agent. *KI—Lünstliche Intelligenz* 25, 49–52 (2011)
12. Koller, D., Pfeffer, A.: Representations and solutions for game-theoretic problems. *Artificial Intelligence* 94(1), 167–215 (1997)
13. Kuhlmann, G., Dresner, K., Stone, P.: Automatic heuristic construction in a complete general game player. In: Proceedings of the AAAI Conference on Artificial Intelligence, Boston, pp. 1457–1462 (2006)
14. Love, N., Hinrichs, T., Haley, D., Schkufza, E., Genesereth, M.: General Game Playing: Game Description Language Specification. Technical Report LG-2006-01, Stanford Logic Group, Stanford University (2006), games.stanford.edu (2006)
15. Méhat, J., Cazenave, T.: A parallel general game player. *KI—Künstliche Intelligenz* 25, 43–47 (2011)
16. Michulke, D., Schiffel, S.: Distance features for general game playing. In: Proceedings of the IJCAI Workshop on General Intelligence in Game-Playing Agents (GIGA), Barcelona, pp. 7–14 (2011)

17. Michulke, D., Thielscher, M.: Neural networks for state evaluation in general game playing. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5782, pp. 95–110. Springer, Heidelberg (2009)
18. Möller, M., Schneider, M., Wegner, M., Schaub, T.: Centurio, a general game player: Parallel, java-, and asp-based. KI—Künstliche Intelligenz 25, 17–24 (2011)
19. Pell, B.: Strategy Generation and Evaluation for Meta-Game Playing. PhD thesis. University of Cambridge (1993)
20. Pitrat, J.: Realization of a general game playing program. In: Morrell, A. (ed.) Proceedings of IFIP Congress, Edinburgh, pp. 1570–1574 (1968)
21. Saffidine, A., Cazenave, T.: A forward chaining based game description language compiler. In: Proceedings of the IJCAI Workshop on General Intelligence in Game-Playing Agents (GIGA), Barcelona, pp. 69–75 (2011)
22. Schiffel, S.: Symmetry detection in general game playing. In: Proceedings of the AAAI Conference on Artificial Intelligence, Atlanta, pp. 980–985 (2010)
23. Schiffel, S., Thielscher, M.: Fluxplayer: A successful general game player. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, pp. 1191–1196 (2007)
24. Schiffel, S., Thielscher, M.: Automated theorem proving for general game playing. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Pasadena, pp. 911–916 (2009)
25. Schiffel, S., Thielscher, M.: Reasoning about general games described in GDL-II. In: Proceedings of the AAAI Conference on Artificial Intelligence (2011)
26. Schiffel, S., Thielscher, M., Zhao, D.: Decomposition of multi-player games. In: Nicholson, A., Li, X. (eds.) AI 2009. LNCS, vol. 5866, pp. 475–484. Springer, Heidelberg (2009)
27. Schkufza, E., Love, N., Genesereth, M.: Propositional automata and cell automata: Representational frameworks for discrete dynamic systems. In: Wobcke, W., Zhang, M. (eds.) AI 2008. LNCS (LNAI), vol. 5360, pp. 56–66. Springer, Heidelberg (2008)
28. Shafiei, M., Sturtevant, N., Schaeffer, J.: Comparing UCT versus CFR in simultaneous games. In: Proceedings of the IJCAI Workshop on General Intelligence in Game-Playing Agents (GIGA), Pasadena, pp. 75–82 (2009)
29. Sheng, X., Thuente, D.: Extending the general game playing framework to other languages. In: Proceedings of the IJCAI Workshop on General Intelligence in Game-Playing Agents (GIGA), Barcelona, pp. 7–14 (2011)
30. Sheng, X., Thuente, D.: Using decision trees for state evaluation in general game playing. KI—Künstliche Intelligenz 25, 53–56 (2011)
31. Thielscher, M.: Answer set programming for single-player games in general game playing. In: Hill, P., Warren, D. (eds.) ICLP 2009. LNCS, vol. 5649, pp. 327–341. Springer, Heidelberg (2009)
32. Thielscher, M.: A general game description language for incomplete information games. In: Fox, M., Poole, D. (eds.) Proceedings of the AAAI Conference on Artificial Intelligence, Atlanta, pp. 994–999 (2010)
33. Thielscher, M., Voigt, S.: A temporal proof system for general game playing. In: Fox, M., Poole, D. (eds.) Proceedings of the AAAI Conference on Artificial Intelligence, Atlanta, pp. 1000–1005 (2010)
34. Waugh, K.: Faster state manipulation in general game playing using generated code. In: Proceedings of the IJCAI Workshop on General Intelligence in Game-Playing Agents (GIGA), Pasadena, pp. 91–97 (2009)

Conversational Agents in a Virtual World

Peter Adolphs¹, Anton Benz², Núria Bertomeu Castelló², Xiwen Cheng¹,
Tina Klüwer¹, Manfred Krifka², Alexandra Strekalova²,
Hans Uszkoreit¹, and Feiyu Xu¹

¹ Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI) GmbH,
Alt-Moabit 91c, 10559 Berlin, Germany

² Zentrum für Allgemeine Sprachwissenschaft (ZAS),
Schützenstrasse 18, 10117 Berlin, Germany

Abstract. This paper presents a system that builds on theoretical and experimental insights from linguistic pragmatics, uses novel techniques from computational linguistics and combines them with robust baseline technologies to provide intelligent Non Player Characters (NPCs), which naturally act and talk in a virtual world. Current NPCs still lack the necessary linguistic knowledge and methods to apply them to the numerous conversational application areas in virtual worlds. The system presented in this paper manages two NPCs, a barkeeper and a furniture sales agent, which highly depend on conversational abilities.

1 Introduction

In the latter decade, Massive Multiplayer Online Role Playing Games and virtual reality environments, such as *Warcraft* and *Second Life*, are proliferating. In such games, players cohabit a 3D-environment with other players. To populate such 3D-worlds in the early stages of the game's existence and to fill roles which humans do not want to play, such as waiter, shop-assistant, etc. non-player-characters (NPC) are created. NPCs are virtual characters, who can serve different purposes in the game, ranging from providing information to helping the user to carry out some task. However, the application areas of NPCs are currently very limited, mainly due to their marginal linguistic capabilities. There are very few commercial games which handle linguistic input¹.

Recently, there has been some research on providing virtual characters in 3D-worlds with conversational capabilities. The NICE fairy-tale game [9] and the Mission Rehearsal Exercise System [10] are some of the most sophisticated resulting prototypes. Moreover, several research projects develop stand-alone embodied conversational agents (ECA) which are not part of a virtual game, for example the Companion project [4], Justina [12], Max [14] or relational agents projects [3]. Nevertheless, a lot of research still needs to be done. Especially the processing of unrestricted natural language input and the integration of

¹ *Lifeline*, released by SCEI and Konami, is an example of a game which allows for spoken commands.

human-like dialogue capabilities enhance the naturalness of the NPCs. These capabilities may include e.g. the understanding of implicatures, the ability of producing pragmatically adequate responses as well as the interpretation of the user's utterances and the spatial situation. Additionally, NPC systems always need to preserve the robustness of real-world applications.

The system described in this paper builds on theoretical and experimental insights from linguistic pragmatics, uses novel techniques from computational linguistics and combines them with robust baseline technologies to provide NPCs which possess the necessary intelligence to act and talk in a virtual world. Our NPCs offer various services pertinent to virtual worlds through conversation with game users. Furthermore, the utilisation of detailed knowledge representations enables semantic search and inference.

The paper is organised as follows: The next section describes the virtual world we use as a basis for our system as well as the application scenarios. Section 3 details the human-system interaction corpus which was used for an initial analysis of the domain and the training of our methods. Section 4 presents the underlying knowledge of the NPCs and how it was acquired. Important aspects of dialogue management within a virtual world are explained in section 5: the interpretation of the user's input, the selection of an appropriate reaction, as well as the ability of handling spatially situated dialogue. Finally, section 6 summarises the paper and suggests future work.

2 Application Scenarios

We use the virtual world *Twinity*² as a testbed for our system. *Twinity* provides a digital mirror of an urban part of the real world. At the time of this writing,



Fig. 1. Our two NPCs in interaction with human customers

² <http://www.twinity.com/>, accessed 1 May 2011.

the simulated section of reality already contains 3D models of the cities of Berlin, Singapore, London, Miami and New York and it keeps growing. Users can log into the virtual world, where they can meet other users and communicate with them using the integrated text chat function. They can style their virtual appearance, rent or buy their own flats and decorate them as to their preferences and tastes.

We model two specific characters with different facets, a furniture sales agent and a barkeeper (cf. Fig. 1). The task of the furniture seller is to help users with the interior design of their virtual apartments. Users can buy pieces of furniture and room decoration from the NPC by describing their demands and wishes. The barkeeper, on the other hand, sells cocktails to the users, but moreover, he can entertain his guests by providing trivia-type information about pop stars, movie actors and other celebrities.

We chose these two characters not only because of their value for the *Twinity* application but also for our research goals. Furniture sales conversations are governed by a complex task model involving rich knowledge models about objects to be sold and pragmatic strategies of goal negotiation that guide the conversation. The barkeeper agent on the other hand has to be able to handle less constrained situations. The main role of the barkeeper is to be a conversation companion, which allows us to study and model small talk strategies.

3 Interaction Data

Although there exist some corpora of interactions between humans, mostly children, and virtual agents, e.g. [159], data about human-NPC interactions in 3D-environments are still rather scarce. In order to better understand the nature of this type of interaction, we collected two corpora of human-NPC interactions by means of Wizard-of-Oz (WoZ) experiments.

The interactions of our WoZ-experiments took place in the virtual Twinity environment through a chat-interface by means of written typed language. A furniture-sales scenario was simulated, in which the NPC played the role of an interior designer/furniture saleswoman, whose task was to help the subjects furnishing a living-room. She had to determine the subject's preferences for the different types of furniture, show objects according to these preferences and place them in the room following the subject's instructions. The experimental setting is thoroughly described in [2]. An example dialogue is shown in (1).

The second experiment differed from the first one only in that subjects were made aware of the possibility to do small talk with the NPC and the Wizard was told to initiate small talk when possible.

USR.1: And do we have a little side table for the TV?

NPC.1: I could offer you another small table or a sideboard.

(1) USR.2: Then I'll take a sideboard that's similar to my shelf.

NPC.2: Let me check if we have something like that.

In a third experiment, the NPC played the role of a barkeeper, whose task was to serve a drink and carry out a small talk conversation with the subject. The

subject was told to order a drink and have a conversation with the barkeeper. The latter experiments were carried out in German language.

As a result of the first two experiments, we obtained two mixed-initiative human-NPC corpora consisting of eighteen interactions with one hour of duration each. The first corpus with 23.015 alpha-numeric strings, 4.313 utterances and 3.171 turns was fully annotated. The third experiment resulted in a corpus of 12 dialogues with 1477 utterances. We developed an annotation scheme and a dialogue state representation format with *minimal joint projects* [5] as annotation unit. A minimal joint project has a purpose and is usually realised by an adjacency pair. An adjacency pair consists of two ordered actions carried out by different agents. The first action initiates the joint project, by raising an issue, and the second action completes it. For each joint project we annotated its function, goal, existence of embedded projects, information state at the time of initiating the project, and the initiating and completing actions. The annotation scheme and representation format are presented in detail in [2].

4 Knowledge Representation and Acquisition

4.1 Furniture Ontology

The interior designer has to be able to give guidance and advice. Hence, we need structured knowledge about both decoration, e.g. styles, colours, materials etc. and the catalogue of furniture items available to the user. For this purpose we constructed a detailed ontology specified in the OWL 1.1 language. Currently the ontology consists of 975 classes, 54 properties, 327 instances and 1712 facts (property and class assertions). The ontology contains general concepts (e.g. Room, Object), information about furniture objects (e.g. Sofa, Table) and particular catalogue items from the virtual environment (e.g. Sofa_Isadora). Each item is characterised by several properties, such as *hasStyle*, *madeOf*.

Among concepts modelled in the ontology colour and style are particularly interesting. All instances of the general class *Colour* are defined through properties *hue*, *saturation* and *value* (according to the HSV colour model) and *relative luminance*. Luminance values allow us to absolutely order colours according to their paleness. This enables us to answer comparative queries, e.g. a sofa in a paler or darker colour than the one currently shown. The HSV values allow to answer queries about bright (100% saturation) or dull (unsaturated) colours. Abstract classes, e.g. *Pale Colours*, *Dark Colours* are defined through numeric intervals for the HSV or luminance values. Reasoners can then identify all instances that belong to an abstract class and classify them accordingly.

Some further information about colours is specified, for example, feelings or abstract elements that a particular colour is associated with, and the temperature of a colour (warm vs. cold). This information allows us to find adequate objects for unspecific wishes such as *I want a calm atmosphere*. In these cases, reasoning can be applied to find the colour that can achieve the desired effect and use it in the query to retrieve appropriate objects, without asking the user to directly specify a colour preference.

Our style modelling accounts for the fact that users might have different knowledge degrees about architectural styles. A user with little knowledge about styles may ask for something modern or antique, whereas an expert can ask for a Modern Classic or Art Deco furniture piece. For this purpose styles like *ArtDeco* or *ArtNouveau* can be classified into more coarse-grained categories, such as *Traditional* or *Modern*. These more general classes were defined in our ontology based on numeric (year) intervals for the value of the *stylePeriodBeginning* and *stylePeriodEnd* properties, e.g. *Retro* = 1950-1980.

4.2 Information Extraction and Data Merging

Knowledge bases that are large enough to satisfy user's information requests are very laborious to create. Therefore, we follow the decentralised creation of knowledge bases used in the Semantic Web for building our agents' knowledge base. The knowledge base is in particular important for the barkeeper scenario, where the NPC has to be able to interpret user statements and queries about the world. We created a biographical ontology, the "gossip ontology", defining biographical and career-specific concepts for people. This ontology is accompanied by a huge data resource of celebrities by combining i) existing Semantic Web resources, ii) Semantic Web resources which have been created from semi-structured textual data in the Web and iii) with relation information extracted from free natural language texts [19], [18]. This resource covers nearly 600,000 persons and relations between them like family relationships, marriages and professional relations.

When combining different knowledge sources with overlapping domains, it is crucial to solve the object identity resolution task [7]. This involves finding records in both data sources that refer to the same entities in the world. The resources we use have overlapping concepts and instances for people, groups and locations. Instances were merged by applying a set of heuristics based on common sense as well as on cultural-specific knowledge. For example, we assumed that two instances of type *Person* belong to the same entity when they share at least some features such as their name descriptions, birthdays or birthplaces. However, it occurs in many cases that some important features are unavailable, or that their property values do not match exactly each other. In such cases, we decided to soften the matching constraints. For instance, we have to utilise a normalisation rewriting grammar for names in order to cope with spelling variants or we have to make use of the existence of common relatives.

5 Dialogue Management

The core of the dialogue system is the dialogue manager whose main tasks are the interpretation of the input from the user as well as the selection of an appropriate reaction. Interpretation focuses on the recognition of the dialogue acts which the user performs by his utterances. Due to the mixed-initiative approach used in the agent, the dialogue system is also responsible for the control of the system initiative on the basis of dialogue context.

5.1 Dialogue Act Recognition Using Syntactic and Semantic Relations

Dialogue Acts (DAs) represent the functional level of an utterance, such as a greeting, a request or a statement. Dialogue acts are verbal or nonverbal actions that incorporate participant's intentions.

The dialogue act recognition in the described system is a hybrid component, which uses patterns, statistical methods and rules to detect the appropriate intention belonging to the input. Patterns contain assignments of regular expressions matching the input to a special dialogue act. This is useful for highly predictable, idiosyncratic or one word input such as "hi" or "you are welcome". The rules and the statistical model use a cue-based method for dialogue act classification with various features from multi-level knowledge sources. Features include information extracted from the incoming utterance as well as information about the previous dialogue. In contrast to existing systems using bag-of words representations of the utterances [6], lexical features, i.e. words, single markers such as punctuation [7] or combinations of various features [16], the results of our interpretation component are based on syntactic relations and a minimal dialogue context [13]. Relations are extracted from the utterance through a predicate argument analysis. Context features are: the last preceding dialogue act, equality between the last preceding topic and the actual topic, and sentence mood. Syntactic relation features are: syntactic predicate class, arguments, and negation.

The rules are generalised assignments of feature combinations to dialogue acts. The rules are learned from the same feature lists which serve as input for the statistical model. Learning is supervised since we decided to name some feature values explicitly (e.g. personal pronouns) whereas others are treated anonymously. Rules contain conditions regarding the existence or non-existence of elements as well as the equality of element's values and are organised according to their specificity. The following example shows the rule covering questions such as "Can we add a table?" in XML.

```
(2) <rule pred="add" subj="we" dobj="+">
    <da>r_ar</da>
    </meaning>
  </rule>
```

The statistical model is generated by a Bayesian classifier on the basis of the corpus annotated with dialogue acts and relational information. The model is integrated into the interpretation component and deals with input which does not match a pattern or a rule.

The dialogue act recognition was evaluated using 10-folded cross validation with the annotated corpus data described in section 3.

| Method | Accuracy |
|--------|----------|
| AODEsr | 68,7% |
| Rules | 49,5% |

The table presents the results of the dialogue act classification methods. The Bayesian Classifier is the most reliable method with 68,7% accuracy. The best result we could achieve with the rules is 49,5% accuracy without the information about the preceding dialogue act. If the preceding dialogue act is included in the classification process the accuracy of the rules drops to 34,4%.

5.2 Question Answering

Interaction data show that users ask NPCs about all sorts of things, as for instance their personal information and their doing, the surrounding, available choices in selling situations, or the state of affairs in the world. Question-answering (QA) technology forms a central pillar for handling many of these information-seeking requests. Following a general design paradigm for building our system, we use detailed knowledge representations for solving this task (cf. section 3). On top of this knowledge base, we built a question answering module, which allows users to access information in a smooth natural language dialog.

The QA module is embedded into our input processing pipeline: each user input is first linguistically analysed and interpreted with respect to the current dialog context. The result of the input interpretation is a dialogue act and a more fine-grained semantic representation of the users input in case of information-seeking requests. This question semantics is turned into a query to the knowledge base. The query results are then turned into an abstract representation of the answer and spelt out by a natural-language generator [1]. Our module is able to provide the QA functionality in a smooth and connected dialog. A dialog memory and the dialog state allow the module to resolve pronouns from previous entity mentions as well as to pose and react to clarification questions in case of a possible misunderstanding.

In order to achieve robustness and accuracy for question processing, we take a hybrid approach by combining two strategies. One is a fuzzy pattern matching algorithm which utilises regular lexico-syntactic patterns based on surface strings and recognised named entities, while another makes use of dependency tree structures as patterns. The lexico-syntactic patterns are very robust and not dependent on the performance of a parser. However, the enhancement of the pattern set with the dependency trees allowed us to reduce the 1067 lexico-syntactic patterns to 212 dependency tree patterns, with almost the same linguistic coverage. Thus the utilisation of syntactic parsing results eases maintenance of the pattern set in a significant way.

While tree-based rules help to abstract from surface variations and thus to reduce the number of rules that have to be coded, relevant expressions for the domain still have to be found and formalised for a specific domain. In order to discover all the different means to express a certain fact or question, we drastically extended our initial pattern base by using crowd-sourcing methods in a further effort. We built a platform for acquiring paraphrases of seed questions for biographical facts from multiple human annotators. The seed questions are associated with their semantic arguments and functions. As before, the users input is mapped against this set of paraphrases using fuzzy pattern matching.

The resulting resource can be both used for deriving further modular pieces of expression for a compositional input analysis or it can serve as a gold resource for pattern acquisition and system evaluation.

5.3 Generation of Optimal Answers

A problem often faced by recommender systems is that there is no product in the catalogue exactly matching the user's requirements, that is, the user's query is over-constrained. In such situations recommender systems typically inform the user about the failure to retrieve a valid match and provide some kind of cooperative response. In our two corpora for the sales scenario, we find several types of cooperative responses to retrieval failures; see e.g. (4).

(4) *USR: Let me see a modern one ... If it's possible a yellow one, please.*

SYS: If you would like something modern it is going to be in white or grey. But I can offer you a yellow vintage sideboard.

A pre-requisite for the generation of cooperative answers to retrieval failures is a judgement about how adequate the available alternatives are with respect to the user's preferences. Preference statements are qualitative in nature and in many cases they do not reveal the relative strength of customer's preferences over the different attributes and values. For example, the preference statement '*I want a purple leather sofa*' translates into the attribute–value matrix [TYPE = sofa, COLOUR = purple, MATERIAL = leather].

In decision theory, preferences are represented by utility functions which map the possible outcomes of decisions, in our case the objects of the catalogue, to real values. As is often done in applied decision theory (11), we make the simplifying assumption that the customer's preferences can be represented by an *additive multi-attribute utility function*. This means that each database object a can be identified with a sequence of attribute–values $\langle a_1, \dots, a_n \rangle$ such that the customer's utility function F can be decomposed into the sum of his preferences over the different attributes which in turn can be represented by a non-negative real valued function F_i for the i 'th attribute:

$$F(a) = F_1(a_1) + F_2(a_2) + \dots + F_n(a_n). \quad (1)$$

The utility function F can be further constrained by dividing the attributes into hard and soft attributes. For example, when the customer states that he wants a *purple leather sofa*, we can assume that [TYPE = sofa] is a hard constraint, and that COLOUR and MATERIAL are soft attributes. Hence, searching for optimal alternatives is equivalent to a constraint optimisation problem for which a database object a has to be found which satisfies all hard constraints and optimises $F(a)$, where F is a sum of the utilities $F_i(a_i)$ for soft attributes i . The main problem is to optimise $F(a)$ without actually knowing the objective function F .

This can be solved by assuming that the domains of values of the different attributes have the geometrical structure of *conceptual spaces* as argued for by

Gärdenfors [8]. A conceptual space consists of geometrical representations corresponding to the different quality dimensions of the concept. The customer's preference statements define a *target* t in a conceptual space. d_i measures the distance between two values for attribute i . The goal is to minimise the distance to the target value, so that $d_i(t, a_i) < d_i(t, a'_i) \Rightarrow F_i(a_i) > F_i(a'_i)$. We can then order all values of the i 'th dimension according to increasing distance. This allows us to retrieve Pareto-optimal objects, which are better than any other object in at least one dimension. Whatever the strength of the different preferences are, there is at least one object in the retrieval set that optimally fulfils them.

Let's look at an example. Consider the target [TYPE = sofa, COLOUR = purple, MATERIAL = leather] and the following catalogue of items:

- (5) (a) [Sofa_Alatea: COLOUR = red, MATERIAL = fabric],
- (b) [Sofa_Nadia: COLOUR = black, MATERIAL = leather],
- (c) [Sofa_Isadora: COLOUR = amethyst, MATERIAL = fabric]

Figure 2 shows the catalogue items with respect to the target in the conceptual space. The points (3,0) and (1,1) correspond to Sofa_Nadia(b) and Sofa_Isadora(c), respectively. They dominate the other four sofas and are therefore best candidates. If we consider the properties of these objects, it is apparent that Sofa_Nadia is the only sofa made of leather, the requested material, and Sofa_Isadora, though made of fabric, is of colour amethyst, a shade of purple and the closest to the requested colour.

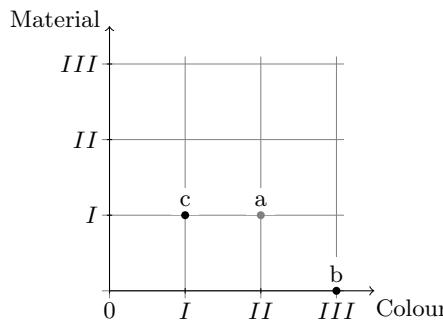


Fig. 2. Geometric representation of the search space for optimal candidates

5.4 Spatial Model and Spatial Expressions

Human perception and cognition is spatially grounded. Users move in space and refer to space, also when being in a virtual reality. In order to give our NPCs an understanding of the surrounding space, we use a three-layered spatial representation introduced for natural-language dialogue components in mobile robots [20]. Following this approach, spatial information is represented in the following layers: (1) *Metrical representation*: Rooms and physical objects are represented by their coordinates. (2) *Topological representation*: Spatial objects

(rooms and items) are topologically ordered. The topological model represents which objects contain which other objects, which objects are next to each other, etc. (3) *Conceptual representation*: Objects from the ontological representation are connected to object representations in the knowledge base, which provides class information, specific properties and information about similar concepts.

The metrical representation is fed to our system by the game engine. For each spatial object (room or item) in the game, we create an RDF representation in our dynamic knowledge base, which represents its spatial position and dimensions as well as basic object identifiers. We mimic human perception in that we limit the perceived objects to a certain radius and to the view angle. The resulting metrical representation is constantly topologically ordered, and the topological relations are as well added to our dynamic knowledge base. Finally, the dynamic item representations are linked with the ontological models by mapping the item identifiers provided by the game to the classes modelled in our rich furniture ontology. This link is established in form of a type assertion about the dynamic item representation. The employed reasoner will then ensure that all supertypes and properties associated with the class will be available for the item.

Spatially referring expressions are then generated and resolved with respect to the topological and conceptual spatial representations. Generation and resolution are realised similarly; for the sake of brevity, we only explain the generation part. In order to generate a spatially referring expression, we need an anchor, from which the spatial expression ought to be understood, and a reference, i.e. the object for which the expression should be generated. In order to establish the referring expression, we search for a shortest path between anchor and reference within the topological model. The natural-language reference is then generated by traversing the path from anchor to reference. Each node and edge on the path is realised verbally by looking up the names of nodes from the corresponding classes in the conceptual model and by mapping topological relations to language templates (usually prepositional constructions) for edges. In case of possible ambiguities arising from a node fragment, additional distinguishing properties such as the colour may be used. Finally, the node and edge fragments are concatenated. The generated spatial expression neither contains too less nor too much information and shall allow the addressee to dereference the expression unambiguously.

6 Conclusion and Future Work

This paper describes a system for conversational agents which are used for non-player characters in a virtual world. The system makes use of sophisticated, novel methods from computational linguistics on the one hand as well as robust state-of-the-art approaches on the other hand. Novel technologies include pragmatically-motivated modules for dialogue management and input interpretation as well as the processing of situated dialogue and knowledge acquisition and organisation. Due to this combination we can provide NPC agents for the

conversational application areas of sales assistance and information provision. The paper describes the main technologies used in the dialogue system as well as the used corpus and knowledge sources.

As the system is still work in progress, evaluation results can only be given for some units of the system. However, we plan to perform a user evaluation of the complete system with regard to the two scenarios in the near future.

Other future work includes the further personalisation of the agents, mainly through the handling of personal opinions and opinion mining. In a first step, the agent will be able to browse a huge database of user's reviews, extract the predominant opinion and provide a summarisation to the user. The next step will include on-line extraction of the user's opinion from the input.

Acknowledgement. The research reported in this paper is supported from the project KomParse, funded by the ProFIT programme of the Federal State of Berlin, cofunded by the EFRE programme of the European Union. The research work in the areas of information extraction and question answering is additionally supported through a grant to the project TAKE, funded by the German Ministry for Education and Research (BMBF, FKZ: 01IW08003) and the German DFG Cluster of Excellence on Multimodal Computing and Interaction". Many thanks go to the supporting company Metaversum.

References

1. Adolphs, P., Cheng, X., Klüwer, T., Uszkoreit, H., Xu, F.: Question answering biographic information and social network powered by the semantic web. In: Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., Tapias, D. (eds.) Proceedings of the 7th International Conference on Language Resources and Evaluation, European Language Resources Association, ELRA (May 2010)
2. Bertomeu, N., Benz, A.: Annotation of joint projects and information states in human-NPC dialogues. In: Proceedings of the First International Conference on Corpus Linguistics CILC 2009, pp. 723–740 (2009)
3. Bickmore, T., Cassell, J.: Social Dialogue with Embodied Conversational Agents. In: van Kuppevelt, J., Dybkjaer, L., Bernsen, N.O. (eds.) Advances in Natural Multimodal Dialogue Systems, Text, Speech and Language Technology. ch.2, vol. 30, pp. 23–54. Springer, Heidelberg (2005)
4. Cavazza, M., de la Cámara, R.S., Turunen, M., Gil, J.R.N., Hakulinen, J., Crook, N., Field, D.: 'How was your day?': an affective companion ECA prototype. In: Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGDIAL 2010, pp. 277–280 (2010)
5. Clark, H.H.: Using Language. Cambridge University Press, Cambridge (1996)
6. Crook, N., Granell, R., Pulman, S.G.: Unsupervised classification of dialogue acts using a dirichlet process mixture model. In: Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 341–348 (2009)
7. Elmagarmid, A., Panagiotis, I., Verykios, V.: Duplicate record detection: A survey. IEEE Transactions on Knowledge and Data Engineering (TKDE) 19(1) (2007)
8. Gärdenfors, P.: Conceptual Spaces: The Geometry of Thought. The MIT Press, Cambridge (2009)

9. Gustafson, J., Boye, J., Fredriksson, M., Johannesson, L., Königsmann, J.: Providing computer game characters with conversational abilities. In: Panayiotopoulos, T., Gratch, J., Aylett, R.S., Ballin, D., Olivier, P., Rist, T. (eds.) IVA 2005. LNCS (LNAI), vol. 3661, pp. 37–51. Springer, Heidelberg (2005)
10. Hill, A.W., Gratch, J., Marsella, S., Rickel, J., Swartout, W., Traum, D.: Virtual humans in the mission rehearsal exercise system. KI Embodied Conversational Agents 17, 32–38 (2003)
11. Keeney, R., Raiffa, H.: Decisions with Multiple Objectives - Preferences and Value Tradeoffs. Cambridge University Press, Cambridge (1993)
12. Kenny, P.G., Parsons, T.D., Gratch, J., Rizzo, A.A.: Evaluation of Justina: A Virtual Patient with PTSD. In: Prendinger, H., Lester, J.C., Ishizuka, M. (eds.) IVA 2008. LNCS (LNAI), vol. 5208, pp. 394–408. Springer, Heidelberg (2008)
13. Klüwer, T., Uszkoreit, H., Xu, F.: Using syntactic and semantic based relations for dialogue act recognition. In: Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010). Association for Computational Linguistics, o.A (2010)
14. Kopp, S., Gesellensetter, L., Krämer, N., Wachsmuth, I.: A conversational agent as museum guide – design and evaluation of a real-world application. In: Panayiotopoulos, T., Gratch, J., Aylett, R.S., Ballin, D., Olivier, P., Rist, T. (eds.) IVA 2005. LNCS (LNAI), vol. 3661, pp. 329–343. Springer, Heidelberg (2005)
15. Narayanan, S., Potamianos, A.: Creating conversational interfaces for children. IEEE Transactions on Speech and Audio Processing 10, 65–78 (2002)
16. Stolcke, A., Ries, K., Coccato, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Van, C., dykema Marie Metteer, E.: Dialogue act modeling for automatic tagging and recognition of conversational speech. Computational Linguistics 26, 339–373 (2000)
17. Verbree, A., Rienks, R., Heylen, D.: Dialogue-act tagging using smart feature selection: results on multiple corpora. In: Raorke, B. (ed.) First International IEEE Workshop on Spoken Language Technology SLT 2006. IEEE Computer Society, Palm Beach (2006)
18. Xu, F., Uszkoreit, H., Krause, S., Li, H.: Boosting relation extraction with limited closed-world knowledge. In: Proceedings of the 23rd International Conference on Computational Linguistics. Association for Computational Linguistics, o.A. (2010)
19. Xu, F., Uszkoreit, H., Li, H.: A seed-driven bottom-up machine learning framework for extracting relations of various complexity. In: Proceedings of ACL 2007, 45th Annual Meeting of the Association for Computational Linguistics. Czech Republic, Prague (June 2007)
20. Zender, H., Kruijff, G.J.: Multi-Layered conceptual spatial mapping for autonomous mobile robots. In: Schultheis, H., Barkowsky, T., Kuipers, B., Hommel, B. (eds.) Papers from the AAAI Spring Symposium on Control Mechanisms for Spatial Knowledge Processing in Cognitive / Intelligent Systems. Papers from the AAAI Spring Symposium, pp. 62–66. AAAI Press, Menlo Park (2007)

Dependency Graphs as a Generic Interface between Parsers and Relation Extraction Rule Learning

Peter Adolphs, Feiyu Xu, Hong Li, and Hans Uszkoreit

DFKI, LT-Lab
Alt-Moabit 91c, 10559 Berlin, Germany
`{peter.adolphs,feiyu,lihong,uszkoreit}@dfki.de`
<http://www.dfki.de/lt/>

Abstract. In this paper, we propose to use dependency graphs rather than trees as the interface between a parser and the rule acquisition module of a relation extraction (RE) system. Dependency graphs are much more expressive than trees and can easily be adapted to the output representations of various parsers, in particular those with richer semantics. Our approach is built on top of an existing minimally supervised machine learning system for relation extraction. We extend its original tree-based interface to a graph-based representation. In our experiments, we make use of two different dependency parsers and a deep HPSG parser. As expected, switching to a graph representation for the parsers outputting dependency trees does not have any impact on the RE results. But using the graph-based representation for the extraction with deep HPSG analyses improves both recall and *f*-score of the RE and enables the system to extract more relation instances of higher arity. Furthermore, we also compare the performance among these parsers with respect to their contribution to the RE task. In general, the robust dependency parsers are good in recall. However, the fine-grained deep syntactic parsing wins when it comes to precision.

1 Introduction

Information extraction (IE) employs various language analysis technologies. The demand for depth of the linguistic analysis is almost parallel to the complexity of the IE task. Shallow components such as part-of-speech tagging and phrase recognition often provide sufficient information for the named entity recognition task. However, for tasks such as relation extraction (RE), the desired setup would be one in which the linguistic parser can provide information about dependencies among linguistic chunks, e.g., grammatical functions or predicate argument structures and the IE system only has to provide corresponding task- or domain-specific interpretation of the linguistic relations and their arguments.

In recent years, parsing systems have achieved great progress with respect to efficiency and robustness (e.g., [4,8,10,23]). In particular, the availability of

large-scale treebanks has given rise to many statistical constituent-based or dependency grammars (e.g., [10]). Meanwhile, the existing hand-written grammars such as Head-driven Phrase Structure Grammars (HPSG) [14] also benefit from statistical parse disambiguation models trained on treebanks for their parse selection and domain adaptation (e.g., [6][8]). Therefore, more and more IE systems applied large-scale generic linguistic parsers for their RE tasks (e.g., [11][12]).

An important research area of IE is to develop methods that can learn RE rules automatically from parsing results for a new domain [13]. The aim is to automatically discover rules that map the appropriate phrases in the parsing results to the semantic roles specified in the target relation. In the previous approaches [5][15][16][17][22], the linguistic representations allowed in RE rules are much less expressive than the parsing results. This leads to the loss of many syntactic or semantic structures among the linguistic arguments, which might be essential indications of the mentioning of the target relations.

Our research builds upon an existing system for minimally supervised relation extraction – the DARE system [20][21]. The DARE system presents a recursive rule representation which supports bottom-up rule learning from dependency parsers for relations with various complexity. The rule learning is embedded in a minimally supervised bootstrapping framework. DARE allows dependency trees with their full expressiveness as interface between the parsers and the rule learning. However, their tree-based representation does not fully support semantic representations containing graph-based predicate argument structures. In this paper, we propose to use dependency graphs as an interface between parsers and the rule learning system, since dependency graphs are expressive enough to be adapted to various syntactic and semantic representations provided by state-of-the-art large-scale parsers. In order to realise this new interface, we extend the DARE system by modifying its rule learning algorithm and its rule representation. In our experiments, we use three parser systems: two dependency parsers – MINIPAR [8] and Stanford Parser [10] – and the HPSG parser PET [11] with a broad-coverage deep linguistic grammar, the English Resource Grammar (ERG) [4]. The two dependency parsers deliver grammatical functions, while the HPSG parser provides graph-based predicate argument structures where an argument can be shared by several predicates. The evaluation results tell us that dependency graphs are very useful to keep as much linguistic information as possible for the rule learning. Without dependency graphs, many linguistic phenomena cannot be covered by the IE systems properly. Thus, it helps to learn more rules, yielding in better recall, in particular, for the HPSG semantic output.

The remainder of the paper is organised as follows: Section 2 discusses related work; Section 3 introduces the DARE system architecture and its rule representation; Section 4 explains the three parsers with their output representations; Section 5 describes the new graph-based DARE rule representation and the corresponding new rule discovery method; In Section 6 various experiments are conducted to compare the dependency graphs as interface with tree-based interface and a systematic quality and error analysis is presented too; Section 7 closes off with a conclusion and discusses the future directions.

2 Related Work

Many minimally supervised or unsupervised approaches targeted to learning rules for extracting complex relations develop their rule representations on top of dependency trees [5,15,16,17,22]. All these representations are less expressive than the dependency trees from which they learn their rules. The learned linguistic patterns for the RE rules are mostly restricted to trees with limited depth and branching factor, e.g., single paths [16] or flat subject-verb-object constructions (binary trees with depth one) [22]. Furthermore, they are only verb-centered. Thus, with this strong constraint, they cannot deal with relation mentions that are expressed in complex nominal compounds or nominalization constructions. Moreover, in comparison to DARE [20], patterns acquired by all these models do not specify the mapping between the linguistic arguments and the relation-specific semantic roles. The DARE rule representation allows the full expressiveness of the dependency trees. But, like other approaches, it cannot deal with semantic representations allowing graph-structures.

For the RE tasks in the biomedical domain, [11,12] give a detailed evaluation of various parsers. They discover that more semantic-oriented parsing representation such as dependency or predicate-argument structures achieve better overall performance than pure syntactic representation, e.g., phrase structures.

3 DARE: Minimally Supervised Rule Learning for Relation Extraction

DARE [20,21] is a minimally supervised machine learning system for RE on free texts, consisting of two parts: 1) rule learning and 2) relation extraction (RE). Rule learning and RE feed each other in a bootstrapping framework. The bootstrapping starts from so-called "semantic seeds", which is a small set of instances of the target relation. The rules are extracted from sentences annotated with semantic entity types and parsing results, e.g., dependency structures, which match with the seeds. RE applies acquired rules to texts in order to discover more relation instances, which in turn are employed as seed for further iterations. The core system architecture of DARE is depicted in Fig. 1. The entire bootstrapping stops when no new rules or new instances can be detected. Relying entirely

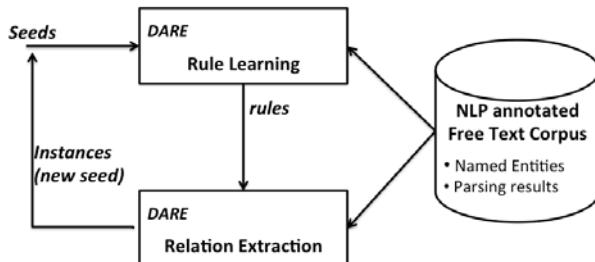


Fig. 1. DARE core architecture

on semantic seeds as domain knowledge, DARE can accommodate new relation types and domains with minimal effort.

DARE can handle target relations of varying arity through a compositional and recursive rule representation and a bottom-up rule discovery strategy. A DARE rule for an n -ary relation can be composed of rules for its projections, namely, rules that extract a subset of the n arguments. Furthermore, it defines explicitly the semantic roles of linguistic arguments for the target relation. The following examples illustrate the DARE rule and its extraction strategy. *Example 1* is a relation instance of the target relation from [20] concerning Prize awarding event, which contains four arguments: *Winner*, *Prize_Name*, *Prize_Area* and *Year*. *Example 2* refers to an event mentioned in *Example 1*.

Example 1. $\langle \text{Mohamed ElBaradei}, \text{Nobel}, \text{Peace}, 2005 \rangle$.

Example 2. Mohamed ElBaradei, won the 2005 Nobel Prize for Peace on Friday.

Given *Example 1* as a seed, *Example 1* matches with the sentence in *Example 2* and DARE assigns the semantic roles known in the seed to the matched linguistic arguments in *Example 2*. Fig. 2 is a simplified dependency tree of *Example 2* with named entity annotations and corresponding semantic role labelling after the match with the seed. DARE utilises a bottom-up rule discovery strategy to extract rules from such semantic role labelled dependency trees.

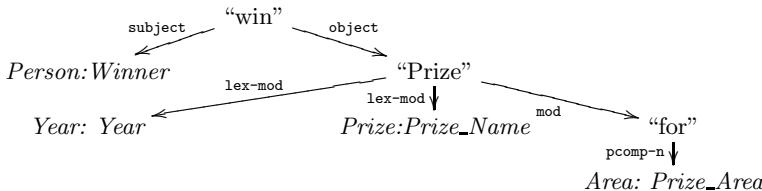


Fig. 2. Dependency tree of *Example 2* matched with the seed

Rule name :: winner-prize-area-year_1

Rule body ::

| | |
|-----------|---|
| head | $\begin{bmatrix} \text{pos} & \text{verb} \\ \text{mode} & \text{active} \\ \text{lex-form} & \text{"win"} \end{bmatrix}$ |
| daughters | $< \begin{bmatrix} \text{subject} & \begin{bmatrix} \text{head} & \boxed{1} \text{Person} \end{bmatrix} \end{bmatrix},$ $\begin{bmatrix} \text{object} & \begin{bmatrix} \text{rule} & \text{year_prize_area_1 ::} \\ & < \boxed{1} \text{Year}, \boxed{2} \text{Prize_Name}, \\ & \boxed{3} \text{Prize_Area} > \end{bmatrix} \end{bmatrix} >$ |
| Output | $:: < \boxed{1} \text{Winner}, \boxed{2} \text{Prize_Name}, \boxed{3} \text{Prize_Area}, \boxed{4} \text{Year} >$ |

Fig. 3. DARE extraction rule

From the tree in Fig. 2, DARE learns three rules in a bottom-up manner, each step with a one tree depth. The first rule is extracted from the subtree dominated by the preposition “for”, extracting the argument *Prize_Area* (*Area*), while the second rule makes use of the subtree dominated by the noun “Prize”, extracting the arguments *Year* (*Year*) and *Prize_Name* (*Prize*), and calling the first rule for the argument *Prize_Area* (*Area*). The third rule “winner_prize_area_year_1” is depicted in Fig. 3. The value of *Rule body* is extracted from the dependency tree. In “winner_prize_area_year_1”, the subject value *Person* fills the semantic role *Winner*. The object value calls internally the second rule called “year_prize_area_1”, which handles the other arguments *Year* (*Year*), *Prize_Name* (*Prize*) and *Prize_Area* (*Area*).

4 Parsers

MINIPAR [8] is a broad-coverage parser for English, implementing a constraint-based parsing algorithm which is reminiscent of chart parsing with rewrite rules¹. Parse results are available in a dependency tree format. They can also be partial; in this case, the analysis for the sentence itself is a parse forest, consisting of several unconnected dependency trees.

One of the parsing options of the **Stanford Parser**² is an unlexicalised PCFG [7], which outputs phrase-structure analyses. These can be converted to labelled dependency representations [9], that are more useful for applications such as IE. Dependency labels denote grammatical functions. The conversion tool can further simplify the dependency structures by collapsing function words, most prominently prepositions, and their associated dependencies, yielding simpler graph structures where content words are directly related to each other via more specialised relation types. Furthermore, dependencies of the head of a conjunction can be optionally propagated to all coordinated elements. Both the collapsing and propagation of dependencies may lead to cyclic structures, i.e. to general dependency graphs.

The **English Resource Grammar (ERG)** is a broad-coverage grammar for English [4], written in the framework of Head-Driven Phrase Structure Grammar (HPSG) [14]. We use the ERG in combination with the efficient HPSG parser PET [1]. Both the ERG and PET are available as Open Source components³. Semantics utilised in the ERG is expressed in the Minimal Recursion Semantics (MRS) formalism [3], which essentially encodes a predicate-argument structure with generalised quantifiers and underspecified scopes. MRS representations can be converted to ‘Dependency MRS’ (DMRS; [2]), and vice versa, a simplified representation that resembles classical dependency structures. In order to gain classical token-to-token dependencies, we further simplify these representations by converting nodes representing compounds to edges and merging overlapping nodes which decompose the meaning of a particular word. The ERG analyses

¹ <http://webdocs.cs.ualberta.ca/~lindek/minipar.htm>, accessed 26 April 2011.

² <http://nlp.stanford.edu/software/lex-parser.shtml>, accessed 26 April 2011.

³ <http://www.delph-in.net/>, 26 April 2011.

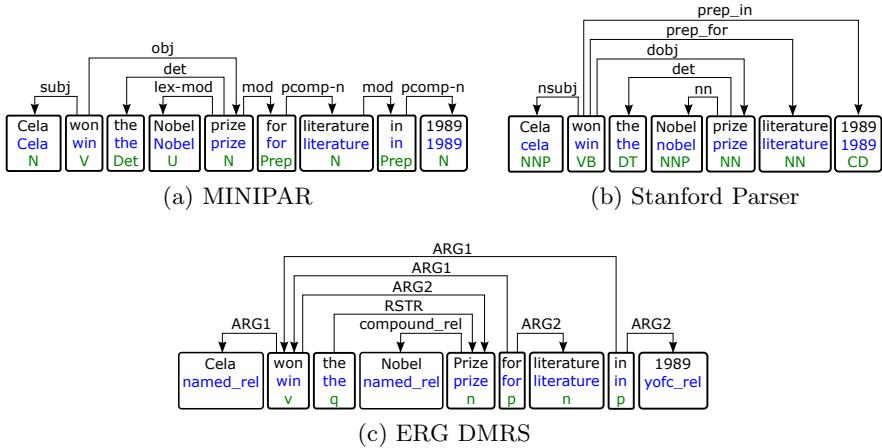


Fig. 4. Example analyses for the 3 parsers

certain constructions systematically differently than the other two parsers. For instance, modification is modeled as modifiers selecting for the heads they modify. Thus, heads with multiple modifiers will have multiple parents. The MRS also provides linguistically more adequate representations of phenomena such as control structures, where phrases are arguments of different predicates at the same time and thus the corresponding nodes have multiple parents. For instance, in the sentence “*John promises us to come*”, “*John*” is the subject of “*promise*” as well as the subject of “*come*”. In all these cases, the resulting dependency representation will therefore usually be a genuine graph rather than a tree.

Fig. 4 illustrates some of the systematic differences between the parsers. The prepositions in the Stanford analysis are incorporated into a dependency edge and the corresponding node has been eliminated since edges have been collapsed. The reversed dependency directions for the specification and modification structures in the ERG analysis result in nodes with several incoming edges. The actual analyses are structurally quite similar. The analyses of all parsers pick up the correct nodes as the subject and object of the verb. The only real difference for this particular sentence is how the inherently ambiguous choices of attaching the PPs are resolved: while MINIPAR chooses to attach both PPs low, Stanford Parser and ERG attach the PPs high at the verbal node.

5 Dependency Graph as Interface

The strategy to match tree fragments in the dependency structure is inappropriate in cases where relation argument nodes are connected by paths with reversed directions as in Fig. 4 (c). Though it is possible to identify two tree fragments in the structure connecting either the winner, prize and area combination or the winner, prize and year arguments (the trees being rooted at either the preposition “*for*” or “*in*”, respectively), there is no tree fragment connecting all four

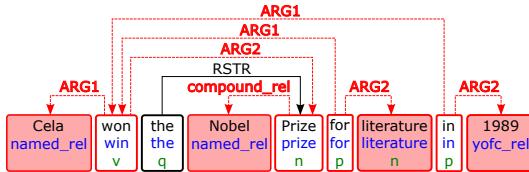


Fig. 5. Graph fragment in an ERG analysis

arguments. Rather than learning tree rules from dependency trees bottom up, we therefore extend the DARE rule learning algorithm to learn graph rules which identify subgraphs in arbitrary graphs. Fig. 5 shows such a subgraph within the original analysis, connecting all relation arguments (which are highlighted). Since graph-based models provide a general framework for representing all kinds of linguistic information, this strategy also promises to facilitate the combination of different parsing methods using a uniform rule representation.

A DARE graph rule has three components:

1. **rule name:** r_i
2. **output:** a set A containing n arguments of the n -ary relation, labelled with their argument roles.
3. **rule body:** a graph $G = (N, E)$ where N is a set of nodes with – possibly underspecified – features to be matched, and E is a set of – possibly labelled – edges connecting these nodes. The elements of A are coindexed with the reference feature of the corresponding argument nodes in N .

As before, the rule learning happens in two steps. Matching subgraphs are first extracted and then generalised to form extraction rules by underspecifying the nodes and introducing place-holders labelled with the role for the argument nodes. The pattern subgraphs are extracted from the dependency graph by the following procedure:

1. For a given n -ary seed $S = (s_1, \dots, s_n)$ and a given dependency graph G , collect the set T of all terminal nodes from G that are instantiated with seed arguments in S .
2. For each acceptable combination of seed argument terminal nodes $C = \{t_1, \dots, t_m\}$ ($m \geq 2$), find a shortest path S_i between t_i and t_{i+1} for $0 < i < m$.
3. For each combination of seed argument terminal nodes C and the corresponding set of shortest paths $S_C = \{S_1, \dots, S_m\}$, extract the corresponding pattern subgraph P_C from G , where the set of nodes is the union of the nodes of S_i and the set of edges is the union of the edges of S_i ($0 < i < m$).

Note that we iterate over all acceptable combinations of argument nodes, where an acceptable combination is one that contains at least two arguments. Further constraints on argument combinations (required arguments) might be desired in order to exclude the extraction of uninformative bits of information. By iterating over all acceptable combinations, we ensure to learn also all projections of the RE rule. Although we do not define an interface for calling RE rules within

other RE rules as in the original DARE rule format, this does not affect the performance as it leads to smaller rule set descriptions but not to an increased recognition capacity if no further rule generalisation means are used.

6 Experiments and Evaluation

6.1 Data and Experiment Setup

For several reasons we decided to adopt for our experiments the freely available Nobel Prize award corpus of [21]. The target relation is the 4-ary prize-winning relation $\langle \text{Winner}, \text{Prize_Name}, \text{Prize_Area} \text{ and } \text{Year} \rangle$. Previous results have shown i) that not every data collection is suited for the minimally supervised approach to RE [20] and ii) that the freely available Nobel Prize award corpus actually has the required properties [19]. The corpus contains 2,864 free text documents from BBC, CNN and NYT, together 143,289 sentences. The total corpus has also a version for evaluation, manually annotated with prize-winning event mentions.

The corpus was automatically preprocessed with named-entity recognition and coreference resolution. Only those sentences are considered for the experiments that can potentially satisfy the following criterion, i.e. that contain at least a person reference and a prize mentioning. The resulting corpus comprises 2,902 potentially relevant sentences for the target relation.

We applied the three parsers described in section 4, namely, MINIPAR 0.5, Stanford Parser 1.6.5, and ERG 1010. The ERG was configured to use the vanilla reading selection model (Redwoods) and a maximum of 1 GB main memory for each sentence. We gained analyses for 2,896 sentences each for MINIPAR and Stanford Parser (99.79% parse coverage), and 2,081 sentences for ERG 1010 (71.71%). The parse coverage for the ERG is lower than the expected 80 – 90%, which is usually observed for texts of similar origins. All parser results were stored in the same dependency graph format.

We performed rule learning and RE on separate subcorpora. Applying the rules to unseen data allows us to judge the quality and reusability of the learned rules. To this end, we split the corpus into two equal-sized parts – the *learning* and the *extraction* corpus. For each parser, we started the bootstrapping process with the same seeds on the learning corpus to learn RE rules. In a second step, we applied the learned rules to each sentence in the extraction corpus to extract relation instances. In a third step, compatible relation instances that are learned from the same sentence are merged, leading to the most specific relation instances for the sentence, that is, only relation instances of higher arity are considered in the evaluation.

Many factors may shape the relation extraction quality of RE rules learned in a bootstrapping framework. The domain and data properties and the selection of semantic seeds play an important role of the overall performance of the RE system [19]. In order to eliminate the possibility that the evaluation results are only due to a luckily picked semantic seed, we conducted experiments with different seeds: a) exactly one semantic seed ($\langle \text{Ahmed Zewail}, \text{Nobel}, \text{Chemistry}, 1999 \rangle$), b) 99 randomly chosen semantic seeds, c) all Nobel prize winning events

Table 1. Results for the full learning / extraction corpora ($\mathcal{O}A$: average arity)

| Nr. Seeds | Parser | (a) Tree Rules | | | | (b) Graph Rules | | | |
|--------------|----------|----------------|--------|--------|----------------|-----------------|--------|--------|----------------|
| | | Prec | Recall | f_1 | $\mathcal{O}A$ | Prec | Recall | f_1 | $\mathcal{O}A$ |
| 1 | MINIPAR | 81.97% | 46.93% | 59.69% | 2.77 | 82.01% | 46.69% | 59.51% | 2.77 |
| | Stanford | 79.42% | 53.78% | 64.13% | 2.83 | 79.48% | 53.78% | 64.16% | 2.84 |
| | ERG | 84.06% | 34.02% | 48.44% | 2.80 | 83.08% | 35.22% | 49.47% | 2.85 |
| 99 | MINIPAR | 81.97% | 46.93% | 59.69% | 2.81 | 82.01% | 46.69% | 59.51% | 2.81 |
| | Stanford | 79.42% | 53.78% | 64.13% | 2.84 | 79.48% | 53.78% | 64.16% | 2.84 |
| | ERG | 84.09% | 34.10% | 48.53% | 2.82 | 82.99% | 35.38% | 49.61% | 2.86 |
| all | MINIPAR | 82.18% | 49.00% | 61.40% | 2.84 | 82.31% | 48.69% | 61.18% | 2.84 |
| | Stanford | 79.58% | 54.26% | 64.53% | 2.88 | 79.67% | 54.26% | 64.56% | 2.88 |
| | ERG | 83.08% | 34.74% | 48.99% | 2.83 | 82.94% | 36.33% | 50.53% | 2.87 |

that happened so far. Using all seeds for rule learning is an interesting endeavour as it allows us to estimate an upper bound for the RE quality that can be achieved with the current preprocessing tools and learning approach.

An event mention extracted from a sentence is considered to be recognised successfully if it is compatible with one of the annotated event mentions available for this sentence. We use standard precision, recall and f_1 -score measures for evaluation. In order to assess one of the strengths of the DARE approach, namely its ability to extract relation instances of higher arity, we also calculate the average arity of extracted relations.

6.2 Evaluation

Table I shows an evaluation of the RE results of the systems on the full extraction corpus, using (a) tree rules and (b) graph rules with the three different seed sets. As expected, switching from the tree-based to the graph-based relation extractor has no substantial impact on the RE performance using MINIPAR and the Stanford Parser. However, it increases both recall and f -score of RE with the ERG. This is also reflected in the average arity of the extracted instances. While the average arity is virtually unchanged for MINIPAR and Stanford Parser, using graph rules with the ERG helps the system to extract more relation instances of higher arity. It means that graph rules are useful for rich semantic representations and can extract more information than tree rules.

Furthermore, the results confirm the observations made in earlier studies where the same RE task is carried out on these data, namely that the choice of seeds does not substantially influence the RE results for this corpus. Even using one semantic seed can be enough to learn all relevant RE rules. RE with the ERG using the full corpora performs worse than with MINIPAR or the Stanford Parser. This is not surprising since coverage of the ERG on the corpus is much lower than for the other parsers. In order to compare the RE results obtained with the HPSG grammar to the results with the two other parsers more closely, we ran a second series of experiments for all parsers on the HPSG-parsable

learning and extraction subcorpora only. In this scenario, all parsers see exactly the same sentences during learning and extraction, levelling the differences due to different paths during bootstrapping. Table 2 shows the results of these experiments with the graph extractor in column (b). Obviously parse coverage is not the only reason for the performance differences in table 1. Though RE with the ERG achieves the best precision scores, using Stanford Parser analyses leads to considerably better recall, which counterweights the lower precision.

A detailed comparison between the RE results obtained with the Stanford Parser and the ERG quickly shows that some of the mismatches were due to systematically different coordination analyses. While MINIPAR and the Stanford Parser anchor an incoming dependency to a coordinated NP at the first conjunct and then link the remaining conjuncts with a special conjunction dependencies, the ERG creates explicit (for words such as “and”) or implicit (covert) conjunction nodes which link the conjuncts and places the incoming dependency at the head conjunction node. Given an RE rule learned from a structure without coordination, this analysis allows MINIPAR and the Stanford Parser to extract the first conjunct in a similar structure with coordination, while the ERG cannot extract anything at the target node with a corresponding rule as the coordination node will not match the argument node in the rule. We therefore further extended the graph extractor to interpret coordination structures during extraction. During rule matching, the extractor may follow any coordination links found in the graph. This strategy is applied for all parsers, and column (c) in table 2 shows that all parsers benefit from this extended extraction strategy. Although this extraction strategy has been a useful step to improve the RE results, it is also apparent that the differences in the coordination analysis are not responsible for the performance differences between the parsers. Given enough learning data, rules for extracting from coordinated structures will be learned with the DARE learning approach.

The remaining possible reasons for the differences between parsers on the same corpus are different grammar coverage, i.e. missing analyses for certain

Table 2. Results for the HPSG-parsable learning / extraction corpora

| Nr. Seeds | Parser | (b) Graph Rules | | | | (c) G. R. + Coord. Extr. | | | |
|--------------|----------|-----------------|--------|--------|-----------------|--------------------------|--------|--------|-----------------|
| | | Prec | Recall | f_1 | $\bar{\Omega}A$ | Prec | Recall | f_1 | $\bar{\Omega}A$ |
| 1 | MINIPAR | 82.36% | 46.54% | 59.48% | 2.79 | 81.95% | 50.27% | 62.32% | 2.82 |
| | Stanford | 80.26% | 53.57% | 64.25% | 2.86 | 80.59% | 55.87% | 65.99% | 2.90 |
| | ERG | 83.08% | 48.52% | 61.26% | 2.85 | 81.58% | 51.48% | 63.13% | 2.90 |
| 99 | MINIPAR | 82.36% | 46.54% | 59.48% | 2.83 | 81.95% | 50.27% | 62.32% | 2.86 |
| | Stanford | 80.26% | 53.57% | 64.25% | 2.87 | 80.72% | 55.87% | 66.04% | 2.90 |
| | ERG | 82.99% | 48.74% | 61.41% | 2.86 | 81.87% | 51.48% | 63.21% | 2.92 |
| all | MINIPAR | 82.44% | 48.74% | 61.26% | 2.87 | 81.72% | 51.37% | 63.09% | 2.89 |
| | Stanford | 80.48% | 54.88% | 65.26% | 2.93 | 80.58% | 57.19% | 66.90% | 2.92 |
| | ERG | 82.94% | 50.05% | 62.43% | 2.87 | 81.85% | 52.80% | 64.19% | 2.91 |

constructions, different strengths of the reading selection models and suitability of the granularity of analysis for the RE task. Since we do not have gold treebanks for the three parsers yet, we cannot systematically assess grammar coverage and reading selection quality. However, we saw a tendency during our qualitative comparison of the RE results with the Stanford Parser and the ERG that the Stanford dependency representation, which provides semantically motivated interpretations and also flatter analyses for some linguistic structures, is beneficial for the RE task.

7 Conclusion and Future Work

In order to faithfully express the semantics of linguistic phenomena such as multiple modifiers of a head word or raising and control constructions, graph structures are needed for representation. We have extended the interface between DARE and parsers from trees to generic dependency graphs, allowing us to deal with the output representations of various parsers, in particular those with rich semantics. Our experiments confirm that the graph-based interface is expressive enough for learning extraction rules exhaustively from linguistic analyses provided by various parsers. As expected, switching to a graph representation for the dependency tree parsers does not have any impact on the RE results. But using the graph-based representation for the extraction with deep HPSG analyses improves both recall and *f*-score of the RE and the arity of the extracted instances is higher, i.e., more information can be detected. During our experiments, we also discover that the Stanford dependency representation is well designed for semantically oriented NLP applications. Last but not least, we have demonstrated the general applicability of a mature deep grammar for English, the ERG, in such a task. It is impressive to see the big steps forward towards real-world applications that the grammar made over the last years.

For the future, we plan an extensive empirical analysis of the parser differences w.r.t. success or failure in the RE task. If we succeed to pin down the advantages of each parser to distinguishing criteria, we can improve the quality of RE by learning RE rules from the merged output of a parser ensemble. Finally, we plan to exploit the rich modelling of important but challenging semantic relations in the ERG such as modality, negation and their scopal interactions by directly operating on the MRS representations.

Acknowledgements. This research was conducted in the context of the DFG Cluster of Excellence on Multimodal Computing and Interaction (M2CI), project KomParse (funded by the ProFIT program of the Federal State of Berlin and the EFRE program of the EU, contract 1014 0149), projects Theseus Alexandria and Alexandria for Media (funded by the German Federal Ministry of Economy and Technology, contract 01 MQ 07 016), and project TAKE (funded by the German Federal Ministry of Education and Research, contract 01IW08003).

References

1. Callmeier, U.: Preprocessing and encoding techniques in PET. In: Oepen, S., Flickinger, D., Tsujii, J., Uszkoreit, H. (eds.) Collaborative Language Engineering. A Case Study in Efficient Grammar-based Processing. CSLI Publications, Stanford (2002)
2. Coopestake, A.: Dependency and (R)MRS (December 2008),
<http://www.cl.cam.ac.uk/~aac10/papers/dmrs.pdf> unpublished Draft (December 9, 2008)
3. Coopestake, A., Flickinger, D., Pollard, C., Sag, I.A.: Minimal recursion semantics: An introduction. Research on Language and Computation 3(4) (2005)
4. Flickinger, D.: On building a more efficient grammar by exploiting types. Natural Language Engineering 6(1) (2000)
5. Greenwood, M., Stevenson, M., Guo, Y., Harkema, H., Roberts, A.: Automatically Acquiring a Linguistically Motivated Genic Interaction Extraction System. In: Proc. of the 4th Learning Language in Logic Workshop (LLL 2005), Bonn, Germany (2005)
6. Hara, T., Miyao, Y., Tsujii, J.: Adapting a probabilistic disambiguation model of an HPSG parser to a new domain. In: Dale, R., Wong, K.-F., Su, J., Kwong, O.Y. (eds.) IJCNLP 2005. LNCS (LNAI), vol. 3651, pp. 199–210. Springer, Heidelberg (2005)
7. Klein, D., Manning, C.D.: Fast exact inference with a factored model for natural language parsing. In: Advances in Neural Information Processing Systems 15 (NIPS 2002), vol. 15. MIT Press, Cambridge (2003)
8. Lin, D.: Dependency-based evaluation of MINIPAR. In: Abeillé, A. (ed.) Treebanks - Building and Using Parsed Corpora. Kluwer Academic Publishers, Dordrecht (2003)
9. de Marneffe, M., MacCartney, B., Manning, C.: Generating typed dependency parses from phrase structure parses. In: Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006), Genoa, Italy (2006)
10. de Marneffe, M., Manning, C.D.: The stanford typed dependencies representation. In: Coling 2008: Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation, Manchester, UK (2008)
11. Miwa, M., Pyysalo, S., Hara, T., Tsujii, J.: A comparative study of syntactic parsers for event extraction. In: Proceedings of the 2010 Workshop on Biomedical Natural Language Processing, Uppsala, Sweden (July 2010)
12. Miyao, Y., Sagae, K., Satre, R., Matsuzaki, T., Tsujii, J.: Evaluating contributions of natural language parsers to protein-protein interaction extraction. Bioinformatics 25 (2009)
13. Muslea, I.: Extraction patterns for information extraction tasks: A survey. In: AAAI Workshop on Machine Learning for Information Extraction, Orlando, Florida (July 1999)
14. Pollard, C.J., Sag, I.A.: Head-Driven Phrase Structure Grammar. University of Chicago Press, Chicago (1994)
15. Stevenson, M., Greenwood, M.A.: Comparing information extraction pattern models. In: Proceedings of the Workshop on Information Extraction Beyond The Document. Association for Computational Linguistics, Sydney, Australia (July 2006)
16. Sudo, K., Sekine, S., Grishman, R.: An improved extraction pattern representation model for automatic IE pattern acquisition. In: Proceedings of ACL 2003 (2003)

17. Sudo, K., Sekine, S., Grishman, R.: Automatic pattern acquisition for japanese information extraction. In: Proceedings of the First International Conference on Human Language Technology Research (HLT 2001), Morristown, NJ, USA (2001)
18. Toutanova, K., Manning, C.D., Flickinger, D., Oepen, S.: Stochastic HPSG parse selection using the Redwoods corpus. *Journal of Research on Language and Computation* 3(1) (2005)
19. Uszkoreit, H., Xu, F., Li, H.: Analysis and improvement of minimally supervised machine learning for relation extraction. In: Horacek, H., Métais, E., Muñoz, R., Wolska, M. (eds.) NLDB 2009. LNCS, vol. 5723, pp. 8–23. Springer, Heidelberg (2010)
20. Xu, F.: Bootstrapping Relation Extraction from Semantic Seeds. Phd-thesis, Saarland University (2007)
21. Xu, F., Uszkoreit, H., Li, H.: A seed-driven bottom-up machine learning framework for extracting relations of various complexity. In: Proceedings of ACL 2007. Czech Republic, Prague (2007)
22. Yangarber, R.: Scenario Customization for Information Extraction. Dissertation, New York University, New York, USA (2001)
23. Zhang, Y.: Robust Deep Linguistic Processing. Phd-thesis, Saarland University, Saarbruecken, Germany (2007)

Evaluation and Comparison Criteria for Approaches to Probabilistic Relational Knowledge Representation*

Christoph Beierle¹, Marc Finthammer¹,
Gabriele Kern-Isberner², and Matthias Thimm²

¹ Dept. of Computer Science, FernUniversität in Hagen, 58084 Hagen, Germany

² Dept. of Computer Science, TU Dortmund, 44221 Dortmund, Germany

Abstract. In the past ten years, the areas of *probabilistic inductive logic programming* and *statistical relational learning* put forth a large collection of approaches to combine relational representations of knowledge with probabilistic reasoning. Here, we develop a series of evaluation and comparison criteria for those approaches and focus on the point of view of knowledge representation and reasoning. These criteria address abstract demands such as language aspects, the relationships to propositional probabilistic and first-order logic, and their treatment of information on individuals. We discuss and illustrate the criteria thoroughly by applying them to several approaches to probabilistic relational knowledge representation, in particular, Bayesian logic programs, Markov logic networks, and three approaches based on the principle of maximum entropy.

1 Introduction

Originally, probabilistic logic was based on propositional logic, using conditionals of the form $(B \mid A)[x]$ to express that *if A, then B with probability x*. In order to exploit its more expressive power, various approaches to combine probabilistic logic with first-order logic have been proposed (see [3][7]) like Bayesian logic programs (BLP) [4, Ch. 10], Markov logic networks (MLN) [5], or relational Bayesian networks [8]. The principle of maximum entropy [14] is used to define the probabilistic relational approaches in [13][2][6].

There are different motivations and objectives for choosing a particular representation for probabilistic relational knowledge. Suppose we want to model situations in a zoo (this scenario is adapted from [4]). There are elephants and keepers, and we want to say something about whether elephants like their keepers. Thus, we want to formalize generic statements like *Generally, elephants like their keepers* or *Elephants like their keepers with a probability of 0.9*. Furthermore, we might want to state information about individuals, e.g., that Fred is an elephant keeper the elephants do not like very much; this might be expressed by *Elephants like Fred only with a probability of 0.3*. There are also situations where it is useful to list all individual elephants and keepers that are in the zoo. Given a knowledge base representing our zoo model, we would like to be able

* This research was partially supported by the DFG (BE 1700/7-2 and KE 1413/2-2).

to use inference methods to answer questions about individuals occurring in the model and relationships among them.

Example 1. To give a concrete example, consider the knowledge base KB with

$$c_1: (\text{likes}(X, Y) \mid \text{elephant}(X), \text{keeper}(Y))[0.9]$$

$$c_2: (\text{likes}(X, \text{fred}) \mid \text{elephant}(X))[0.3] \quad c_3: (\text{likes}(\text{clyde}, \text{fred}))[1.0]$$

where X and Y are variables. There is a general statement (c_1) that represents the probability of elephants liking their keepers, and two more specific statements (c_2 resp. c_3) that model the relationships for some individuals Clyde and Fred.

Note that naive grounding of KB using all possible instantiations yields contradictory information since, for instance, we get both $\text{likes}(\text{clyde}, \text{fred}))[0.3]$ and $\text{likes}(\text{clyde}, \text{fred}))[1]$. Nonetheless, KB makes perfect sense from a commonsense point of view as, for instance, rule c_2 could be treated as an exception to c_1 , inhibiting the instantiation of Y with the constant fred in c_1 .

Despite the variety of different approaches to probabilistic relational knowledge representation, inference, and learning, not much work has been done on systematically comparing them. In [2], a comparison between several statistical relational learning systems is done, with an emphasis on the learning aspects. In [9], a schema for expressivity analysis is proposed and used to show that relational Bayesian networks [8] are at least as expressive as MLNs. While providing access to different modeling and learning approaches, the focus of the software suite ProbCog [10] is its practical use and integration in technical systems. A software platform providing a common interface to a series of probabilistic relational systems and supporting their evaluation and comparison is presented in [18]. In [18], also some meta-level criteria for the evaluation and comparison of different approaches are given. In this paper, we extend the discussion of these criteria, focusing on knowledge representation aspects, apply them to further approaches, and develop a series of new criteria especially with respect to the role of individuals, prototypical elements, and universes. It has to be noted that our investigation of these criteria stays on an abstract level since the objective is not to take specific formalizations into account, but to address desirable properties and interesting features from a general and commonsense perspective. A technical comparison with respect to default reasoning properties of the three approaches employing the principle of maximum entropy can be found in [12].

After briefly recalling the notions of BLP, MLN, and three approaches based on maximum entropy in Sec. 2 our comparison and evaluation criteria are presented along several dimensions, dealing with language aspects (Sec. 3), the relationship to strict and propositional knowledge (Sec. 4), and individuals and universes (Sec. 5). In Sec. 6, we conclude and point out further work.

2 Background: Probabilistic Relational Approaches

Bayesian logic programming combines logic programming and Bayesian networks [7, Ch. 10]. The basic structure for knowledge representation in Bayesian logic programs are *Bayesian clauses* which model probabilistic dependencies between *Bayesian atoms* as in the following BLP corresponding to Ex. 1:

$$\begin{aligned} c_1 &: (\text{likes}(X, Y) \mid \text{elephant}(X), \text{keeper}(Y)) \\ c_2 &: (\text{likes}(X, \text{fred}) \mid \text{elephant}(X)) \qquad \qquad c_3: \text{likes}(\text{clyde}, \text{fred}) \end{aligned}$$

While in Ex. II a probability for each clause is given, expressing a constraint on a satisfying distribution, for each Bayesian clause c , a function cpd_c must be defined, expressing the conditional probability distribution $P(\text{head}(c) \mid \text{body}(c))$ and thus partially describing an underlying probability distribution P . For instance, $\text{cpd}_{c_1}(\text{true}, \text{true}, \text{true}) = 0.9$ would express our subjective belief that $\text{likes}(X, Y)$ is true with probability 0.9 if $\text{elephant}(X)$ and $\text{keeper}(Y)$ are true. In order to aggregate probabilities that arise from applications of different Bayesian clauses with the same head, BLPs make use of *combining rules*. Semantics are given to Bayesian logic programs via transformation into propositional forms, i.e. into Bayesian networks [15] (see [7, Ch. 10] for details).

Markov logic [5] establishes a framework which combines Markov networks [15] with first-order logic to handle a broad area of statistical relational learning tasks. The Markov logic syntax complies with first-order logic where each formula is quantified by an additional weight value, e.g.

$$\begin{aligned} (\text{elephant}(X) \wedge \text{keeper}(Y) \Rightarrow \text{likes}(X, Y), & 2.2) \\ (\text{elephant}(X) \Rightarrow \text{likes}(X, \text{fred}), & -0.8) \qquad (\text{likes}(\text{clyde}, \text{fred}), \infty) \end{aligned}$$

Semantics are given to sets of Markov logic formulas by a probability distribution over propositional possible worlds that is calculated as a log-linear model over weighted ground formulas. The fundamental idea in Markov logic is that first-order formulas are not handled as hard constraints (which are indicated by weight ∞), but each formula is more or less softened depending on its weight. A *Markov logic network (MLN)* L is a set of weighted first-order logic formulas (F_i, w_i) together with a set of constants C . The semantics of L is given by a ground Markov network $M_{L,C}$ constructed from F_i and C [7, Ch. 12]. The standard semantics of Markov networks [15] is used for reasoning, e.g. to determine the consequences of L (see [7, Ch. 12] for details).

The syntax of *relational probabilistic conditional logic* (RPCL) [19] has already been used in the representation of Ex. II and employs conditionals of the form $(B \mid A)[x]$ with first-order formulas A, B and $x \in [0, 1]$. A conditional $(B \mid A)[x]$ represents a constraint on a probability distribution $P : \Omega \rightarrow [0, 1]$ on the set of possible worlds Ω and states that the conditional probability of B given A is x . In order to interpret conditionals containing free variables several relational semantics have been proposed, see [19, 13]. The *grounding semantics* [13] uses a *grounding operator* \mathcal{G} , e.g. universal instantiation, that translates a set \mathcal{R} of conditionals with free variables into a set of ground conditionals. Then, a probability distribution P \mathcal{G} -*satisfies* \mathcal{R} , denoted by $P \models_{\mathcal{G}} \mathcal{R}$, iff $P(B' \mid A') = x$ for every ground $(B' \mid A')[x] \in \mathcal{G}(\mathcal{R})$. Both *averaging* and *aggregating semantics* [12, 19] do not require a grounding operator but interpret the intended probability x of a conditional with free variables only as a guideline for the probabilities of its instances and the actual probabilities may differ from x . More precisely, a probability distribution P \emptyset -*satisfies* \mathcal{R} , denoted by $P \models_{\emptyset} \mathcal{R}$,

iff for every $(B | A)[x] \in \mathcal{R}$ it holds that $P(B_1 | A_1) + \dots + P(B_n | A_n) = nx$ where $(B_1 | A_1), \dots, (B_n | A_n)$ are the ground instances of $(B | A)$. A probability function P \odot -*satisfies* \mathcal{R} , denoted by $P \models_{\odot} \mathcal{R}$, iff for every $(B | A)[x] \in \mathcal{R}$ it holds that $P(B_1 \wedge A_1) + \dots + P(B_n \wedge A_n) = x(P(A_1) + \dots + P(A_n))$ where $(B_1 | A_1), \dots, (B_n | A_n)$ are the ground instances of $(B | A)$. Note that these three semantics are extensions of classical probabilistic semantics for propositional probabilistic conditional logic [11]. Based on any of these semantical notions the *principle of maximum entropy* [14][11] can be used for reasoning. The *entropy* H is an information-theoretic measure on probability distributions and is defined as a weighted sum on the information encoded in every possible world $\omega \in \Omega$: $H(P) = -\sum_{\omega \in \Omega} P(\omega) \log P(\omega)$. By employing the *principle of maximum entropy* one can determine the unique probability distribution that is the optimal model for a consistent knowledge base \mathcal{R} in an information-theoretic sense via

$$P_{\mathcal{R}}^{ME_{\circ}} = \arg \max_{P \models_{\circ} \mathcal{R}} H(P) \quad (1)$$

with \circ being one of \mathcal{G} , \emptyset , or \odot . We abbreviate the approaches of reasoning based on the principle of maximum entropy with grounding, averaging, and aggregating semantics with $ME_{\mathcal{G}}$, ME_{\emptyset} , and ME_{\odot} , respectively. We say that a formula $(B | A)[x]$ is \circ -inferred from \mathcal{R} iff $P_{\mathcal{R}}^{ME_{\circ}} \models_{\circ} (B | A)[x]$ with \circ being one of \mathcal{G} , \emptyset , or \odot .

3 Language Aspects

We start with discussing properties concerning the language of an approach to probabilistic relational knowledge representation, i. e. aspects relating to syntax and semantics. Firstly, the semantics of the components of the knowledge representation language should be as declarative as possible. In particular, it should be possible to express basic concepts directly and to have an intuitive meaning for all language constructs, inference and learning results:

- (L-1) **Direct expression of probabilities** in the form of “ A holds with a probability of x ”.
- (L-2) **Direct expression of conditional probabilities** as in “Provided that A holds, then the probability of B is x ”.

Since an RPCL knowledge base supports representing formulas of the form $(B | A)[x]$ which constrain the conditional probability of B given A to x in any model, $ME_{\mathcal{G}}$, ME_{\emptyset} , ME_{\odot} obviously fulfill (L-1) and (L-2). The same holds for BLPs when taking into account the conditional probability distribution functions cpd_c which must be defined for any Bayesian clause c . Since in an MLN there is no obvious correspondence between the weight of a formula and its corresponding probability and because conditionals are not supported, (L-1) and (L-2) do not apply to MLNs.

- (L-3) **Qualitative statements** like “ A is more probable than B ” or “ A is very probable”.

Such qualitative statements can be expressed within none of the five approaches.

(L-4) Commonsense meaning: Probabilities are used for expressing uncertainty, and for each basic construct of the knowledge representation language there should be a clear intuitive or commonsense meaning. Examples of such meanings are a statistical interpretation of expressions (with respect to a population), or a subjective degree of belief (with respect to the set of possible worlds).

The difference between statistical and subjective interpretations can be illustrated in the elephant-keeper-example \square by contrasting “Most elephants like their keepers” (statistical, as it refers to a whole population) vs. “Mostly, an elephant likes its keeper” (subjective, as it refers to situations, i.e., possible worlds).

A Bayesian clause c in a BLP expresses qualitative information about the conditional probability of the clause’s head given the body of the clause; the actual conditional probability is given by cpd_c which is applied for each instance. Thus, these informations (together with the combining rules) yield subjective conditional probabilities as a commonsense meaning of a BLP.

Although it can be observed that the greater the weight w of an MLN clause F the more impact F will have on the probability distribution in the resulting ground Markov network $M_{L,C}$, a more precise intuitive meaning of (F, w) is not evident. Besides this general negative statement, the probabilities resulting from an MLN can be classified as subjective, as the MLN semantics is based on possible worlds.

For each ground conditional $(B \mid A)[x]$ in an RPCL knowledge base, its commonsense meaning is given by the conditional probability of B given A being x for grounding, averaging, and aggregating semantics. However, the commonsense interpretation of conditionals with free variables is substantially different in these three semantics. For grounding semantics, a relational conditional is understood as a universal quantification of the subjective conditional probability of each ground instance within the range of the grounding operator. For averaging and aggregating semantics, the commonsense meaning is a mixture of statistical interpretation and degrees of belief. The averaging semantics yields a statistics of subjective conditional beliefs. For instance, the conditional c_1 in Ex. \square with an interpretation via ME_{\emptyset} reads as “Considering a random elephant-keeper-pair, the average subjective probability that the elephant likes its keeper is 0.9.” The aggregating semantics exchanges the role of statistical (or population based) and subjective view by providing kind of a subjectively weighted statistics. Here, c_1 is understood as “Considering all elephant-keeper-pairs, the expected subjective probability that elephants like their keepers is 0.9.” In contrast to the averaging semantics, the aggregating semantics gives more weight to individuals (or tuples of individuals) that are more likely to fulfill the premise of a conditional.

By taking both statistical and subjective aspects into account, both averaging and aggregating semantics allow a more realistic approach to commonsense reasoning in a relational probabilistic context. When entering a zoo (or considering the vague population of all elephants and keepers in the world) and uttering conditional c_1 of Ex. \square , human beings are very likely to express something like

"In (about) 90 % of all situations that involve an elephant and its keeper, I will notice that the elephant likes the keeper." This statement takes both beliefs about possible worlds and about the population into account, and it is exactly this perspective that averaging and aggregating semantics aim to represent. For a further discussion and evaluation of these semantics, see [12].

(L-5) Closure properties: The results obtained by inference should be expressible in the knowledge representation language, thus enabling, e.g., the integration of inferred knowledge into a knowledge base. Another closure aspect refers to the query language: Can any formula being allowed in a knowledge base be used in a query?

Given a (ground) query Q for a BLP, BLP inference can be used for computing a cpd_Q for Q by generating all possible combinations of evidence for Q , allowing one to add this information as a BLP clause. Since with MLNs also probabilities are computed, MLN inference results can not be used directly in an MLN knowledge base where a weight is required for a formula. On the other hand, ME inference results can be directly integrated into an RPCL knowledge base (independently of the actual semantics).

In all approaches, queries must be ground, and taking a logic formula F from a corresponding knowledge base, every ground instance of F can be used in a query. For example, given the body of the BLP clause

$$(likes(clyde, jim) \mid elephant(clyde), keeper(jim))$$

as evidence, the BLP inference mechanism will determine the conditional probability of $likes(clyde, jim)$ given the evidence. Consequently, open queries are not allowed in any of the approaches; if a support system offers posing queries with free variables (as it is allowed e.g. in Alchemy [5]), then such a query is being treated as an abbreviation for posing a sequence of all possible ground instantiations of that query.

(L-6) Semantical invariance: A general requirement for logic-based representations applies also here: The semantics should be the same if the same knowledge is expressed by syntactic variants.

Let KB be a knowledge base in any of the three relational approaches. Since for any variable renaming σ , the respective semantics of KB and $\sigma(KB)$ coincide, semantical equivalence with respect to variable renaming holds for BLPs, MLNs, and ME_o.

Another form of syntactic variants arises from propositionally equivalent formulas, e.g. A and $A \wedge A$. In places where such formulas are allowed, they do not give rise to a different semantics in any of the five approaches. However, it should be noted that this case has to be distinguished carefully from the case of adding a syntactic variant of a knowledge base element to that knowledge base: If $F \in KB$ and σ is a variable renaming replacing some variable in F with a

new variable not occurring in KB , then in general $KB \cup \{\sigma(F)\}$ has a different semantics both for BLPs and for MLNs. For instance, when using *noisy-or* as the combining function, the probability expressed by F —and thus also by $\sigma(F)$ —will typically increase when adding $\sigma(F)$ to KB .

Example 2. Consider a BLP consisting of the single clause $c = (A(\mathbf{X}) \mid B(\mathbf{X}))$ with $\text{cpd}_c(\text{true}, \text{true}) = 0.9$, $\text{cpd}_c(\text{true}, \text{false}) = 0.5$ and with *noisy-or* being the combining rule for predicate A . Then querying this BLP with $(A(d) \mid B(d))$ results (obviously) in the probability 0.9 for $A(d)$ being true given $B(d)$ is true. However, adding the clause c' with $c' = (A(\mathbf{Y}) \mid B(\mathbf{Y}))$ (with $\text{cpd}_c = \text{cpd}_{c'}$) which is a syntactical variant of c results in a probability of $1 - (1 - 0.9) \cdot (1 - 0.9) = 0.99$ as both c and c' determine a probability of 0.9 and these probabilities have to be combined by the corresponding combining function (*noisy-or* in this case) to obtain the final answer to the given query.

Example 3. Similarly, consider an MLN consisting of the single formula $(B(\mathbf{X}) \Rightarrow A(\mathbf{X}), 1)$. Querying this MLN with $(A(d) \mid B(d))$ results in the (approximated) probability 0.764974 for $A(d)$ being true given $B(d)$ is true. However, adding the syntactic variant $(B(\mathbf{Y}) \Rightarrow A(\mathbf{Y}), 1)$ results in an (approximated) probability of 0.861964 (these probabilities have been computed with the Alchemy system).

As inference in RPCL is defined on well-defined semantics, syntactical variants do not influence the outcome of inference (for grounding, averaging, and aggregating semantics).

(L-7) Explanation capabilities for inference: It is desirable to have explanation capabilities of inference results. Which elements of the knowledge base are responsible (to what degree) for an inferred result? Or which elements of the knowledge base did not affect a result in any way? Can every result (or at least some results) be derived (more or less) directly from certain elements of the knowledge base? Or does any result essentially require the calculation of an appropriate model?

The explanation of a BLP inference result is given by the obtained local Bayes net which also encodes a (logical) derivation of the query. Therefore, it is obvious which clauses of the BLP knowledge base were involved in the calculation of the result. So the BLP approach offers some distinct level of explanation capability.

MLN inference is based on a log-linear model that has to be normalized in order to represent a probability distribution, cf. [7, Ch. 12]. The value of this normalization constant depends on the relationships among the formulas of an MLN knowledge base. Therefore, an inferred probability depends on all formulas of the knowledge base, because the weights of the formulas are relative values, where the higher the weight the greater the influence of the formula. Since MLN inference involves the construction of an appropriate ground Markov network, independencies among certain ground atoms are indicated by this network. So some independency aspects of inferred results can be explained by the net structure.

Inference in RPCL relies on solving the optimization problem (II). In some special cases (regarding the query and the conditionals in the knowledge base), the result of a query might be estimated directly considering how reasoning under the maximal entropy distribution "behaves". So in such rare cases, the inferred result can be explained by certain aspects of the knowledge base (having the principle of maximum entropy in mind). But in general, no intuitive explanation of inference results is evident for both the MLN and RPCL approaches.

4 Strict and Propositional Knowledge

In a probabilistic relational modeling language two essential dimensions are present that distinguish the respective approach from propositional logic: The *probabilistic* and the *relational* dimension. From a knowledge representation point of view, the following questions arise naturally. What happens if one cuts down any of these two dimensions? Which kind of logic does one obtain?

(SP-1) Strict Knowledge: Suppose one restricts the sentences occurring in a knowledge base such that only strict probabilistic knowledge can be expressed. What is the representation level of this degenerated case, and what are its semantics and inference properties? In particular, what is its relationship to classical non-probabilistic (first-order) logic?

Of the formalisms BLP, MLN, and RPCL, only MLNs allow for existential quantifiers (which in the Alchemy system are replaced by corresponding finite disjunctions over instantiations with the elements of the underlying universe). Looking at the language of logical MLN formulas we thus have first-order logic, restricted to a finite fixed universe. In order to express that a particular formula F represents strict knowledge, the weight of F must be set to infinity [5]. In this case, all possible worlds violating the strict formula are assigned zero probabilities by the MLN, and the probabilities of the satisfying worlds sum up to 1. Hence, the formulas that can be inferred with probability 1 from such an MLN \mathcal{F} containing only strict formulas are the same as the formulas that can be derived from \mathcal{F} in a classical way, provided \mathcal{F} is satisfiable.

A Bayesian knowledge base containing only strict knowledge can be expressed by a BLP containing only conditional probabilities with values 0 and 1. In this case, also BLP semantics and BLP inference coincide with the semantics and inference in first-order logic. In RPCL, a strict knowledge base is also obtained by allowing just the two extreme probabilities 0 and 1. For a more detailed look at the relationship of the obtained logics to first-order logic, let FOL_{\forall} be the set of quantifier-free first order formulas without function symbols, with all variables being implicitly universally quantified. For strict formulas of BLPs, we get only a subset of FOL_{\forall} since in a BLP we can not express a disjunction like $A \vee B$.

Every set \mathcal{F} of formulas of FOL_{\forall} can be expressed by the RPCL knowledge base $\mathcal{F}_P = \{(A \mid \top)[1] \mid A \in \mathcal{F}\}$ containing only strict formulas. Then inference based on \mathcal{F} and \mathcal{F}_P is the same (independently of the actual used semantics for RPCL). Looking at the other direction, let KB be a strict RPCL knowledge

base, and let $KB_{FOL_V} = \{\neg A \vee B \mid (B \mid A)[1] \in KB\} \cup \{A \wedge \neg B \mid (B \mid A)[0] \in KB\}$. If KB is consistent with respect to grounding, averaging, or aggregating semantics then inference in KB and KB_{FOL_V} is the same. However, for the strict RPCL knowledge base $KB' = \{(B \mid A)[1], (A \mid \top)[0]\}$ we observe that KB' has no models since a probability distribution P can satisfy a conditional $\{(B \mid A)[x]\}$ only if $P(A) > 0$, independently of the actual semantics. On the other hand, $KB'_{FOL_V} = \{\neg A \vee B, \neg A\}$ does have a model. Thus, reducing a conditional to material implication is not adequate even in the case of only strict probabilistic conditionals (see also [1]).

Likewise, we can look at the degenerated knowledge representation formalism obtained by cutting out any relational representation aspects.

(SP-2) Propositional Knowledge: What kind of logic does one obtain if a knowledge base contains only ground knowledge? What are its semantics and inference properties, and in particular, what is its relationship to propositional probabilistic logic?

A BLP where all occurring atoms are ground obviously corresponds to a propositional Bayesian network. Restricting the formulas in an MLN to be variable-free yields the semantics of a propositional Markov net: If L is an MLN containing only ground atoms, then for any set C of constants the corresponding ground Markov net is independent of C . For a ground RPCL knowledge base grounding, averaging, and aggregating semantics coincide with classical probabilistic semantics in probabilistic conditional logic and inference based on the principle of maximum entropy is the same as in the propositional case, cf. [19].

5 Individuals and Universes

The core idea of relational knowledge representation is to talk about a set of elements (a *universe*) and the relations among them. Thus, methods are needed for specifying elements belonging to the universe, to refer to elements in the universe, and to reason about elements and their properties and relationships. In general, relational approaches may differ according to whether and how they support any of the following criteria.

(U-1) Listing of elements: Can universes be specified by explicitly listing all its elements?

The given facts in a BLP must all be ground; they determine the specific context of the BLP, thus allowing to list all elements of a universe by mentioning them in the atoms of the BLP. When defining an MLN, an explicit listing of all constants C must be given, and the semantics of an MLN requires that different constants denote different elements and that there are no elements other than the ones denoted by constants. Similarly, all constants in an RPCL knowledge base denote different elements, and there are no other elements.

(U-2) Open universe: Is it possible to have an *open* universe whose number of elements is not a-priori known?

In BLP, MLN, and RPCL it is not possible to specify such open universes directly. However, in all approaches the extensional part—i. e. the ground atoms resp. the given constants—can be exchanged while reusing the given generic knowledge. For instance, the constants occurring in a query Q together with the constants in a BLP P determine the Herbrand universe used to construct the ground Bayesian network for answering Q .

(U-3) Prototypical elements: Specification of prototypical elements of a universe.

A universally quantified variable X in a relational statement says that this statement applies to all elements of the considered universe. However, as Ex. II demonstrates, there is the need to also express knowledge about individuals, referred to by specific constants; in any of the five approaches, generic statements using variables may be combined with statements about individuals. In the elephant-keeper example, asking about a keeper *jim* will return the same probability as asking the same question about a keeper *james* since the respective knowledge bases do not contain any specific information neither about *jim* nor about *james*. More generally, let $C_{\mathcal{R}}$ be the set of constants occurring in a set of rules \mathcal{R} and let C_U be the set of all constants under consideration. (Note that for MLN and ME, C_U is given explicitly, and that for a BLP, C_U is determined when a query is posed.) Then the elements in $C_{prot} = C_U \setminus C_{\mathcal{R}}$ are all prototypical as they can not be distinguished by any query asked w.r.t. \mathcal{R} : If $d_1, d_2 \in C_{prot}$ and Q is a query containing d_1 , then the query Q' obtained from Q by replacing d_1 by d_2 (and possibly also d_2 by d_1) yields the same probability as Q . This observation holds for all of the five approaches.

(U-4) Inference for individuals: There should be a well-defined inference mechanism to infer probabilities for particular individuals (either prototypical individuals or specific, named individuals). Does such inference depend on the number of elements in a universe, and if so, what is the dependency?

Obviously, all approaches provide for querying about specific individuals. For example, given a BLP, a ground Bayes net can be constructed to infer probabilities for some ground query involving arbitrary constants. Similarly, this holds for MLNs and the approaches based on maximum entropy. Further, the number of elements in the universe might influence the probability of a query in all approaches. Consider the BLP B containing the clauses $(B(X) | A(X, Y))$ and $(A(X, Y))$. Given the query $B(c)$ for some constant c the probability of $B(c)$ depends on the number of instances of $A(c, Y)$, i. e., on the number of constants in the universe. If *noisy-or* is the combining rule for B then the probability of $B(c)$ tends towards one when the number of constants in the universe tends towards infinity, independently of the actual conditional probability distributions of $(B(X) | A(X, Y))$ and $(A(X, Y))$. A similar observation can be made for MLNs and RPCL. Another dependency of the number of elements in the universe and probabilities of queries arises for RPCL under averaging and aggregating semantics. Consider now the conditional $(B | A)[x]$ and averaging semantics. If $(B' | A')$ is an instance of $(B | A)$ that does not mention any constants in the knowledge

base then it is easy to see that the probability of $(B' | A')$ tends towards x if the number of elements in the universe tends towards infinity, cf. [12].

(U-5) Grounding: Is there a mechanism for (consistent) grounding of a knowledge base?

The semantics of a BLP or an MLN knowledge base is defined via complete groundings yielding a (ground) Bayesian network or a (ground) Markov net, respectively. In a BLP, the logic part consists of Horn clauses which do not allow the specification of negated conclusions, so that inconsistencies on the logical level are avoided. Conflicting specifications on the quantitative level may arise when having syntactical variants of a clause, e.g. $(B(X) | A(X))$ and $(B(Y) | A(Y))$ with different cpd's. Such conflicts are resolved via the combining rules like *noisy-or* (cf. Ex. 2). An MLN might contain both (F, w) and $(\neg F, w)$, but the grounded semantics is still consistent and well defined. For RPCL under grounding semantics, complete grounding might generate an inconsistency; therefore, various more sophisticated instantiation strategies have been proposed [13].

Another important aspect connected to the notion of relational knowledge and universes is the question whether probabilities are interpreted statistically or as subjective degrees of belief, cf. the discussion in the context of (L-4).

6 Conclusion and Future Work

During the last years, many different approaches extending probabilistic propositional logic to a relational setting have been proposed. In this paper, we developed and illustrated various evaluation and comparison criteria and applied them to five different modeling and inference methods, thereby putting emphasis on the knowledge representation point of view.

There are several additional criteria that require further research and more investigation in detail. When considering the expressivity of a particular modeling method, it is easy to see that any of the approaches discussed in this paper can be used to define an arbitrary probability distribution over a finite domain, but the more interesting question is *how* this can be done. Jaeger [9] proposes a schema of comparing different formalisms by using two components: A generic component that is independent of a particular universe, and a component that takes into account a universe of constants. The sharp separation of generic and specific knowledge as required in the expressivity analysis proposed in [9] is problematic since it prohibits a modeling taking into account both types of knowledge in the form as it is done for instance in Ex. 11.

Another criterion is to ask what kind of queries can be answered, and which can be answered efficiently. In the context of (L-5), we already discussed the syntactic form of queries that can be answered in the considered approaches. With respect to the complexity of inference, further experimental and theoretical work is needed. For instance, inference in RPCL requires solving the numerical optimization problem (11) whose complexity grows in the number of possible groundings. Work on lifted first-order probabilistic inference is done in e.g. [16][17], and in [6] for reasoning under maximum entropy.

References

1. Beierle, C., Kern-Isberner, G.: The relationship of the logic of big-stepped probabilities to standard probabilistic logics. In: Link, S., Prade, H. (eds.) FoIKS 2010. LNCS, vol. 5956, pp. 191–210. Springer, Heidelberg (2010)
2. Bruynooghe, M., et al.: An Exercise with Statistical Relational Learning Systems. In: Domingos, P., Kersting, K. (eds.) International Workshop on Statistical Relational Learning (SRL 2009), Leuven, Belgium (2009)
3. De Raedt, L., Kersting, K.: Probabilistic inductive logic programming. In: De Raedt, L., Frasconi, P., Kersting, K., Muggleton, S.H. (eds.) Probabilistic Inductive Logic Programming. LNCS (LNAI), vol. 4911, pp. 1–27. Springer, Heidelberg (2008)
4. Delgrande, J.: On first-order conditional logics. Artificial Intelligence 105, 105–137 (1998)
5. Domingos, P., Lowd, D.: Markov Logic: An Interface Layer for Artificial Intelligence. In: Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool, San Rafael (2009)
6. Fisseler, J.: Learning and Modeling with Probabilistic Conditional Logic. Dissertations in Artificial Intelligence, vol. 328. IOS Press, Amsterdam (2010)
7. Getoor, L., Taskar, B. (eds.): Introduction to Statistical Relational Learning. MIT Press, Cambridge (2007)
8. Jaeger, M.: Relational Bayesian Networks: A Survey. Electronic Transactions in Artificial Intelligence 6 (2002)
9. Jaeger, M.: Model-Theoretic Expressivity Analysis. In: De Raedt, L., Frasconi, P., Kersting, K., Muggleton, S.H. (eds.) Probabilistic Inductive Logic Programming. LNCS (LNAI), vol. 4911, pp. 325–339. Springer, Heidelberg (2008)
10. Jain, D., Mösenlechner, L., Beetz, M.: Equipping Robot Control Programs with First-Order Probabilistic Reasoning Capabilities. In: International Conference on Robotics and Automation (ICRA), pp. 3130–3135 (2009)
11. Kern-Isberner, G.: Characterizing the principle of minimum cross-entropy within a conditional-logical framework. Artificial Intelligence 98, 169–208 (1998)
12. Kern-Isberner, G., Thimm, M.: Novel Semantical Approaches to Relational Probabilistic Conditionals. In: Proc. Twelfth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2010), pp. 382–392 (2010)
13. Loh, S., Thimm, M., Kern-Isberner, G.: On the problem of grounding a relational probabilistic conditional knowledge base. In: Proceedings of the 14th International Workshop on Non-Monotonic Reasoning (NMR 2010), Toronto, Canada (May 2010)
14. Paris, J.B.: The uncertain reasoner’s companion – A mathematical perspective. Cambridge University Press, Cambridge (1994)
15. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Francisco (1998)
16. Poole, D.: First-order probabilistic inference. In: Gottlob, G., Walsh, T. (eds.) Proc. IJCAI 2003, pp. 985–991. Morgan Kaufmann, San Francisco (2003)
17. Singla, P., Domingos, P.: Lifted first-order belief propagation. In: Fox, D., Gomes, C.P. (eds.) Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, pp. 1094–1099. AAAI Press/The MIT Press (2008)
18. Thimm, M., Finthammer, M., Kern-Isberner, G., Beierle, C.: Comparing approaches to relational probabilistic reasoning: Theory and implementation (2010) (submitted)
19. Thimm, M., Kern-Isberner, G., Fisseler, J.: Relational probabilistic conditional reasoning at maximum entropy. In: Liu, W. (ed.) ECSQARU 2011. LNCS, vol. 6717, pp. 447–458. Springer, Heidelberg (2011)

Segmentation of Action Streams

Human Observers vs. Bayesian Binning

Dominik Endres, Andrea Christensen, Lars Omlor, and Martin A. Giese

Section for Computational Sensomotorics, Dept. of Cognitive Neurology,
University Clinic, CIN, HIH and University of Tübingen,
Fröndenbergstr 23, 72070 Tübingen, Germany
dominik.endres@klinikum.uni-tuebingen.de
{andrea.christensen,martin.giese}@uni-tuebingen.de

Abstract. Natural body movements are temporal sequences of individual actions. In order to realise a visual analysis of these actions, the human visual system must accomplish a temporal segmentation of action sequences. We attempt to reproduce human temporal segmentations with Bayesian binning (BB) [8]. Such a reproduction would not only help our understanding of human visual processing, but would also have numerous potential applications in computer vision and animation. BB has the advantage that the observation model can be easily exchanged. Moreover, being an exact Bayesian method, BB allows for the automatic determination of the number and positions of segmentation points. We report our experiments with polynomial (in time) observation models on joint angle data obtained by motion capture. To obtain human segmentation points, we generated videos by animating sequences from the motion capture data. Human segmentation was then assessed by an interactive adjustment paradigm, where participants had to indicate segmentation points by selection of the relevant frames. We find that observation models with polynomial order ≥ 3 can match human segmentations closely.

1 Introduction

Temporally segmenting (human) action streams is interesting for a variety of reasons: firstly, if we had a model which reproduced human segmentations closely, it might reveal important insights in human action representation. Previous work in this direction has studied in detail the segmentation of sequences of piecewise linear movements in the two-dimensional plane [23][1]. Secondly, a good temporal segmentation would have numerous applications in the field of computer vision. Worth mentioning in this context is the Human Motion Analysis (HMA) [24]. HMA concerns the detection, tracking and recognition of people from image sequences involving humans and finds its application in many areas such as smart surveillance and man-machine interfaces. Thirdly, extraction of important key frames by improved motion segmentation would not only contribute to computer vision research but also to computer graphics and motion synthesis. Animations of human motion data can be done with less computational costs if the key frames are defined optimally (e.g. [6][5]).

While most researchers base their temporal segmentation approaches on real video data and focus on the computer vision problem to analyse human motion data by tracking of skeleton models or feature sequences [21][2][13], we address here specifically the problem of the segmentation of action streams based on motion capture data. We compare Bayesian binning (BB) for segmentation of human full-body movement with human responses, which were assessed in an interactive video segmentation paradigm.

BB is a method for modelling data with a totally ordered structure, e.g. time series, by piecewise defined functions. Its advantages include automatic complexity control, which translates into automatic determination of the number and length of the segments in our context. BB was originally developed for density estimation of neural data and their subsequent information theoretic evaluation [8]. It was later generalised for regression of piecewise constant functions [14] and further applications in neural data analysis [10][9]. Concurrently, a closely related formalism for dealing with multiple change point problems was developed in [11].

We give a concise description of the data recordings in section 2, since these data have not been published before. The psychophysical experiments and their results are described in section 3. We use BB for the segmentation of joint angle data obtained by motion capture in section 4. Furthermore, we show how to use BB with non-constant observations models in the bins. In section 5 we present the segmentations achieved by BB and compare them with the psychophysical results. Finally, we discuss the advantages and limitations of our approach and give an outlook for further investigations in section 6.

2 Kinematical Data

The action streams we studied are solo Taekwondo activities performed by ten internationally successful martial artists. Each combatant performed the same fixed sequence of 27 kicks and punches, forming a so called *hyeong*. A complete hyeong had a full length of about 40 seconds. The kinematical data was obtained by motion capture using a VICON 612 system with 11 cameras, obtaining the 3D positions of 41 passively reflecting markers attached to the combatants' joints and limbs with a 3D reconstruction error of below 1 mm and at a sampling frequency 120 Hz.

The use of the obtained kinematical data was twofold. First, joint angle trajectories were computed from a hierarchical kinematic body model (skeleton) which was fitted to the original 3D marker positions. The rotations between adjacent segments of this skeleton were described by Euler angles, defining flexion, abduction and rotations about the connecting joint (e.g. [19][22]). Second, from the derived joint angle trajectories we created movie clips showing computer animations of the Taekwondo movements. Those videos served as stimuli in our psychophysical experiment to obtain human segmentation ratings.

3 Human Action Segmentation

To test the validity of the segmentation results obtained using our algorithmic approach we conducted a psychophysical study to achieve action segmentations of human observers.

Stimulus Preparation: short video clips displaying the Taekwondo movements served as stimuli in the psychophysical paradigm. Volumetric grey puppet constructed from simple geometric shape elements were animated with the combatants' movements. An illustration of the puppet's appearance is shown in fig. (II)A. To avoid stimuli of uncomfortable length each complete hyeong was split into five sub-sequences of comparable length each containing between three and eight separate Taekwondo actions. We restricted the number of stimuli in the experiment in order to prevent the participants from experimental fatigue and frustration. Thirty video clips corresponding to the complete hyeong of six representative combatants served as stimuli in this study. The puppet within the stimuli subtended approximately 4 x 8.6 degrees of visual angle and was presented on a computer screen viewed from a distance of 50 cm.

Experimental Procedure: the experiment started with a training phase in which participants familiarised themselves with the procedure. Five video clips corresponding to the complete hyeong of one combatant were only shown during this training phase. The remaining 25 movies served as test stimuli. Each was shown three times resulting in 75 segmentation trials per subject. Human observers watched video clips displaying the Taekwondo movements animated as puppets and segmented the complete hyeong into actions. In every trial the current video clip was first presented twice to enable the subjects to acquaint themselves with this action sequence. During the third presentation of the animation participants segmented the action sequence by pressing a marked key on the keyboard at each point which they perceived as the endpoint of one single, separable action. The segmentation was then replayed and if the participants felt insecure about their responses they had the opportunity to correct themselves up to two times. Noteworthy, it was completely left to the participants' own judgement what exactly defines *one single action* and the corresponding *endpoint*. They never received feedback regarding their segmentation neither during training nor during testing.

Participants: thirteen naïve subjects (mean age 26 years 6 month, ranging from 21 years 11 month to 38 years 11 month, 10 female) participated in this study. None of them had experience performing Taekwondo or other sports related to martial arts. All participants had normal or corrected-to-normal vision, gave informed written consent and were paid for their participation.

Segmentation Results of Human Observers: the results of the human action segmentation for the hyeong of one representative Taekwondo combatant are shown in fig. (II)B. Each single black dot represents one key press indicating the perception of an intersection between two Taekwondo actions. The 39 rows correspond to the three segmentation repetitions of each of the thirteen

participants. The lack of feedback and explicit definitions of a single action resulted in differences in the interpretation of *one separable action* between participants. Most participants (11) tended to divide the action sequences on a very fine-grained level resulting in many endpoints (mean number of segmentation points = 25.36, standard error = 2.49). Though, two subjects concentrated only on the coarse separation of the hyeong by setting only 5 respectively 8 segmentation points. In direct comparison with the timing of the 27 expected endpoints as defined by the Taekwondo combatants themselves (see coloured bars in fig. (D)B), naive participants placed 47.1% of the segmentations (standard error 5%) accurately within a very tight time window of +/-250 ms around the expected time point. Although a hit rate of 47.1% seems low at a first glance the following has to be taken into account. First, the complete hyeong was presented in 5 video clips. Each video ended at one expected endpoint. Some participants did not indicate a segmentation point at the video boundaries because they thought it would be redundant. Second, human observers tended to set the endpoint of the actions slightly too early compared to the expected endpoints. This happened especially when the combatant remained still for a longer time after he had completed an action. Shifting the accuracy time window from +/-250 ms to -380 ms to +120ms and excluding the video boundaries from the hit rate analysis results in a hit rate of 56.6%. Despite the slight shift in timing compared to the expected time points the set segmentations are consistent across subjects (see fig. (D)C and D for the segmentation density). These results are in accordance with previous findings about the agreement of human raters on boundary placing in movement sequences [7,18,25].

4 Bayesian Binning for Action Segmentation

We now briefly specify the BB model used for the segmentation of joint angle data. The following sections describe the prior over bin boundaries (section 4.1) and the used observation models (section 4.2). The algorithmic details of evaluating posterior expectations are only outlined schematically, they are detailed in [8].

4.1 The Bin Boundary Prior

Our aim is to model a time series D in the time interval $[t_{min}, t_{max}]$. We want to be able to draw conclusions about change point estimates from small amounts of data, let the model complexity be driven by D and handle D corrupted by (large amounts of) noise. We therefore take a Bayesian approach. Let $[t_{min}, t_{max}]$ be discretised into T contiguous intervals of duration $\Delta t = (t_{max} - t_{min})/T$, such that interval j is $[j \cdot \Delta t, (j+1) \cdot \Delta t]$ (see fig. (2)). Assume that Δt is small enough so that all relevant features of the data are captured in the discretised version of D . We model the generative process of D by $M + 1$ non-overlapping, contiguous bins, indexed by m and having *inclusive* upper boundaries $k_m \in \{k_m\}$. The bin m therefore contains the time interval $T_m = (\Delta t k_{m-1}, \Delta t k_m]$. Let D_m be that part of the data which falls into bin m . We presuppose that the probability of

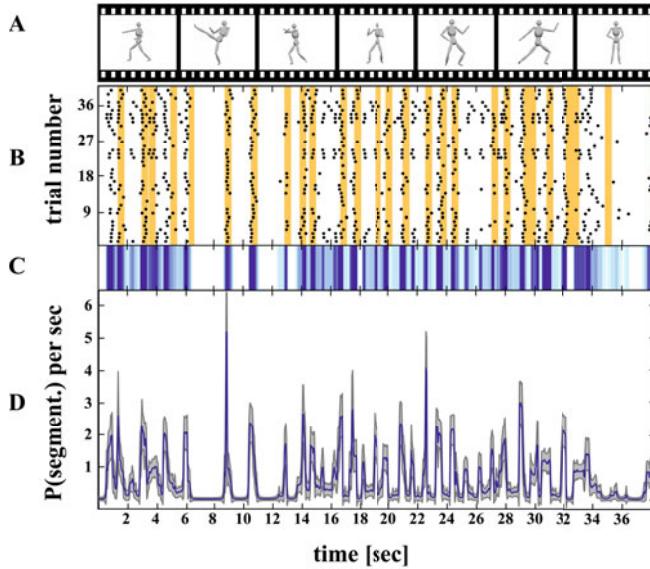


Fig. 1. Human Action Segmentation. A) Illustration of Stimuli. Snapshots taken from the stimuli videos showing the custom-built volumetric grey puppet performing different Taekwondo kicks and punches. B) Subjective Segmentation Points. Black dots correspond to the intersection points participants perceived between two Taekwondo actions. Results for the individual participants are shown row-wise. The coloured areas mark the time windows ± 250 ms around the expected endpoints as defined by experts. C) Predictive Segmentation Density I. Predictive segmentation density estimated from human key presses. Estimation was carried out by Bayesian binning with a Bernoulli-Beta observation model (see section 4.2). Colour saturation indicates density (darker = higher). D) Predictive Segmentation Density II. Same density as in C). Blue line represents the predictive segmentation density using Bayesian binning, the shaded grey area indicates \pm one posterior std. dev.

D given $\{k_m\}$ can be factorised as

$$P(D|\{k_m\}, M) = \prod_{m=0}^M P(D_m|k_{m-1}, k_m, M) \quad (1)$$

where we additionally define $k_{-1} = -1$, $k_M = T - 1$.

Prior on $\{k_m\}$: since we have no preferences for any bin boundary configuration (other than $m' < m \Rightarrow k_{m'} < k_m$), our prior is

$$P(\{k_m\}|M) = \binom{T-1}{M}^{-1} \quad (2)$$

$$\begin{aligned} P(D|\{k_m\}, M) &= \color{red}P(D_0|k_{-1}, k_0)\color{blue}P(D_1|k_0, k_1)\dots\color{green}P(D_M|k_{M-1}, k_M) \\ &= \prod_{m=0}^M P(D_m|k_{m-1}, k_m) \end{aligned}$$

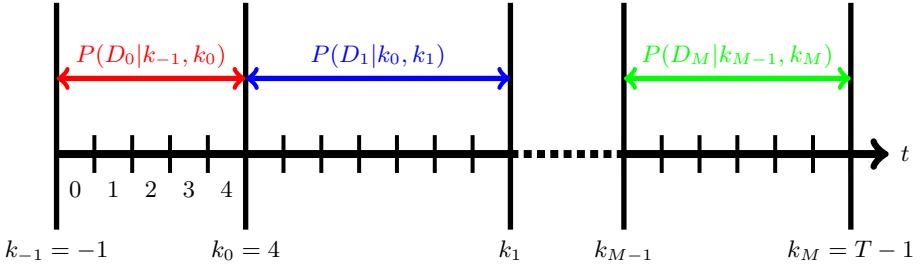


Fig. 2. Exemplary binning of a discrete time series of length T into $M + 1$ contiguous, non-overlapping bins with (inclusive) upper bin boundaries $k_m \in \{k_m\}$. Within each bin m , the observation model for data D is given by $P(D_m|k_{m-1}, k_m)$, where D_m is that part of the data which falls into bin m . We assume that the data are independent across bins given the $\{k_m\}$ and M .

where $\binom{T-1}{M}$ is just the number of possibilities in which M ordered bin boundaries can be distributed across $T - 1$ places (bin boundary M always occupies position $T - 1$, hence there are only $T - 1$ positions left).

Prior on M : we have no preference for any number of bin boundaries (which controls the model complexity). Thus, we let

$$P(M) = \frac{1}{T} \quad (3)$$

since the number of bin boundaries M must be $0 \leq M \leq T - 1$.

For temporal segmentation, the most relevant posterior is that of the $\{k_m\}$ for a given M :

$$P(\{k_m\}|D, M) = \frac{P(D|\{k_m\}, M)P(\{k_m\}|M)}{P(D|M)} \quad (4)$$

This requires the evaluation of $P(D|M)$:

$$P(D|M) = \sum_{k_0=0}^{k_1-1} \sum_{k_1=1}^{k_2-1} \dots \sum_{k_{M-1}=M-1}^{T-1} P(D|\{k_m\}, M) \quad (5)$$

which appears to be $\mathcal{O}(T^M)$ since it involves M sums of length $\mathcal{O}(T)$. However, exploiting the form of $P(D|\{k_m\}, M)$ (eqn. (1)) allows us to “push sums” past all factors which do not depend on the variable being summed over:

$$P(D|M) = \sum_{k_0=0}^{k_1-1} \sum_{k_1=1}^{k_2-1} \dots \sum_{k_{M-1}=M-1}^{T-1} \prod_{m=0}^M P(D_m|k_{m-1}, k_m)$$

$$\begin{aligned}
&= \sum_{k_0=0}^{k_1-1} P(D_0|k_{-1}, k_0) \sum_{k_1=1}^{k_2-1} P(D_1|k_0, k_1) \dots \\
&\quad \dots \sum_{k_{M-1}=M-1}^{T-1} P(D_M|k_{m-1}, k_M)
\end{aligned} \tag{6}$$

Now each sum over $\mathcal{O}(T)$ summands has to be evaluated $\mathcal{O}(T)$ times for the possible values of the upper summation boundary. Since there are M sums, this calculation has complexity $\mathcal{O}(MT^2)$, which is feasible. This way of computing $P(D|M)$ is an instance of the **sum-product** algorithm [16]. As detailed in [8], the expectation of any function of the model parameters (e.g. bin boundary position, bin width or probability of a bin boundary at a given point in time) can be evaluated with a similar approach, given that the function depends only on the parameters of one bin for any given $\{k_m\}$.

4.2 Observation Models $P(D|\{k_m\})$ for Action Streams

We employed two different observation models. For both, conjugate priors can be specified on their parameters which allow for an evaluation of expectations and marginal probabilities in closed form. This enables us to compute efficiently the marginal probability of the data given the number of bin boundaries (eqn. (6)).

Bernoulli-Beta: human segmentation events (i.e. key presses by observers) are binary. It is therefore natural to model these data with a Bernoulli process having a conjugate Beta prior (one per bin). This is analogous to modelling neural spike trains with BB [10]. Thus, for a segmentation event $e(t) \in D$ at time t in bin m , i.e. $t \in T_m$ we have

$$P(e(t)|t \in T_m) = P_m \tag{7}$$

$$p(P_m) = \text{B}(P_m; \gamma_m, \delta_m) \tag{8}$$

where $\text{B}(P_m; \gamma_m, \delta_m)$ is the Beta density with parameters γ_m, δ_m (see e.g. [4]).

Multivariate Gaussian with Polynomial Time-Dependence: joint angles are real numbers in $[-\pi, \pi]$. We could thus employ a multivariate von-Mises density or generalisations thereof [17]. Instead, we chose to model joint angles with a multivariate Gaussian whose mean has a polynomial time dependence, because its conjugate priors are tractable analytically. The exponential family conjugate prior on the mean μ and the precision matrix \mathbf{P} (inverse covariance) is then given by an extended Gauss-Wishart density (see e.g. [4]). Let $\mathbf{X}_t \in D$ be a L -dimensional vector of joint angles at time $t \in T_m$, and S be the chosen polynomial order. Let $t_m = \Delta t k_{m-1}$ be the start time of bin m . Then

$$p(\mathbf{X}_t | t \in T_m) = \mathcal{N}(\mathbf{X}_t; \boldsymbol{\mu}_m, \mathbf{P}_m^{-1}) \tag{9}$$

$$p(\mathbf{P}_m | \nu_m, \mathbf{V}_m) = \mathcal{W}(\mathbf{P}_m; \nu_m, \mathbf{V}_m) \tag{10}$$

$$\boldsymbol{\mu}_m = \sum_{i=0}^S \boldsymbol{a}_{i,m} (t - t_m)^i \quad (11)$$

The $\boldsymbol{a}_m = (\boldsymbol{a}_{i,m})$ are the polynomial coefficients in bin m . Note that this vector has $(S + 1) \cdot L$ components. $\mathcal{N}(\mathbf{X}, \boldsymbol{\mu}, \boldsymbol{\Sigma} = \mathbf{P}^{-1})$ is a multivariate Gaussian density in \mathbf{X} with means $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. $\mathcal{W}(\mathbf{P}; \nu, \mathbf{V})$ is a Wishart density in \mathbf{P} with ν degrees of freedom and scale matrix \mathbf{V} . To construct a prior which is conjugate to the likelihood (eqn. 9), we choose a vector $\boldsymbol{\alpha}_m = (\boldsymbol{\alpha}_{i,m})$ with $(S + 1) \cdot L$ components, which are the biases on \boldsymbol{a}_m . Furthermore, we introduce a symmetric, positive (semi-)definite $(S + 1) \times (S + 1)$ matrix \mathbf{B}_m , which contains the concentration parameters on \boldsymbol{a}_m . The prior on \boldsymbol{a}_m given \mathbf{P}_m is then a multivariate Gaussian density

$$p(\boldsymbol{a}_m | \boldsymbol{\alpha}_m, \mathbf{B}_m, \mathbf{P}_m) = \mathcal{N}(\boldsymbol{a}_m; \boldsymbol{\alpha}_m, \mathbf{Q}_m^{-1}) \quad (12)$$

where the $(S+1)L \times (S+1)L$ matrix \mathbf{Q}_m is obtained by block-wise multiplication of the entries $\mathbf{B}_{m,i,j}$ of \mathbf{B}_m with \mathbf{P}_m :

$$\mathbf{Q}_m = \begin{pmatrix} \mathbf{B}_{m,0,0}\mathbf{P}_m & \cdots & \mathbf{B}_{m,0,S}\mathbf{P}_m \\ \vdots & \ddots & \vdots \\ \mathbf{B}_{m,S,0}\mathbf{P}_m & \cdots & \mathbf{B}_{m,S,S}\mathbf{P}_m \end{pmatrix} \quad (13)$$

Lengthy but straightforward calculations confirm that the product of the Gaussian (eqn. 12) with the Wishart (eqn. 10) does indeed constitute a conjugate prior on the likelihood given by eqn. 9. We omit these calculations here for brevity. Since the prior is conjugate with a known normalisation constant (i.e. that of the Gaussian times the Wishart), the marginal likelihood of the data in each bin can be computed, and thus Bayesian binning can be applied with this observation model.

5 Results

We applied BB to joint angle trajectories of shoulder and elbow angles, and combinations thereof, to determine the segmentation densities. Fig. (3), left, panel A shows the predictive trajectories of an elbow angle computed with a 0th order and a 4th order observation models. Both models fit the data well, but the 4th order model yields a better fit while needing less bin boundaries, as indicated by the M posterior in fig. (3), right. Panels B and C in fig. (3), left, depict the predicted segmentation densities, showing where the 0th order model inserts the additional boundaries compared to the 4th order model.

Fig. (4) shows comparisons between human and BB segmentation densities. Note that the human (panel A, in fig. (1)) and the BB segmentation densities peak usually in close temporal vicinity. The 0th order model (panel B) over-segments, this over-segmentation is already reduced for the 2nd order model (panel C) and virtually gone for the 4th order model (panel D).

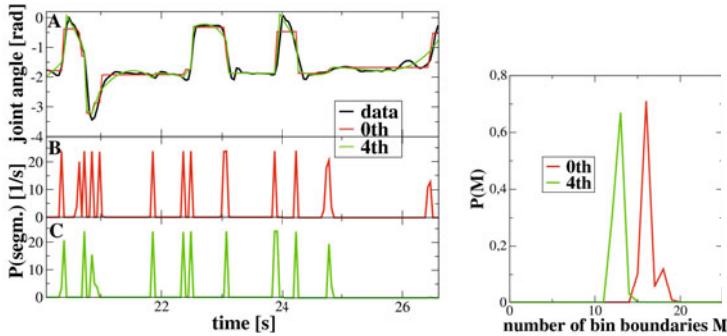


Fig. 3. *Left:* A: fitting a part of a joint angle trajectory with Bayesian binning. Joint angles have not been wrapped around at $-\pi$ to avoid creation of artificial segmentation points. Red lines shows predictive joint angles with a 0th order (i.e. bin-wise constant) observation model (see section 4.2), green lines show predictions from 4th order observation model. B,C: predictive segmentation densities for these two observation models. The 4th order model needs less segmentation points than the 0th order model, and also yields a more faithful fit of the joint angle trajectory. *Right:* posterior distribution of the number of bin boundaries M . The M -posterior of the 4th order observation model peaks at smaller values of M than the 0th order model, indicating that the 0th order model requires more bins to fit the data well. Note that both peaks are far from the maximum $M = 171$, i.e. over-fitting is avoided.

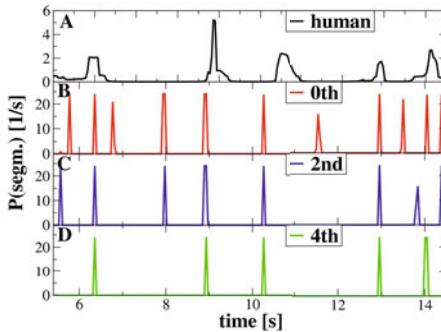


Fig. 4. Comparison of human segmentation densities with those obtained by Bayesian binning. Shown is an interval with a few, relatively clear segmentation points and good agreement between human subjects. Note that the human segmentation density (panel A) peaks usually closely to a peak in the density obtained by Bayesian binning. The 0th order model (panel B) predicts more segmentation points than the higher-order models (panels C,D), and the higher-order models are in better agreement to the human segmentation, both in number and location of the segmentation points.

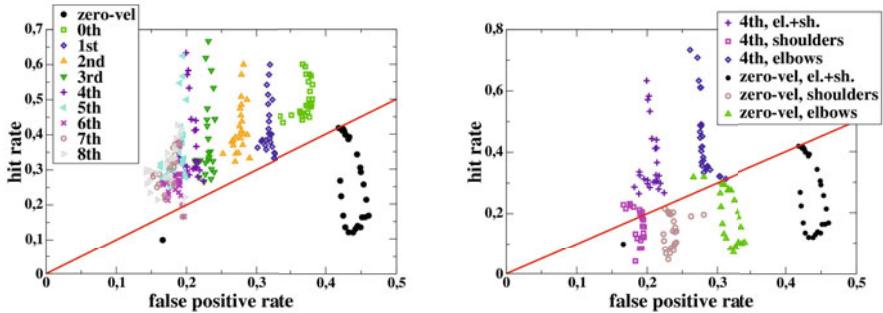


Fig. 5. Hit rate performance analysis. Red line: line of no discrimination. *zero-vel*: segmentation based on zero-crossings of angular velocity. *Left*: comparison between observation models of different polynomial orders (S in eqn. (II)). Elbow and shoulder angles were jointly segmented. An observation model with $S \in \{3, 4, 5\}$ offers the best compromise between a high hit rate and a low false positive rate. *Right*: performance dependence on joint angles for a model with $S = 4$ and the *zero-vel* segmentation. We segmented either elbow angles only, or shoulder angles only, or both together (*el.+sh.* in the legend). The latter yields the best segmentation results. For details, see text.

For a more quantitative evaluation of the agreement between human subjects and BB, we performed a hit rate/false positive rate analysis. Hits and false positives were computed by thresholding the segmentation densities (see fig. (4) and fig. (5)D), thereby yielding a binary segmentation event signal for each point in time. Every human segmentation event in a 400 ms accuracy window after a BB segmentation event was counted as a hit, the lack of a human segmentation event in this window counted as a false positive. This choice of accuracy window length was motivated by the comparison between naïve and expert human observers presented in section 3. We varied the threshold between 0.1 and 3.0 to obtain the data shown in fig. (5). As a simple baseline for comparison, we also computed segmentation points by searching for zero-crossings of angular velocity (*zero-vel* in fig. (5)). Using angular velocity zero-crossings as a baseline method was inspired by [15]. The zero-crossing search was carried out by computing local (300 ms window) parabolic fits to the joint angle data at every point in time, and checking whether the 1st order coefficient of the fit was close to 0.

Fig. (5), *left* shows a comparison between observation models of different polynomial orders (S in eqn. (II)). Elbow and shoulder angles were jointly segmented. Observation models with $S \in \{3, 4, 5\}$ offer the best compromise between a high hit rate and a low false positive rate. For all orders S , BB is a lot better than the baseline method. Fig. (5), *right* depicts the performance dependence on joint angles for a model with $S = 4$ and the *zero-vel* segmentation. We segmented either elbow angles only, or shoulder angles only, or both together. Segmenting both angles together yields the best segmentation results.

The fact that models with $S \in \{3, 4, 5\}$ provide a better match than the lower orders indicates that humans employ (the visual equivalent of) angular acceleration discontinuities, rather than discontinuities in angular velocities when segmenting action streams. This agrees with the 'minimum jerk' hypothesis [12].

6 Conclusion

In this paper, we have shown how to extend Bayesian binning by piecewise polynomial observation models and demonstrated its usefulness for action stream segmentation. Furthermore, we have created a ground truth data set for the evaluation of machine segmentation methods against human observers. Comparing our method to other automatic motion segmentation approaches, e.g. [3], will be interesting future work.

Previously, trajectories were successfully fitted with parabolic pieces [20]. We showed that higher orders yield a yet better agreement with human psychophysical data. One might also consider using a hidden Markov model (HMM) in each bin. The BB prior might be a feasible way of switching between HMMs, which were used for action segmentation in [13].

Our approach does not yet include context information into the segmentation process, we utilised only purely kinematic information. [25] reports that humans use context information for segmentation tasks when such is available, and rely increasingly on kinematics when context is reduced. Thus, including context can be expected to improve performance further.

Acknowledgements. This work was supported by EU projects FP7-ICT-215866 SEARISE, FP7-249858-TP3 TANGO, FP7-ICT-248311 AMARSi and the DFG. We thank Engelbert Rotalsky, Hans Leberle and the Taekwondo Unions of Nordrhein-Westfalen and Baden-Württemberg for cooperation on the data acquisition. We thank S. Cavdaroglu, W. Ilg and T. Hirscher for their help with data collection and post-processing.

References

1. Agam, Y., Sekuler, R.: Geometric structure and chunking in reproduction of motion sequences. *Journal of Vision* 8(1) (2008)
2. Albu, A.B., Bergevin, R., Quirion, S.: Generic temporal segmentation of cyclic human motion. *Pattern Recognition* 41(1), 6–21 (2008)
3. Barbić, J., Safanova, A., Pan, J.Y., Faloutsos, C., Hodgins, J.K., Pollard, N.S.: Segmenting motion capture data into distinct behaviors. In: *Proceedings of Graphics Interface GI 2004*. Canadian Human-Computer Communications Society, School of Computer Science, pp. 185–194. University of Waterloo, Waterloo (2004), <http://portal.acm.org/citation.cfm?id=1006058.1006081>
4. Bishop, C.M.: *Pattern Recognition and Machine Learning*. Springer, Heidelberg (2007)
5. Bruderlin, A., Williams, L.: Motion signal processing. In: *SIGGRAPH*, pp. 97–104 (1995)

6. Chen, W., Zhang, J.J.: Parametric model for video content analysis. *Pattern Recognition Letters* 29(3), 181–191 (2008)
7. Dickman, H.R.: The perception of behavioral units. In: Barker, R.G. (ed.) *The stream of behavior*, pp. 23–41. Appleton-Century-Crofts, New York (1963)
8. Endres, D., Földiák, P.: Bayesian bin distribution inference and mutual information. *IEEE Transactions on Information Theory* 51(11), 3766–3779 (2005)
9. Endres, D., Oram, M.: Feature extraction from spike trains with bayesian binning: latency is where the signal starts. *Journal of Computational Neuroscience* 29, 149–169 (2009), doi:10.1007/s10827-009-0157-3
10. Endres, D., Oram, M., Schindelin, J., Földiák, P.: Bayesian binning beats approximate alternatives: estimating peri-stimulus time histograms. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) *Advances in Neural Information Processing Systems*, vol. 20, pp. 401–408. MIT Press, Cambridge (2008)
11. Fearnhead, P.: Exact and efficient bayesian inference for multiple changepoint problems. *Statistics and Computing* 16(2), 203–213 (2006)
12. Flash, T., Hogan, N.: The coordination of arm movements: an experimentally confirmed mathematical model. *J. Neurosci.* (5), 1688–1703 (1985)
13. Green, R.D.: Spatial and temporal segmentation of continuous human motion from monocular video images. In: *Proceedings of Image and Vision Computing, New Zealand*, pp. 163–169 (2003)
14. Hutter, M.: Exact bayesian regression of piecewise constant functions. *Journal of Bayesian Analysis* 2(4), 635–664 (2007)
15. Ilg, W., Bakir, G., Mezger, J., Giese, M.: On the representation, learning and transfer of spatio-temporal movement characteristics. *International Journal of Humanoid Robotics* 1(4), 613–636 (2004)
16. Kschischang, F., Frey, B., Loeliger, H.A.: Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory* 47(2), 498–519 (2001)
17. Marida, K.V., Jupp, P.E.: *Directional Statistics*. Wiley, Chichester (2000)
18. Newtson, D., Engquist, G.: The perceptual organization of ongoing behavior. *Journal of Experimental Social Psychology* 12(5), 436–450 (1976)
19. Omlor, L.: New methods for anechoic demixing with application to shift invariant feature extraction. PhD in informatics, Universität Ulm. Fakultät für Ingenieurwissenschaften und Informatik (2010) urn:nbn:de:bsz:289-vts-72431
20. Polyakov, F., Stark, E., Drori, R., Abeles, M., Flash, T.: Parabolic movement primitives and cortical states: merging optimality with geometric invariance. *Biol. Cybern.* 100(2), 159–184 (2009)
21. Quirion, S., Branzan-Albu, A., Bergevin, R.: Skeleton-based temporal segmentation of human activities from video sequences. In: *Proceedings WSCG 2005 - 13-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2005* (2005)
22. Roether, C.L., Omlor, L., Christensen, A., Giese, M.A.: Critical features for the perception of emotion from gait. *Journal of Vision* 9(6) (2009)
23. Shipley, T.F., Maguire, M.J., Brumberg, J.: Segmentation of event paths. *Journal of Vision* 4(8) (2004)
24. Wang, L., Hu, W., Tan, T.: Recent developments in human motion analysis. *Pattern Recognition* 36(3), 585–601 (2003)
25. Zacks, J.M., Kumar, S., Abrams, R.A., Mehta, R.: Using movement and intentions to understand human activity. *Cognition* 112(2), 201–216 (2009)

Speedy Local Search for Semi-Supervised Regularized Least-Squares

Fabian Gieseke¹, Oliver Kramer¹, Antti Airola², and Tapio Pahikkala²

¹ Department Informatik

Carl von Ossietzky Universitat Oldenburg

26111 Oldenburg, Germany

fabiangieseke@googlemail.com,

okramer@icsi.berkeley.edu

² Turku Centre for Computer Science,

Department of Information Technology,

University of Turku, 20520 Turku, Finland

{antti.airola,tapio.pahikkala}@utu.fi

Abstract. In real-world machine learning scenarios, labeled data is often rare while unlabeled data can be obtained easily. Semi-supervised approaches aim at improving the prediction performance by taking both the labeled as well as the unlabeled part of the data into account. In particular, semi-supervised support vector machines favor decision hyperplanes which lie in a “low-density area” induced by the unlabeled patterns (while still considering the labeled part of the data). The associated optimization problem, however, is of combinatorial nature and, hence, difficult to solve. In this work, we present an efficient implementation of a simple local search strategy that is based on matrix updates of the intermediate candidate solutions. Our experiments on both artificial and real-world data sets indicate that the approach can successfully incorporate unlabeled data in an efficient manner.

1 Introduction

If sufficient labeled training data is available, well-known classification techniques like the *k-nearest neighbor*-classifier or *support vector machines* (SVMs) [10] often yield satisfying results. In real-world applications, however, labeled data is mostly rare. One of the current research directions in machine learning is *semi-supervised learning* [5][18]. Compared to purely supervised learning approaches, semi-supervised techniques try to take advantage not only of the labeled but also of the unlabeled data which can often be gathered more easily. One of the most prominent semi-supervised classification approaches are *semi-supervised support vector machines* (S³VMs) [1], which depict the direct extension of support vector machines to semi-supervised scenarios: Given a set of labeled training patterns, the goal of a standard support vector machine consists in finding a hyperplane which separates both classes well such that the “margin” induced by the hyperplane and the patterns is maximized, see Figure 1(a). This concept can

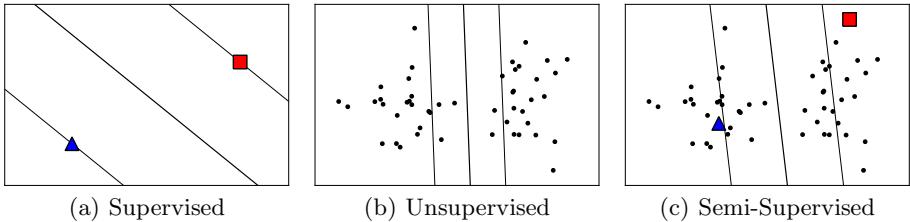


Fig. 1. In supervised scenarios, we are only given labeled patterns (squares and triangles). Thus, given only a small amount of data, a support vector machine cannot yield a good classification model, see Figure (a). Both unsupervised and semi-supervised support vector machines try to incorporate unlabeled patterns (dots) to reveal more information about the structure of the data, see Figures (b) and (c).

also be considered in learning scenarios where only unlabeled training patterns are given. Here, the goal consists in finding the optimal partition of the data into two classes (given some constraints) such that a *subsequent* application of a support vector machine leads to the best possible result, see Figure 1(b). Semi-supervised support vector machines can be seen as “intermediate” approach between the latter two ones. Here, the aim of the learning task consists in finding a hyperplane which separates both classes well (based on the labeled part of the data) and, at the same time, passes through a “low-density area” induced by the unlabeled part of the data, see Figure 1(c).

1.1 Related Work and Contribution

The original problem formulation of semi-supervised support vector machines was given by Vapnik and Sterin [16] under the name *transductive support vector machines*. Aiming at practical settings, Joachims [11] proposed a label-switching strategy which iteratively tries to improve an initial “guess” obtained via a (modified) support vector machine on the labeled part of the data. A variety of different techniques has been proposed in recent years which are based on semi-definite programming [2], the convex-concave procedure [8], deterministic annealing [14], the continuation method [4] and other techniques [17, 17]. Many other approaches exist and we refer to Chapelle *et al.* [5] and Zhu *et al.* [18] for comprehensive surveys. In this paper, we propose an efficient implementation of a least-squares variant for the original problem definition. More precisely, we propose an efficient implementation of a simple local search strategy which is based on matrix update schemes of the intermediate candidate solutions. Our experiments indicate that the resulting approach can incorporate unlabeled data successfully in an extremely efficient manner.¹

¹ The work at hand is related to our previous work for the unsupervised case [9]; we would like to point out that the new (matrix) derivations for the semi-supervised setting comprehend the old ones as a special case.

1.2 Notations

We use $[n]$ to denote the set $\{1, \dots, n\}$. Further, the set of all $n \times m$ matrices with real coefficients is denoted by $\mathbb{R}^{n \times m}$. Given a matrix $\mathbf{M} \in \mathbb{R}^{n \times m}$, we denote the element in the i -th row and j -th column by $[\mathbf{M}]_{i,j}$. For two sets $R = \{i_1, \dots, i_r\} \subseteq [n]$ and $S = \{k_1, \dots, k_s\} \subseteq [m]$ of indices, we use $\mathbf{M}_{R,S}$ to denote the matrix that contains only the rows and columns of \mathbf{M} that are indexed by R and S , respectively. Moreover, we set $\mathbf{M}_{R,[m]} = \mathbf{M}_R$. At last, we use y_i to denote the i -th coordinate of a vector $\mathbf{y} \in \mathbb{R}^n$.

2 Semi-Supervised Regularized Least-Squares

In supervised classification scenarios, a set $T_l = \{(\mathbf{x}'_1, y'_1), \dots, (\mathbf{x}'_l, y'_l)\}$ of training patterns \mathbf{x}'_i belonging to a set X with associated class labels $y'_i \in Y = \{-1, +1\}$ is given. For semi-supervised settings, we are additionally given a set $T_u = \{\mathbf{x}_1, \dots, \mathbf{x}_u\} \subset X$ of training patterns without any class information.

2.1 Regularized Least-Squares Classification

Our approach can be seen as an extension of the *regularized least-squares classification* [12] technique. Both support vector machines and this concept belong to regularization problems of the form

$$\inf_{f \in \mathcal{H}} \frac{1}{l} \sum_{i=1}^l \mathcal{L}(y'_i, f(\mathbf{x}'_i)) + \lambda \|f\|_{\mathcal{H}}^2, \quad (1)$$

where $\lambda > 0$ is a fixed real number, $\mathcal{L} : Y \times \mathbb{R} \rightarrow [0, \infty)$ is a *loss function* measuring the “performance” of the prediction function f on the training set, and $\|f\|_{\mathcal{H}}^2$ is the squared norm in a so-called *reproducing kernel Hilbert space* $\mathcal{H} \subseteq \mathbb{R}^X = \{f : X \rightarrow \mathbb{R}\}$ induced by a *kernel function* $k : X \times X \rightarrow \mathbb{R}$ [13, 15]. By using the square loss $\mathcal{L}(y, t) = (y - t)^2$, one obtains

$$\inf_{f \in \mathcal{H}} \frac{1}{l} \sum_{i=1}^l (y'_i - f(\mathbf{x}'_i))^2 + \lambda \|f\|_{\mathcal{H}}^2. \quad (2)$$

Due to the *representer theorem* [13], any minimizer $f^* \in \mathcal{H}$ of (2) has the form

$$f^*(\cdot) = \sum_{j=1}^l c_j k(\mathbf{x}'_j, \cdot) \quad (3)$$

with appropriate coefficients $\mathbf{c} = (c_1, \dots, c_l)^T \in \mathbb{R}^l$. Hence, by using $\|f^*\|_{\mathcal{H}}^2 = \mathbf{c}^T \mathbf{K} \mathbf{c}$ [13], where $\mathbf{K} \in \mathbb{R}^{l \times l}$ is the symmetric kernel matrix with entries of the form $[\mathbf{K}]_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$, we can rewrite the problem (2) as

$$\underset{\mathbf{c} \in \mathbb{R}^l}{\text{minimize}} \frac{1}{l} (\mathbf{y}' - \mathbf{K} \mathbf{c})^T (\mathbf{y}' - \mathbf{K} \mathbf{c}) + \lambda \mathbf{c}^T \mathbf{K} \mathbf{c}, \quad (4)$$

where $\mathbf{y}' = (y'_1, \dots, y'_l)^T$. This optimization problem is convex and, thus, easy to solve given standard techniques for convex optimization [3].

2.2 Semi-Supervised Extension

The goal of the semi-supervised learning process is to find an “optimal” prediction function for unseen data based on both the labeled and the unlabeled part of the training data. More precisely, we search for a function $f^* \in \mathcal{H}$ and a labeling vector $\mathbf{y}^* = (y_1^*, \dots, y_u^*)^\top \in \{-1, +1\}^u$ for the unlabeled training patterns which are optimal with respect to

$$\begin{aligned} & \underset{f \in \mathcal{H}, \mathbf{y} \in \{-1, +1\}^u}{\text{minimize}} \quad \frac{1}{l} \sum_{i=1}^l \mathcal{L}(y'_i, f(\mathbf{x}'_i)) + \lambda' \frac{1}{u} \sum_{i=1}^u \mathcal{L}(y_i, f(\mathbf{x}_i)) + \lambda \|f\|_{\mathcal{H}}^2 \quad (5) \\ & \text{s.t.} \quad \left| \frac{1}{u} \sum_{i=1}^u \max(0, y_i) - b_c \right| < \varepsilon, \end{aligned}$$

where $\lambda', \lambda > 0$ are cost parameters and where $\varepsilon > 0$. The last equation of the above task is called the *balance constraint*; it enforces the class ratio on the unlabeled part of the data to be approximately the same as an user-defined parameter b_c . We will denote assignments \mathbf{y} fulfilling the latter constraint as *valid*. Again, due to the representer theorem [13], any optimal function $f^* \in \mathcal{H}$ for a *fixed* partition vector $\mathbf{y} \in \{-1, +1\}^u$ has the form

$$f^*(\cdot) = \sum_{j=1}^l c_j k(\mathbf{x}'_j, \cdot) + \sum_{j=l+1}^{l+u} c_j k(\mathbf{x}_{j-l}, \cdot) \quad (6)$$

with appropriate coefficients $\mathbf{c} = (c_1, \dots, c_{l+u})^\top \in \mathbb{R}^{l+u}$. Hence, by plugging in the square loss, one can reformulate the above optimization problem as

$$\begin{aligned} & \underset{\mathbf{c} \in \mathbb{R}^n, \mathbf{y} \in \{-1, +1\}^n}{\text{minimize}} \quad J(\mathbf{c}, \mathbf{y}) = (\mathbf{\Lambda}\mathbf{y} - \mathbf{\Lambda}\mathbf{K}\mathbf{c})^\top (\mathbf{\Lambda}\mathbf{y} - \mathbf{\Lambda}\mathbf{K}\mathbf{c}) + \lambda \mathbf{c}^\top \mathbf{K} \mathbf{c} \quad (7) \\ & \text{s.t.} \quad \left| \frac{1}{u} \sum_{i=l+1}^n \max(0, y_i) - b_c \right| < \varepsilon, \\ & \text{and} \quad y_i = y'_i \text{ for } i = 1, \dots, l, \end{aligned}$$

where \mathbf{K} is the kernel matrix (based on the sequence $\mathbf{x}'_1, \dots, \mathbf{x}'_l, \mathbf{x}_1, \dots, \mathbf{x}_u$), $\mathbf{\Lambda}$ a diagonal matrix with entries of the form $[\mathbf{\Lambda}]_{i,i} = \sqrt{\frac{1}{l}}$ for $i = 1, \dots, l$ and $[\mathbf{\Lambda}]_{i,i} = \sqrt{\frac{\lambda'}{u}}$ for $i = l+1, \dots, n$, and where $n = l+u$.

3 Local Search Revisited

We follow the optimization scheme proposed for the unsupervised case [9] and apply a local search strategy along with efficient matrix updates for the involved intermediate candidate solutions.

Algorithm 1. Local Search

Require: A set of labeled training patterns $\{(\mathbf{x}'_1, y'_1), \dots, (\mathbf{x}'_l, y'_l)\}$, a set of unlabeled training patterns $\{\mathbf{x}_1, \dots, \mathbf{x}_u\}$, and model parameters $\lambda', \lambda, b_c, \varepsilon$.

Ensure: An approximation $(\mathbf{c}^*, \mathbf{y})$ for the task (7).

- 1: Initialize $\mathbf{y} \subseteq \{-1, +1\}^n$ (see text)
- 2: $t = 0$
- 3: **while** $t < \tau$ **do**
- 4: Generate $\bar{\mathbf{y}}$ by flipping a single coordinate $j \in \{l+1, \dots, l+u\}$ of \mathbf{y}
- 5: **if** $F(\bar{\mathbf{y}}) < F(\mathbf{y})$ and $\bar{\mathbf{y}}$ is valid **then**
- 6: Replace \mathbf{y} by $\bar{\mathbf{y}}$
- 7: **end if**
- 8: $t = t + 1$
- 9: **end while**
- 10: Compute \mathbf{c}^* for $\text{minimize}_{\mathbf{c} \in \mathbb{R}^n} J(\mathbf{c}, \mathbf{y})$
- 11: **return** $(\mathbf{c}^*, \mathbf{y})$

3.1 Local Search

The local search strategy is depicted in Algorithm 1. Starting with an initial candidate solution, we iterate over a sequence of τ iterations. For each iteration, we generate a new candidate solution by flipping a single coordinate and take the best performing solution out of the two current ones. The “quality” of an intermediate solution \mathbf{y} is measured in terms of

$$F(\mathbf{y}) = \underset{\mathbf{c} \in \mathbb{R}^n}{\text{minimize}} J(\mathbf{c}, \mathbf{y}). \quad (8)$$

Once the overall process is finished, the best final candidate solution along with its corresponding vector \mathbf{c}^* is returned. For the generation of the initial candidate solution, we resort to a well-known heuristic in this field [48], i.e., we initialize the unlabeled patterns with the predictions provided via a supervised model (which is trained on the labeled part and determined by (4)). If the balance constraint is not fulfilled after this assignment, we use the largest positive (and real-valued) predictions of the model as “positive” class assignments and the remaining ones as “negative” class assignments. We will briefly investigate the benefits and drawbacks of such an initialization strategy in Section 4.

3.2 Convex Intermediate Tasks

The intermediate optimization task (8) for a fixed partition vector \mathbf{y} can be solved as follows: The function $G(\mathbf{c}) = J(\mathbf{c}, \mathbf{y})$ is differentiable with gradient

$$\nabla G(\mathbf{c}) = -2(\boldsymbol{\Lambda}\mathbf{K})^T(\boldsymbol{\Lambda}\mathbf{y} - \boldsymbol{\Lambda}\mathbf{K}\mathbf{c}) + 2\lambda\mathbf{K}\mathbf{c}.$$

Further, G is convex on \mathbb{R}^n since the kernel matrix \mathbf{K} and thus the Hessian

$$\nabla^2 G(\mathbf{c}) = 2(\boldsymbol{\Lambda}\mathbf{K})^T\boldsymbol{\Lambda}\mathbf{K} + 2\lambda\mathbf{K}$$

are positive semidefinite. Hence, $\nabla G(\mathbf{c}) = \mathbf{0}$ is a necessary and sufficient condition for optimality [3] and an optimal solution \mathbf{c}^* for (8) can be obtained via

$$\mathbf{c}^* = \mathbf{\Lambda}(\mathbf{\Lambda}\mathbf{K}\mathbf{\Lambda} + \lambda\mathbf{I})^{-1}\mathbf{\Lambda}\mathbf{y} = \mathbf{\Lambda}\mathbf{G}\mathbf{\Lambda}\mathbf{y}. \quad (9)$$

with $\mathbf{G} = (\mathbf{\Lambda}\mathbf{K}\mathbf{\Lambda} + \lambda\mathbf{I})^{-1}$. Note that $\mathbf{\Lambda}\mathbf{K}\mathbf{\Lambda}$ is positive semidefinite since \mathbf{K} is positive semidefinite; hence, $\mathbf{\Lambda}\mathbf{K}\mathbf{\Lambda} + \lambda\mathbf{I}$ is positive definite and thus invertible.

3.3 Efficient Matrix Updates

The recurrent computation of the objective values in Step 5 of Algorithm 1 is still cumbersome. However, similar to our previous work [9], it is possible to update these intermediate solutions efficiently when only one coordinate (or a constant number of coordinates) of an intermediate candidate solution is flipped: Let $\bar{\mathbf{y}}$ be the current candidate solution and let \mathbf{y} be its predecessor. By plugging in Equation (9) into (8), one gets

$$\begin{aligned} F(\bar{\mathbf{y}}) &= (\mathbf{\Lambda}\bar{\mathbf{y}} - \mathbf{\Lambda}\mathbf{K}\mathbf{c}^*)^T(\mathbf{\Lambda}\bar{\mathbf{y}} - \mathbf{\Lambda}\mathbf{K}\mathbf{c}^*) + \lambda(\mathbf{c}^*)^T\mathbf{K}\mathbf{c}^* \\ &= \bar{\mathbf{y}}^T \underbrace{\mathbf{\Lambda}(\mathbf{I} - \bar{\mathbf{K}}\mathbf{G} - \mathbf{G}\bar{\mathbf{K}} + \mathbf{G}\bar{\mathbf{K}}\mathbf{K}\mathbf{G} + \lambda\mathbf{G}\bar{\mathbf{K}}\mathbf{G})}_{=: \mathbf{H}} \mathbf{\Lambda}\bar{\mathbf{y}}, \end{aligned} \quad (10)$$

where $\bar{\mathbf{K}} = \mathbf{\Lambda}\mathbf{K}\mathbf{\Lambda}$. Now note that, for a given coordinate j to be flipped, one can update the right hand side via

$$\mathbf{H}\mathbf{\Lambda}\bar{\mathbf{y}} = \mathbf{H}\mathbf{\Lambda}\mathbf{y} - 2y_j(\mathbf{H}\mathbf{\Lambda})_{[n],\{j\}} \quad (11)$$

in $\mathcal{O}(n)$ time, assuming that the matrix $\mathbf{H}\mathbf{\Lambda} \in \mathbb{R}^{n \times n}$ and the information $\mathbf{H}\mathbf{\Lambda}\mathbf{y} \in \mathbb{R}^n$ for the predecessor are available (in memory). Further, since $\mathbf{\Lambda}$ is a diagonal matrix and since $\mathbf{H}\mathbf{\Lambda}\bar{\mathbf{y}} \in \mathbb{R}^n$ is a vector, the remaining operations for computing (10) can be performed in $\mathcal{O}(n)$ time too. Thus, by spending $\mathcal{O}(n^3)$ time for the initialization of the corresponding matrices, the solutions of the intermediate tasks can be “updated” in $\mathcal{O}(n)$ time per iteration:

Theorem 1. *One can compute $F(\bar{\mathbf{y}})$ in Step 5 of Algorithm 1 in $\mathcal{O}(n)$ time. Further, $\mathcal{O}(n^3)$ preprocessing time and $\mathcal{O}(n^2)$ space is needed.*

We would like to point out that it is possible to integrate the so-called *Nystrom approximation* in an efficient manner. More specifically, one can replace the original kernel matrix \mathbf{K} by $\tilde{\mathbf{K}} = (\mathbf{K}_R)^T(\mathbf{K}_{R,R})^{-1}\mathbf{K}_R$, where $R = \{i_1, \dots, i_r\} \subseteq [n]$ is a subset of indices. The acceleration can be done in an analogous way to the unsupervised case [9] by making use of the low-rank nature of $\tilde{\mathbf{K}}$; the longish derivations are omitted due to lack of space.

Theorem 2. *By applying the approximation scheme, one can compute $F(\bar{\mathbf{y}})$ in Step 5 of Algorithm 1 in $\mathcal{O}(r)$ time. Further, $\mathcal{O}(nr^2)$ preprocessing time and $\mathcal{O}(nr)$ space is needed.*

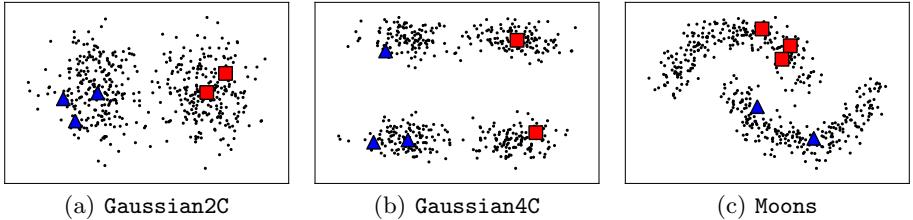


Fig. 2. The (red) squares and (blue) triangles depict the labeled part of the data; the remaining (small) points correspond to the unlabeled part

4 Experiments

We will now describe the experimental setup and the outcome of the evaluation.

4.1 Experimental Setup

Our approach (**S²RLSC**) is implemented in Python using the Numpy package. The runtime analyses are performed on a 2.66 GHz Intel Core™ Quad PC running Ubuntu 10.04.

Data Sets. The first artificial data set is composed of two Gaussian clusters; to generate it, we draw $n/2$ points from each of two Gaussian distributions $X_i \sim \mathcal{N}(\mathbf{m}_i, \mathbf{I})$, where $\mathbf{m}_1 = (-2.5, 0.0, \dots, 0.0) \in \mathbb{R}^d$ and $\mathbf{m}_2 = (+2.5, 0.0, \dots, 0.0) \in \mathbb{R}^d$. The class label of a point corresponds to the distribution it was drawn from, see Figure 2 (a). If not noted otherwise, we use $n = 500$ and $d = 500$ and denote the induced data set by **Gaussian2C**. The second artificial data set aims at generating a (possibly) misleading structure: Here, we draw $n/4$ points from each of four Gaussian distributions $X_i \sim \mathcal{N}(\mathbf{m}_i, \mathbf{I})$, where

$$\mathbf{m}_1 = (-2.5, -5.0, 0.0, \dots, 0.0) \in \mathbb{R}^d, \quad \mathbf{m}_2 = (-2.5, +5.0, 0.0, \dots, 0.0) \in \mathbb{R}^d, \\ \mathbf{m}_3 = (+2.5, -5.0, 0.0, \dots, 0.0) \in \mathbb{R}^d, \quad \mathbf{m}_4 = (+2.5, +5.0, 0.0, \dots, 0.0) \in \mathbb{R}^d,$$

see Figure 2 (b). The points drawn from the first two distributions belong to the first class and the remaining one to the second class. Again, we fix $n = 500$ and $d = 500$ and denote the corresponding data set by **Gaussian4C**. Finally, we consider the well-known **Moons** data set with $n = 500$ points and $\text{dim} = 2$, see Figure 2 (c). In addition to these artificial data sets, we make use of the **USPS** [10] data set, where **USPS(i, j)** is used to denote a single binary classification task, see Figure 3. If not noted otherwise, the first half of each data set is used as training and the second half as test set. To induce a semi-supervised scenario, we split up the training set into a labeled and an unlabeled part and use different ratios for the particular setting; the specific amount of data is given in brackets for each data set, where l, u, t denotes the number of labeled, unlabeled, and test patterns (e.g. **Gaussian2C[1=25, u=225, t=250]**).

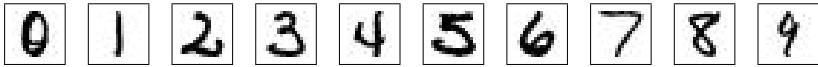


Fig. 3. The USPS data set [10] containing images of handwritten digits. Each digit is represented by a 16×16 gray scale image (8 bits).

Model Selection. To select the final models, several parameters need to be tuned. In a fully supervised setting, this is usually done via an extensive grid search over all involved parameters. However, in a semi-supervised setting, model selection is more difficult due to the lack of labeled data and is widely considered to be an open issue [5]. Due to this model selection problem, we consider two scenarios to select the (non-fixed) parameters: The first one is a *non-realistic* scenario where we make use of the test set to evaluate the model performance.² Note that by making use of the test set (with a large amount of labels), one can first evaluate the “flexibility” of the model, i.e., one can first investigate if the model is in principle capable of adapting to the inherent structure of the data while ignoring the (possible) problems caused by a small validation set. The second one is a *realistic* scenario where only the labels of the labeled part of the training set are used for model evaluation (via 5-fold cross-validation).

Parameters. In both scenarios, we first tune the non-fixed parameters via grid search and subsequently retrain the final model on the training set with the best performing set of parameters. As similarity measures, we consider both a linear kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ and a RBF kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2})$ with kernel width σ . To tune the cost parameters λ and λ' , we consider a small grid $(\lambda, \lambda') \in \{2^{-10}, \dots, 2^{10}\} \times \{0.01, 1, 100\}$ of parameters. Concerning the balance constraint, we set b_c to an estimate obtained from all available labels and fix $\varepsilon = 0.1$. The iterative process of the local search is stopped if no changes have occurred for n consecutive iterations; further, a round-robin scheme is used to select the coordinates to be flipped per iteration.

Competing Approaches. We use the regularized least-squares classifier (RLSC) as supervised model; to tune the parameter λ , we perform a grid search with $\lambda \in \{2^{-10}, \dots, 2^{10}\}$. As semi-supervised competitor, we resort to the UniverSVM approach [8], which depicts one of the state-of-the-art semi-supervised support vector machine implementations. The parameters are also tuned via grid search with $(C, C^*) \in \{2^{-10}, \dots, 2^{10}\} \times \{\frac{0.01}{u}, \frac{1.0}{u}, \frac{100.0}{u}\}$. The ratio between the two classes is provided to the algorithm via the `-w` option; except for the `-S` option (which we set to -0.3), the default values for the remaining parameters are used.

4.2 Results

We will now provide the outcome of the experiments conducted.

² This setup is usually considered in related evaluations [5].

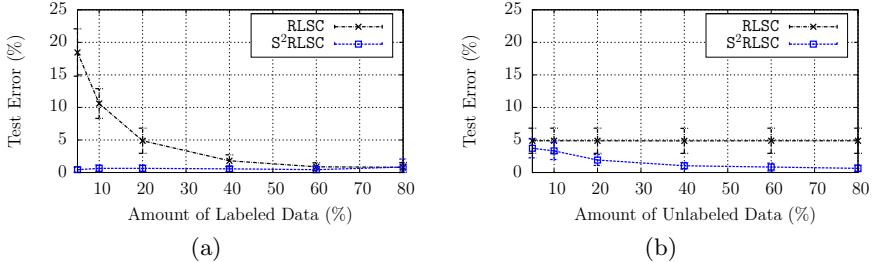


Fig. 4. Our semi-supervised approach can successfully incorporate unlabeled data to improve the generalization performance, see Figure (a). However, sufficient unlabeled data is needed as well for the learner to yield a satisfying performance, see Figure (b).

Amount of Data. For semi-supervised approaches, the amount of unlabeled data used for training is an important issue as well. To analyze how much labeled and unlabeled data is needed for our $S^2\text{RLSC}$ approach, we consider the Gaussian2C data set and vary the amount of labeled and unlabeled data. For this experiment, we consider the non-realistic scenario and use the supervised RLSC approach as baseline. First, we vary the amount of labeled data from 5% to 80% with respect to (the size of) the training set; the remaining part the training set is used as unlabeled data. In Figure 4(a), the result of this experiment is shown: Even with little labeled data, the semi-supervised approach performs better compared to the supervised one. Now, let us fix the amount of labeled data to 20% and and let us vary the amount of unlabeled data from 5% to 80% with respect to (the size of) the training set, see Figure 4(b). Clearly, the semi-supervised approach is only capable of generating an appropriate model once sufficient unlabeled data is given.

Classification Performance. We consider two different ratios of labeled and unlabeled data for each particular data set. The average test errors and the one standard deviation obtained on 10 random partitions of each data set into labeled, unlabeled, and test patterns are reported. For all data sets and for all competing approaches, a linear kernel is used. The results are given in Table II. It can be clearly seen that the semi-supervised approaches yield better results compared to the supervised model. The results also indicate that by taking more labeled data into account, the gap between the performances of the supervised approach and the semi-supervised approaches becomes smaller. Hence, both semi-supervised approaches can incorporate the unlabeled data successfully. Compared to the results for the non-realistic scenario, the performances for the realistic one are clearly worse. Hence, the lack of labeled data for model selection as well as a (possibly) bad estimate for the balance parameter b_c seem to have a negative influence on the final classification performance. The overall classification performances of both semi-supervised approaches is comparable; the $S^2\text{RLSC}$ approach seems work slightly better on the USPS data set whereas the **UniverSVM** approach works slightly better on the artificial data sets in the realistic scenario.

Table 1. The table shows the classification performance of all approaches on all data sets considered. Clearly, both semi-supervised approaches can successfully incorporate unlabeled data to improve the performance on the test set.

| Data Set | RLSC | | UniverSVM | | $S^2\text{RLSC}$ | |
|--------------------------------|---------------|------------|---------------|------------------|-------------------|--------------------|
| | non-realistic | realistic | non-realistic | realistic | non-realistic | realistic |
| Gaussian2C[l=25,u=225,t=250] | 10.6 ± 2.3 | 11.4 ± 2.4 | 1.0 ± 0.5 | 1.8 ± 0.9 | 0.6 ± 0.5 | 6.6 ± 2.6 |
| Gaussian2C[l=50,u=200,t=250] | 4.9 ± 1.9 | 5.3 ± 2.1 | 1.0 ± 0.4 | 1.8 ± 0.8 | 0.6 ± 0.5 | 3.3 ± 2.6 |
| Gaussian4C[l=25,u=225,t=250] | 16.1 ± 7.0 | 16.5 ± 6.8 | 7.6 ± 12.2 | 13.3 ± 15.2 | 6.8 ± 12.5 | 12.0 ± 13.6 |
| Gaussian4C[l=50,u=200,t=250] | 6.1 ± 2.1 | 6.5 ± 1.9 | 1.6 ± 0.8 | 2.5 ± 1.5 | 0.8 ± 0.6 | 4.1 ± 2.4 |
| USPS(2,5) [l=16,u=806,t=823] | 7.9 ± 2.9 | 9.3 ± 2.3 | 3.2 ± 0.5 | 9.0 ± 5.6 | 2.9 ± 0.4 | 6.3 ± 2.8 |
| USPS(2,5) [l=32,u=790,t=823] | 4.4 ± 0.5 | 5.8 ± 1.6 | 3.2 ± 0.5 | 5.7 ± 1.8 | 2.9 ± 0.4 | 6.4 ± 2.0 |
| USPS(2,7) [l=17,u=843,t=861] | 3.6 ± 2.4 | 4.0 ± 2.4 | 1.5 ± 0.3 | 6.1 ± 5.3 | 1.0 ± 0.1 | 1.4 ± 0.2 |
| USPS(2,7) [l=34,u=826,t=861] | 2.2 ± 0.7 | 3.1 ± 1.7 | 1.4 ± 0.2 | 3.4 ± 2.4 | 1.0 ± 0.1 | 1.3 ± 0.2 |
| USPS(3,8) [l=15,u=751,t=766] | 9.8 ± 6.6 | 11.0 ± 6.4 | 4.8 ± 1.1 | 8.7 ± 3.9 | 4.1 ± 1.3 | 6.2 ± 2.7 |
| USPS(3,8) [l=30,u=736,t=766] | 6.3 ± 2.0 | 7.7 ± 1.4 | 4.0 ± 0.1 | 7.1 ± 1.8 | 3.9 ± 1.2 | 6.4 ± 2.8 |
| USPS(8,0) [l=22,u=1108,t=1131] | 4.8 ± 1.9 | 6.6 ± 3.3 | 1.7 ± 0.7 | 3.2 ± 2.2 | 1.2 ± 0.2 | 1.8 ± 0.6 |
| USPS(8,0) [l=45,u=1085,t=1131] | 2.6 ± 0.8 | 3.3 ± 0.9 | 1.3 ± 0.4 | 3.3 ± 1.8 | 1.1 ± 0.2 | 2.0 ± 0.5 |

Computational Considerations. Let us finally analyze the practical runtimes of the considered semi-supervised approaches. Naturally, these runtimes depend heavily on the particular implementation and the used programming language. Thus, the provided results shall only give a rough idea of the runtimes needed in practice (e.g., for generating Table 1). For this sake, we consider the Gaussian2C and the USPS(8,0) data sets. Again, we resort to a linear kernel and fix the model parameters ($\lambda = 1$ and $\lambda' = 1$ for $S^2\text{RLSC}$ and $C = 1$ and $C^* = 1$ for UniverSVM). Further, the amount of labeled patterns is fixed to $l = 50$ and $l = 22$, respectively. In Figure 5, the runtime behavior for both approaches for a varying amount of unlabeled patterns is given. The plots indicate a comparable runtime performance of both approaches. Thus, the computational shortcut renders our simple local search scheme competitive to state-of-the-art implementations.³

More Optimization. Most of the related optimization schemes use the labeled part to obtain an initial “guess” via a supervised model (e.g., UniverSVM) which is then improved iteratively (i.e., no restarts are performed rendering such approaches *deterministic*). While such a strategy reduces the practical runtime, a natural question is whether the results can be improved by putting more effort into optimization. To exemplarily investigate this question, we consider the Moons data set with RBF kernel as similarity measure (and kernel width $\sigma = 0.1s$, where the value s is an estimate of the maximum distance between any pair of samples). Further, we fix the two cost parameters to $\lambda = 2^{-10}$ and $\lambda' = 0.1$. The comparison between the single restart approach (used above) and a multiple restart scheme (with 200 restarts and random initialization) is shown in Figure 6. Clearly, the single restart variant fails on this particular problem instance while using more restarts leads to the global optimum. Thus, there are

³ The UniverSVM implementation is based on C whereas $S^2\text{RLSC}$ is implemented in Python; in general, code in Python is said to be much slower than pure C code and we expect a considerable speed-up with a pure C implementation of our approach.

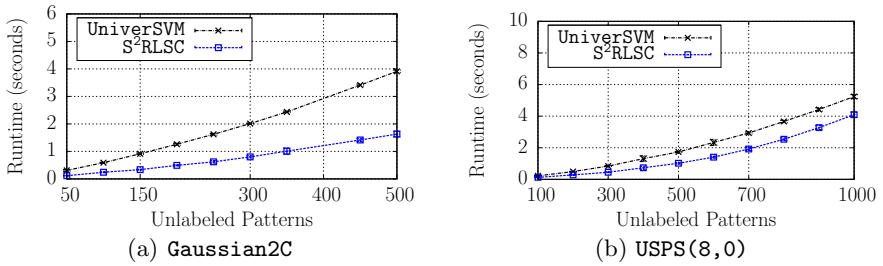


Fig. 5. Runtimes of the semi-supervised competitors on the Gaussian2C and the USPS(8,0) data sets; in both cases, the average runtimes of 10 executions are reported

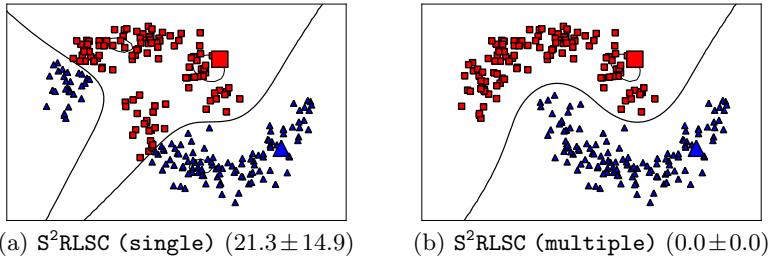


Fig. 6. If only one restart (with initial guess) is performed, the local search approach can converge to a (bad) local optimum, see Figure (a). Performing sufficient restarts (with random initial candidate solutions) yields good solutions, see Figure (b). The average test error (with one standard deviation) on over 10 random partitions is given.

problem instances where the heuristic of improving a single guess fails and where it pays off to spend more effort into optimization. We would like to point out that for such settings, the proposed approach is well suited since performing restarts is very cheap (preprocessing has only to be done once).

5 Conclusions and Outlook

We proposed an optimization framework for semi-supervised regularized least-squares classification. The key idea consists in making use of a simple local search strategy which is accelerated by means of efficient matrix updates for the intermediate candidate solutions. Our experimental evaluation demonstrates that such a simple (but accelerated) search scheme yields classification results comparable to state-of-the-art methods in an extremely efficient manner.

We think that the derivations presented in this work are extendible and applicable in various directions: For instance, Adankon *et al.* [1] have recently proposed an alternating optimization scheme (also based on the square loss) which can potentially be accelerated by means of the matrix-based updates given here. Further, we expect the computational shortcuts to be beneficial in the context of

(exact) branch-and-bound strategies for the task at hand like the one proposed by Chapelle *et al.* [6]. We plan to investigate these issues in near future.

References

1. Adankon, M., Cheriet, M., Biem, A.: Semisupervised least squares support vector machine. *IEEE Transactions on Neural Networks* 20(12), 1858–1870 (2009)
2. Bie, T.D., Cristianini, N.: Convex methods for transduction. In: *Adv. in Neural Information Proc. Systems*, vol. 16, pp. 73–80. MIT Press, Cambridge (2004)
3. Boyd, S., Vandenberghe, L.: *Convex Optimization*. Cambridge Uni. Press, Cambridge (2004)
4. Chapelle, O., Chi, M., Zien, A.: A continuation method for semi-supervised SVMs. In: *Proc. Int. Conf. on Machine Learning*, pp. 185–192 (2006)
5. Chapelle, O., Schölkopf, B., Zien, A. (eds.): *Semi-Supervised Learning*. MIT Press, Cambridge (2006)
6. Chapelle, O., Sindhwani, V., Keerthi, S.S.: Branch and bound for semi-supervised support vector machines. In: *Adv. in Neural Information Proc. Systems*, vol. 19, pp. 217–224. MIT Press, Cambridge (2007)
7. Chapelle, O., Zien, A.: Semi-supervised classification by low density separation. In: *Proc. 10th Int. Workshop on Artificial Intell. and Statistics*, pp. 57–64 (2005)
8. Collobert, R., Sinz, F., Weston, J., Bottou, L.: Trading convexity for scalability. In: *Proc. International Conference on Machine Learning*, pp. 201–208 (2006)
9. Gieseke, F., Pahikkala, T., Kramer, O.: Fast evolutionary maximum margin clustering. In: *Proc. Int. Conf. on Machine Learning*, pp. 361–368 (2009)
10. Hastie, T., Tibshirani, R., Friedman, J.: *The Elements of Statistical Learning*. Springer, Heidelberg (2009)
11. Joachims, T.: Transductive inference for text classification using support vector machines. In: *Proc. Int. Conf. on Machine Learning*, pp. 200–209 (1999)
12. Rifkin, R., Yeo, G., Poggio, T.: Regularized least-squares classification. In: *Adv. in Learning Theory: Methods, Models and Applications*. IOS Press, Amsterdam (2003)
13. Schölkopf, B., Herbrich, R., Smola, A.J.: A generalized representer theorem. In: Helmbold, D.P., Williamson, B. (eds.) *COLT 2001 and EuroCOLT 2001. LNCS (LNAI)*, vol. 2111, pp. 416–426. Springer, Heidelberg (2001)
14. Sindhwani, V., Keerthi, S., Chapelle, O.: Deterministic annealing for semi-supervised kernel machines. In: *Proc. Int. Conf. on Machine Learning*, pp. 841–848 (2006)
15. Steinwart, I., Christmann, A.: *Support Vector Machines*. Springer, New York (2008)
16. Vapnik, V., Sterin, A.: On structural risk minimization or overall risk in a problem of pattern recognition. *Aut. and Remote Control* 10(3), 1495–1503 (1977)
17. Zhang, K., Kwok, J.T., Parvin, B.: Prototype vector machine for large scale semi-supervised learning. In: *Proceedings of the International Conference on Machine Learning* (2009)
18. Zhu, X., Goldberg, A.B.: *Introduction to Semi-Supervised Learning*. Morgan and Claypool (2009)

Model-Based Object Recognition from 3D Laser Data

Martin Günther, Thomas Wiemann, Sven Albrecht, and Joachim Hertzberg

University of Osnabrück, 49069 Osnabrück, Germany

Abstract. This paper presents a method for recognizing objects in 3D point clouds. Based on a structural model of these objects, we generate hypotheses for the location and 6DoF pose of these models and verify them by matching a CAD model of the object into the point cloud. Our method only needs a CAD model of each object class; no previous training is required.

Keywords: object recognition, 3D point clouds, OWL-DL ontology, CAD model matching, semantic mapping.

1 Introduction

Learning maps of a previously unknown environment is a classical task for mobile robots. The large body of literature about the topic would address it mostly as Robotic Mapping or as Simultaneous Localization and Mapping (SLAM). Traditionally, robot maps contain landmarks and/or environment structures and the according geometry information in 2D or 3D only, as they are mostly used for planning collision-free paths and for localization. In more recent work, semantic categories in maps are becoming more and more important: These may be, for example, classes of objects contained in the map, classes of rooms, or topological regions. Semantic information in maps promises to be of help for both the robot control itself and as a basis for human-robot or robot-robot interaction.

A technical challenge for semantic mapping approaches is to extract the semantic categories from the raw sensor data, and to do so in real time on board the robot. Most previous object recognition approaches are appearance-based: Based on a 2D image or 3D point cloud of the scene, features are extracted and passed into a classifier that has previously been trained using labeled training examples.

However, in recent years, CAD models of many types of objects have become widely available. From the CAD model of an object, a declarative, structural model of the object in terms of its primitive geometric constituents (planar patches, cylinders, spheres) and their spatial interrelations can be obtained. This offers the possibility of a complementary, model-based approach to object recognition: Instead of classifying objects by their appearance, we extract the geometric primitives found in the raw sensor data and match them to the structural models of our objects.

Our approach extends and goes beyond previous work [10], where coarse structures like walls, doors, ceiling and floor were labeled. Here, we take this idea one step further and extend it to the recognition of medium-scale objects. We describe a system that recognizes different types of furniture in 3D laser scans and present first empirical results.

2 Related Work

Previous work on model-based object recognition in 3D point clouds has been carried out by Rusu et al. [15][16]. They present a complete semantic mapping framework based on 3D laser data; in contrast to our work, the model fitting part uses cuboids of variable dimensions instead of CAD models.

Most other work in object-recognition is appearance-based (in the sense that no structural model of the object is used, although some use CAD models). Lai and Fox [6] use sampled CAD models from Google 3D Warehouse to train an object detection system used to label objects in urban 3D laser scans. Mian et al. [9] use a geometric hashing scheme to recognize objects by matching CAD models into a 3D point cloud.

Several methods [18][14][20] for extracting interest points and creating stable descriptors based on 3D shape have been proposed which can be used to recognize objects in 3D point clouds. Stiene et al. [19] detect objects in range images using an Eigen-CSS method on the contours of objects, combined with a supervised learning algorithm. In a similar setting, Nüchter et al. [11] use Haar-like features together with AdaBoost for object detection.

An idea similar in spirit to our approach is proposed in [5] where matching of CAD models is utilized to allow for manipulation tasks such as grasping in a household environment. Contrary to our approach, the matching itself is performed within 2D image data, instead of the 3D environment representation.

A robot manipulation system that integrates knowledge processing mechanisms with semantic perception routines, including CAD model matching, is presented in [12].

Part of our method requires the use of spatial relational reasoning, which is often done using constraint calculi [13]. Our task, however, differs from that, because we consider the spatial relations between geometric primitives as part of the definition of an aggregated object. For this reason, we use ontological reasoning for this task. Another advantage of this approach is that numerical ranges can be used via SWRL rules.

3 Model-Based Object Recognition

In recent years, CAD models of many kinds of objects have become widely available. One resource of CAD models is Google's 3D Warehouse, which allows querying and retrieving CAD models of virtually any kind of object via the web. In the domain of furniture recognition, CAD models are often available directly from the manufacturer or from companies specialized in creating CAD

models for interior designers. We use a database of CAD models supplied by our university's furniture manufacturer.

In this paper, we focus on the domain of furniture recognition for several reasons: First, due to the widespread use of CAD models in interior design, the availability of CAD models in this domain is especially strong. Second, most kinds of furniture feature a set of planar surfaces which can be robustly recognized in 3D laser scans. Third, due to the rigidness of furniture, these planar surfaces are in a clearly defined relation to each other.

Figure 1 shows the embedding of our system in a general semantic mapping framework. We see our model-based object recognition method as complementary to appearance-based methods based on 2D image features or 3D shape features.

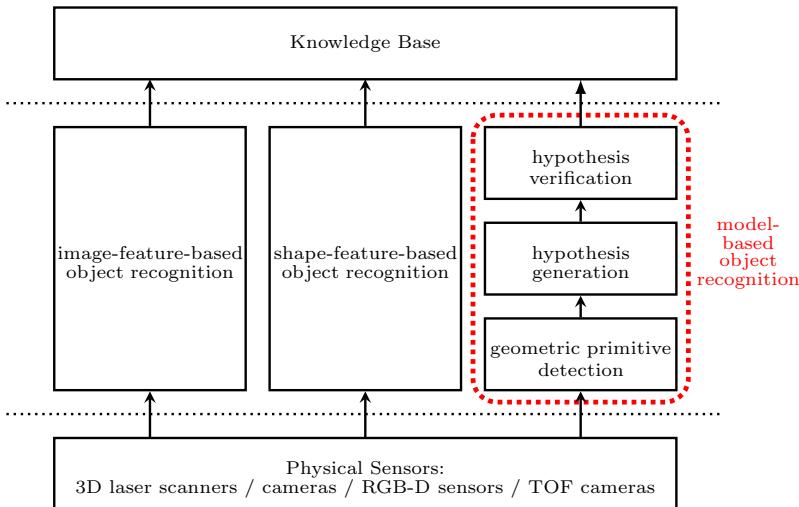


Fig. 1. System overview. While the present paper is focused on model-based object recognition, we consider this method as yielding complementary information to standard recognition methods. So in a more general system architecture, they may well co-exist. We will not deepen this issue here.

Using the information contained in CAD models for object recognition has several advantages. Instead of having one classifier for each kind of object, only the geometric primitives have to be detected. Based on these, objects are reconstructed. Also, no classifier training and no labeled training sets are required; to add a new object class, only the CAD model is required. In the future, it would even be conceivable that such a CAD model could be retrieved on-line from the web. Another advantage is that once an object is recognized, the corresponding part of the sensor data can be replaced by the CAD model, thus filling up occlusions in the sensor data.

On the other hand, appearance-based methods have an advantage where the to-be-recognized object is non-rigid, does not consist of clearly identifiable

geometric primitives of a certain minimum size or where labeled training data, but no CAD model is available.

Our model-based object recognition method consists of three parts, which will be detailed in the remainder of this section: geometric primitive detection, which extracts the geometric primitives from the sensor data; object hypothesis generation, which classifies the geometric primitives to find potential object locations; and object hypothesis verification, where these potential matches are accepted or rejected.

3.1 Geometric Primitive Detection

The first step of the object recognition procedure is to extract geometric primitives from the 3D point cloud. In our approach we rely on planar patches, since they are most relevant for detecting furniture. In the remainder of this paper we will refer to these patches simply as “planes”. However, our approach could be extended to other kinds of geometric primitives, such as cylinders or spheres.

The idea of our plane extraction algorithm is the following: First, a triangle mesh of the scanned scene using an optimized marching cubes [7] implementation is generated. Within this mesh, connected planar regions are extracted using a region growing approach. A detailed description of our procedure can be found in [21]. Figure 2 shows two exemplary results of this procedure: The left picture shows a reconstruction of a closable office shelf with one side open. The right picture shows a filing cabinet. Both reconstructions are based on 3D laser scans taken with a tilting SICK LMS 200 laser scanner (about 156.000 data points per scan). The extracted planes are shown in different colors. Non-planar regions in the mesh that were not considered by our algorithm are rendered in green. As one can see, the relevant planes for our purposes can clearly be distinguished. For each extracted plane, we save the characteristic information that is needed for the OWL-DL reasoner to generate object hypotheses. The relevant key figures for our process are size of the plane, centroid and surface normal.

3.2 Object Hypothesis Generation

Semantic knowledge about identified objects is stored using an OWL-DL ontology in combination with SWRL rules, which will be used to generate hypotheses of possible object locations and initial pose estimation, based on the planes extracted in the previous section.

OWL, the Web Ontology Language, is the standard proposed by the W3C consortium as the knowledge representation formalism for the Semantic Web. It consists of three sublanguages (OWL-Full, OWL-DL and OWL-Lite). The sublanguage OWL-DL [3] corresponds to a Description Logic, a subset of First-Order Logic, allowing to use many expressive features while guaranteeing decidability (in polynomial time). It has been extended by SWRL, the Semantic Web Rule Language [4], which allows to use Horn-like in combinations with an OWL-DL knowledge base and includes so-called built-ins for arithmetic comparisons and calculations. OWL-DL was chosen as the knowledge representation format for

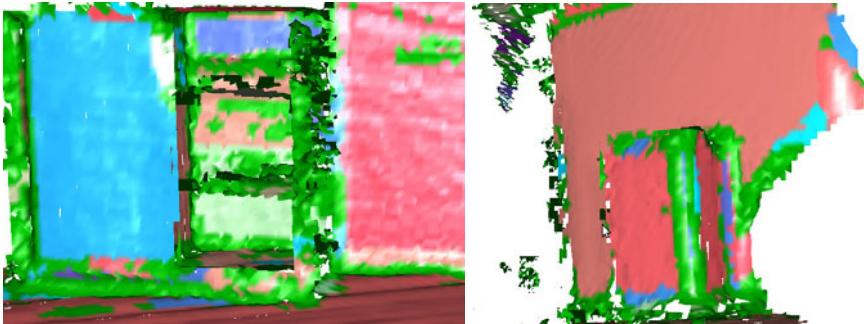


Fig. 2. Two examples for plane extraction. Detected planes are colored individually; non-planar surfaces are colored in green. Left: closable shelf (sliding doors) in front view. Right: a filing cabinet. The pictures show that our plane extraction algorithm is able to extract the relevant geometric information for our model based object recognition procedure.

this paper for several reasons: OWL-DL ontologies can be easily re-used and linked with other sources of domain knowledge from the Semantic Web, they easily scale to arbitrarily large knowledge bases, and fast reasoning support is available. In our implementation, we use the open-source OWL-DL reasoner Pellet [17], which provides full support for OWL-DL ontologies using SWRL rules.

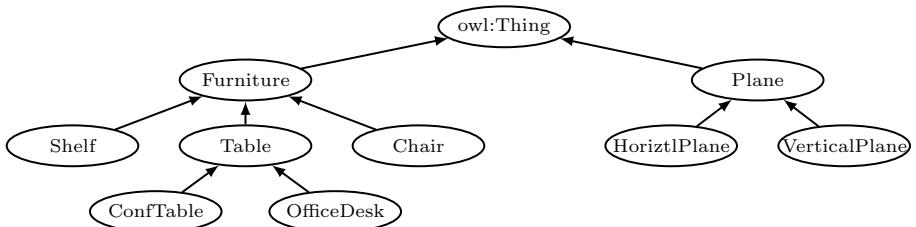


Fig. 3. The parts of the ontology's class hierarchy relevant for the examples in this paper. For our approach, we distinguish between horizontal and vertical planes (right branch). The relations between different kinds of furniture are modeled in the left branch.

Figure 3 shows part of the ontology used in this paper. The right part models the geometric primitives (here: horizontal and vertical planes). Each plane extracted in the previous section is added as an individual to the ontology, along with its orientation (horizontal/vertical, based on the normal), its height above ground (based on the centroid), its bounding box and its area. Additionally, we add two different spatial relations between planes, based on the centroid and surface normal which have been extracted in the previous subsection, as OWL properties. The property *isAbove* is added between two horizontal planes if the

distance of their centroids, projected onto the ground plane, is below a certain threshold. Likewise, the property *isPerpendicular* is added between a horizontal and a vertical plane if their centroid distance is below the threshold.

The definitions of furniture classes in the ontology (the left part of Fig. 3) contain a set of conditions that are used to classify the planes into possible furniture instances. For example, most standard desks have a height of approximately 70 cm. So all horizontal planes that have about this height and have a certain minimal size are valid candidates for table tops. This can be expressed by adding the following SWRL rule to the ontology:

$$\begin{aligned} \text{Table}(\textit{?p}) \leftarrow & \text{HorizontalPlane}(\textit{?p}) \wedge \text{hasSize}(\textit{?p}, \textit{?s}) \\ & \wedge \text{swrlb:greaterThan}(\textit{?s}, 1.0) \wedge \text{hasPosY}(\textit{?p}, \textit{?h}) \\ & \wedge \text{swrlb:greaterThan}(\textit{?h}, 0.65) \wedge \text{swrlb:lessThan}(\textit{?h}, 0.85) \end{aligned}$$

Similar considerations apply to office chairs: A chair has a ground parallel plane to sit on (at a height of around 40 cm) and another perpendicular plane near it (the backrest). Figure 4 presents, as an example, the ontology representation of a shelf. The ranges in the properties reflect possible reconstruction errors or modifications of the actual object that were not represented in the original CAD model. At the moment, these structural object models are encoded into OWL-DL by hand; in the future, these could be extracted automatically from the CAD model or generalized from a set of CAD models.

For each object instance returned by the OWL-DL reasoner, we calculate axis-parallel bounding boxes and center points of the constituting planes. The center point of one predefined plane (e.g., the table top) is used to anchor the position. Information about the orientation depends on the geometry of the expected models. The intrinsic orientation has to be identified and encoded according to the model class. For some objects this orientation is easy to identify, e.g., chairs where the normal of the back rest defines the orientation of the whole object. For other objects like tables, we apply a Principal Component Analysis (PCA) to the points that lie within the plane that defines the intrinsic orientation. This method delivers two new orthonormal basis vectors that approximate the orientation within the global coordinate system. For successful matching, all used models have to be pre-processed to be in a center-point-based local coordinate system that reflects the assumptions described above.

3.3 Object Hypothesis Verification

In order to verify the generated hypotheses of Section 3.2 we match an appropriate CAD model for each hypothesis with the point cloud data. Creating a 3D point surface sampling for a given CAD model yields a point cloud retaining the geometric properties of the model. This point cloud in combination with the initial pose obtained in Section 3.2 allows for a straight forward application of the ICP [2] algorithm. For more details concerning the surface sampling please refer to [1]. Since ICP converges in a local minimum, the average correspondence

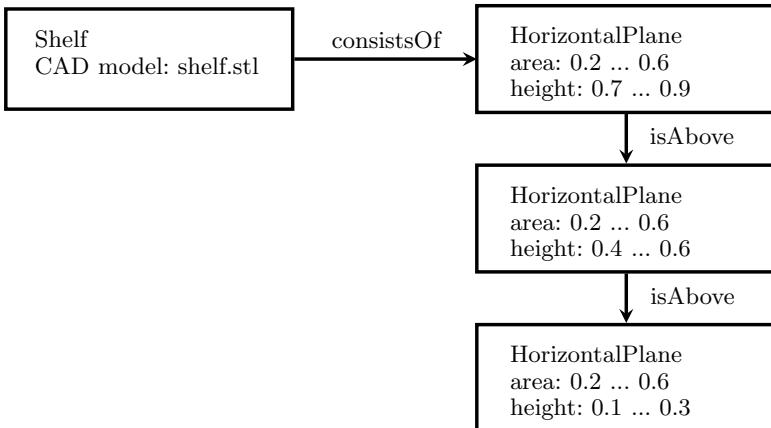


Fig. 4. A fragment of the ontology representing a shelf. The range intervals for the different properties take account of user modifications (someone could have lifted a shelf board) and reconstruction errors.

error between sampled model and scene point cloud can already give a rough estimate about the quality of the match, i.e., if a hypothesis could be verified or not.

Although this approach might correctly reject some false hypotheses, many scenarios are plausible in which the average correspondence error between sampled CAD model and data point cloud is small but still the object is not present in the scene. To have a better estimate to determine if an object is present in the data points we therefore propose another heuristic measure. For this heuristic we change the perspective compared to the ICP algorithm used to obtain the final pose: Instead of fitting the sampled model to the point cloud data we now check how closely the point cloud data resembles the sampled model data in the model's final pose. To this end we discretize the model data into voxels, thus creating a more coarse representation of the model. Now we check how many data points of the scan data are contained in each of these voxels in the final pose determined by ICP. If the number of data points in such a voxel is larger than a given threshold, we assume that this part of the model was present in the scan data, otherwise the point cloud locally does not fit. Once this process is done for each voxel, we compare the ratio of voxels resembled in the scan to voxels not present in the scan data. If this ratio is above a given threshold we assume that the model was present in the scan data.

However a more sophisticated measure to compare the CAD model in its final pose with its surroundings is topic of ongoing research.

4 Experimental Results

For our test scenario, we use a database of CAD models that is directly available from our university's furniture manufacturer. We present an example for

automatically recognizing furniture using our model-based object recognition method: Finding two office table CAD models in a 3D point cloud of an office. The first part displays the instantiation of the object hypotheses from our hypothesis generation method. The second part shows pose refinement for these instances derived from the ICP-based hypothesis verification step.

4.1 Hypothesis Generation

The input to our reasoner is a set of planes that were extracted from the input data. Figure 5 shows the input point set for our experiments together with a surface reconstruction. The data was obtained using a SICK LMS-200 laser scanner mounted on a rotational unit. Several scans from different positions were registered via ICP into a single point cloud. The planar structures found in the scene are rendered in a red to blue gradient, all other surfaces are green. The basic characteristics of these patches (centroid, normal, bounding box, area) are used by the reasoner to identify possibly present models. The current implementation of our plane extraction procedure is highly optimized for parallel processing and scales well with the number of CPU cores. The objects in the presented data set are extracted in less than 4 seconds on a Intel Quad Core processor, including normal estimation for the data points, mesh generation and plane extraction. This time is in the order of magnitude that it takes to capture a single 3D laser scan with our equipment.

As one can see, the large connected surfaces on the floor are recognized, as well as smaller structures like the tabletops (gray) or the backrests of the chairs around the conference table (red, blue, light green). After feeding the extracted planes into the reasoner, two possible present objects were detected: The conference table on the right and the desk on the left. The main remaining problem is

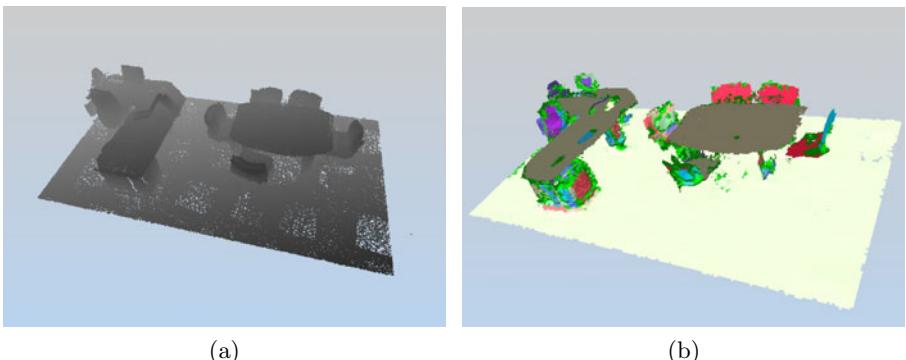


Fig. 5. The used input point cloud (a) and the automatically extracted planes (b). The detected planes are colored in a red to blue gradient based on a running number. All surfaces that were not classified as belonging to a plane are rendered in green. Walls and ceiling in the original data set were manually removed to create a suitable perspective.

to determine the model's orientation. To solve this, we use a PCA implementation by Martagh [8] on the vertices of the table top reconstruction. To analyze the stability of this approach, we rotated a reference model of the conference table to different predefined angles to get ground truth and compared the original rotation angles with the PCA estimation. The results are shown in Table II. The time for PCA computation for the considered planes is negligible for our application (some 100 milliseconds). Although the estimated poses derived from the bounding box show several degrees difference from ground truth, we were able to correct these deviations automatically via ICP and confirm the detected objects in the scanned scene as shown in the following section.

Table 1. Estimated orientations for two table models via PCA compared to ground truth

| Ground Truth | 12.0° | 25.0° | 55.0° | 90.0° | 125.0° | 160.0° |
|------------------|-------|-------|-------|-------|--------|--------|
| Conference Table | 8.2° | 22.1° | 51.4° | 86.0° | 121.0° | 157.0° |
| Office Desk | 4.0° | 28.1° | 46.7° | 82.0° | 118.0° | 153.0° |

4.2 Hypothesis Verification and Model Replacement

After the object hypotheses and pose estimations are generated, we subsample the corresponding CAD models. This synthetic point cloud is then used to refine the initial pose estimation. Figure G shows the results of the matching process for the tables that were detected in the given scene. The pictures on the left clearly show that the ICP process significantly improves the estimated poses. For the conference table we get an almost perfect fit. The fit for the office desk is not as good as the one for the conference table. Here we have an offset to the right of about two centimeters. This is due to registration errors in the used point cloud and differences between the CAD model and the real world object. The real object shows clearances that are not considered in the model.

These two examples show that our ICP based object hypotheses verification procedure is able to instantiate the presumed objects from the OWL-DL reasoner. This instantiation provides additional semantic knowledge about the scanned environment, namely that there are an office table and a conference table present. Furthermore, the replacement of the original point cloud data with the appropriate CAD models of the recognized objects can be used to enhance the initial sensor data, e.g. by filling in missing data points from laser shadows by sampling the surfaces of the CAD model.

This fact shows another advantage of model based object recognition over appearance based methods. These methods usually do not encode the whole geometric information of the trained objects, only the abstract characteristic feature descriptors for the used machine learning algorithm. Although a link between these features and more detailed object descriptions is feasible, the handling of

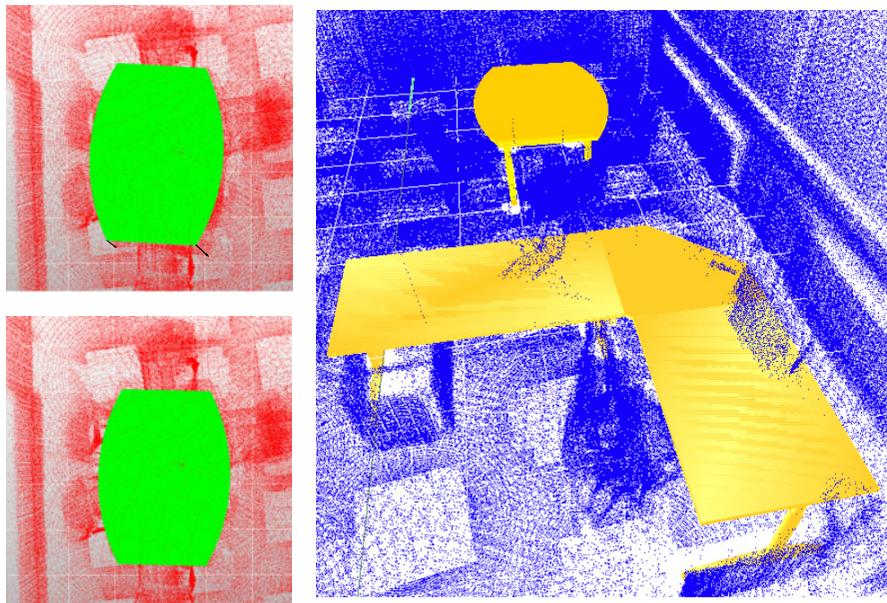


Fig. 6. Results of the ICP model matching process for the two table models. The left column shows the pose of the conference table before and after matching. The offset of the initial pose estimation from the final pose is indicated by the black arrows at the lower edge of the table. The picture on the right shows the CAD models of both detected tables rendered in the original point cloud.

these relations requires additional efforts. With our model based approach we have the geometric properties already encoded in the CAD models themselves.

5 Summary and Future Work

We have presented initial results on a model-based method for recognizing furniture objects in 3D point clouds, based on CAD models, and demonstrated the viability of this approach on a real-world example.

In the future, we plan to extend this approach in several directions: First, we plan to explore alternative representation formalisms for the object hypothesis generation step. In particular, Statistical Relational Models offer themselves here due to their potential robustness to occlusions or false positives in the sensor data. Second, more work needs to be done on the error function of the hypothesis verification step. Third, we intend to expand the approach to articulated furniture (such as a cabinet with sliding or hinged doors, chairs and tables with adjustable height) and variability (such as a bookshelf with a variable number of shelves). One way to do this would be to use parametric CAD models. Fourth and last, we plan to automate the extraction of OWL-DL structural models for hypothesis generation from the CAD models.

References

1. Albrecht, S., Wiemann, T., Günther, M., Hertzberg, J.: Matching CAD object models in semantic mapping. In: Proc. ICRA 2011 Workshop: Semantic Perception, Mapping and Exploration, SPME 2011, Shanghai, China (2011)
2. Besl, P., McKay, N.: A method for registration of 3-D shapes. *IEEE T. Pattern Anal. Mach. Intell.* 14(2), 239–256 (1992)
3. Dean, M., Schreiber, G., Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A.: OWL web ontology language reference. W3C recommendation, W3C (February 2004),
<http://www.w3.org/TR/owl-ref/>
4. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: A semantic web rule language combining OWL and RuleML. W3C member submission, World Wide Web Consortium (2004),
<http://www.w3.org/Submission/SWRL>
5. Klank, U., Pangercic, D., Rusu, R.B., Beetz, M.: Real-time cad model matching for mobile manipulation and grasping. In: 9th IEEE-RAS Intl. Conf. on Humanoid Robots, Paris, France (December 7-10, 2009)
6. Lai, K., Fox, D.: Object recognition in 3D point clouds using web data and domain adaptation. *Int. J. Robot. Res.* 29(8), 1019–1037 (2010)
7. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3D surface construction algorithm. In: Proc. ACM SIGGRAPH (1987)
8. Martagh, F., Heck, A.: Multivariate Data Analysis. Kluwer Academic, Dordrecht (1987)
9. Mian, A.S., Bennamoun, M., Owens, R.A.: Three-dimensional model-based object recognition and segmentation in cluttered scenes. *IEEE T. Pattern Anal. Mach. Intell.* 28(10), 1584–1601 (2006)
10. Nüchter, A., Hertzberg, J.: Towards semantic maps for mobile robots. *Robot. Auton. Syst., Special Issue on Semantic Knowledge in Robotics* 56(11), 915–926 (2008)
11. Nüchter, A., Surmann, H., Hertzberg, J.: Automatic classification of objects in 3D laser range scans. In: Proc. 8th Conf. on Intelligent Autonomous Systems (IAS 2004), Amsterdam, The Netherlands, pp. 963–970 (March 2004)
12. Pangercic, D., Tenorth, M., Jain, D., Beetz, M.: Combining perception and knowledge processing for everyday manipulation. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, Taipei, Taiwan (October 18-22, 2010)
13. Renz, J., Nebel, B.: Qualitative spatial reasoning using constraint calculi. In: Aiello, M., Pratt-Hartmann, I., van Benthem, J. (eds.) *Handbook of Spatial Logics*, pp. 161–215. Springer, Heidelberg (2007)
14. Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (fpfh) for 3d registration. In: Intl. Conf. on Robotics and Automation (ICRA), pp. 3212–3217. IEEE, Kobe (2009)
15. Rusu, R.B., Marton, Z.C., Blodow, N., Dolha, M.E., Beetz, M.: Towards 3D point cloud based object maps for household environments. *Robot. Auton. Syst., Special Issue on Semantic Knowledge in Robotics* 56(11), 927–941 (2008)
16. Rusu, R.B., Marton, Z.C., Blodow, N., Holzbach, A., Beetz, M.: Model-based and learned semantic object labeling in 3D point cloud maps of kitchen environments. In: 2009 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, October 11-15, pp. 3601–3608. IEEE, St. Louis (2009)
17. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. *J. Web Sem.* 5(2), 51–53 (2007)

18. Steder, B., Rusu, R.B., Konolige, K., Burgard, W.: Point feature extraction on 3D range scans taking into account object boundaries. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), Shanghai, China (May 2011)
19. Stiene, S., Lingemann, K., Nüchter, A., Hertzberg, J.: Contour-based object detection in range images. In: Proc. 3rd Intl. Symposium on 3D Data Processing, Visualization and Transmission, 3DPVT 2006, Chapel Hill, NC, USA (June 2006)
20. Unnikrishnan, R.: Statistical Approaches to Multi-Scale Point Cloud Processing. Ph.D. thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA (May 2008)
21. Wiemann, T., Nüchter, A., Lingemann, K., Stiene, S., Hertzberg, J.: Automatic construction of polygonal maps from point cloud data. In: Proc. 8th IEEE Intl. Workshop on Safety, Security, and Rescue Robotics (SSRR 2010), Bremen, Germany (July 2010)

Swarm Intelligence for Medical Volume Segmentation: The Contribution of Self-reproduction

Robert Haase¹, Hans-Joachim Böhme², Daniel Zips³, and Nasreddin Abolmaali^{1,4}

¹ Biological and Molecular Imaging Group, OncoRay, National Center for Radiation Research in Oncology, Medical Faculty Carl Gustav Carus, TU Dresden, Fetscherstraße 74, 01307 Dresden, Germany

{Robert.Haase,Nasreddin.Abolmaali}@OncoRay.de

² Department of Artificial Intelligence, Faculty of Computer Science and Mathematics, HTW Dresden, Friedrich-List-Platz 1, 01069 Dresden, Germany

³ Clinic and Policlinic for Radiation Oncology, University Hospital C.G. Carus, TU Dresden, Fetscherstraße 74, 01307 Dresden, Germany

⁴ Institute and Policlinic for Diagnostic Radiology, University Hospital C.G. Carus, TU Dresden, Fetscherstraße 74, 01307 Dresden, Germany

Abstract. For special applications in diagnostics for oncology the analysis of imaging data from Positron Emission Tomography (PET) is obfuscated by low contrast and high noise. To deal with this issue we propose a segmentation algorithm based on Ant Colony Optimization (ACO) and evolutionary selection of ants for self reproduction. The self reproduction approach is no standard for ACO, but appears to be crucial for volume segmentation. This investigation was focused on two different ways for reproduction control and their contribution to quantity and quality of segmentation results. One of the evaluated methods appears to be able to replace the explicit ant movement through transition rules by implicit movement through reproduction. Finally the combination of transition rules and self reproduction generates best reproducible segmentation results.

Keywords: Swarm Intelligence, Ant Colony Optimization, Collective Behavior, Self Reproduction, Positron Emission Tomography.

1 Introduction

In the field of medical imaging a vast variety of image segmentation approaches have been published. Image segmentation for volume analysis is part of the physicians' daily work in diagnostics and therapy planning. He is often supported by computer systems utilizing either simple deterministic algorithms or complex intelligent ones. In our work field – molecular imaging for radiation oncology – we are confronted to limited abilities of such algorithms. Distinguishing between target object and surrounding ‘healthy’ normal tissues in a three dimensional volume data set is not trivial on the one hand and a must for accurate therapy planning on the other hand. We developed a new algorithm for segmentation of low contrast Positron Emission Tomography (PET) data with the focus on accuracy, robustness and reproducibility.

The algorithm is based on the Ant Colony Optimization approach, first described in [1], in combination with self reproducing ants as proposed in [2]. The self reproductive behavior of ants is no standard in ACO related algorithms, but appears to be beneficially for generating reproducible volume delineations of objects of interest in noisy and poor contrasted PET volume data sets [3]. This investigation was about the contribution of the self reproductive behavior of the ants to quality and quantity of segmentation results. We compare a savage reproduction decision rule with a probabilistic rule to determine the better approach for automatic population size adaption.

2 Motivation and Related Work

The basic idea of using swarm intelligence for image segmentation in medical imaging is to support the physician in solving an unnatural task. Usually the human observer goes through the whole image stack to inspect it, before he draws outlines around the presumably positive regions. This can be very time consuming and in addition delineations might differ dramatically between human observers. A fully automatic and thus observer independent approach is desirable. In many fields of medical imaging simple methods like threshold or gradient based segmentation are sufficient to deliver an appropriate delineation. An example is Computed Tomography (CT), where the voxel values (in Hounsfield Units, HU) indirectly correlate with electron density in tissues. A threshold for differentiating bone and soft tissue can be estimated easily and is transferable from one measurement to another. In PET data analysis such a property cannot be assumed. The grey values express radioactivity inside a volume (in Becquerel per milliliter, Bq/ml) and depend on injected tracer volume, flow dynamics and biological properties. The resulting measures can be interpreted as surrogate parameters of physiological properties, like increased glucose metabolism or lack of oxygen supply (hypoxia). A determined threshold is not transferable from one patient to another [4]. Thus thresholds are usually determined as fraction of maximum activity [5] or relative to activity measured in blood samples [6] or reference volumes of interest [7, 8]. But the resulting delineations are only an approximation of the true positive volume. If additionally PET images are noisy and/or evince low contrast, simple threshold methods do not succeed. Also non-spherical target objects with inhomogeneous activity distribution raise the need for development of alternative algorithms [9].

Utilizing a large number of small entities for inspection of their local environment in the volume and combining their findings to a global view of the volume seems obvious. But the simplification of the classification problem from a global view to a local view results in more complex information retrieval: the data from all entities needs to be collected. The approach of indirect communication through a pheromone field – the established standard approach for ant colony based algorithms – delivers a suitable approach to deal with this issue. Thus the foundation of our proposed method is an ant colony based approach of voxel sized, virtual ants inspecting the PET volume, searching for regions with signal intensity above average and communicating indirectly through pheromone.

3 Ant Colony Optimization

Ant Colony Optimization (ACO) is an established metaheuristic for difficult optimization problems [10]. Entities are seeded in the parameter space of a complex problem and communicate indirectly through a pheromone field to build a distributed organization. The approach was first suggested in the early nineties by [1]. During the first years investigations on ACO concentrated on solving theoretical standard problems like the Traveling Salesman Problem. Investigations on image processing ant colonies resulted in the finding that such ants build cognitive maps of their environment using the virtual pheromone field [11]. In the last years ant colony based medical image segmentation algorithms were developed [12-14].

Another important finding was that ant colonies with variable population size are superior to colonies with fixed population size, when analyzing an object with unknown size [2]. In this approach the ant colonies objective is to fill a region of interest with ants until a criteria, in our case the objects boundary, is obtained. The population must be regulated, ideally autonomously, regarding to the size of the object. Therefore an ant system was proposed using a life cycle: ants are able to reproduce and exist for a limited number of iterations. Each ant is created with a survival probability of $P_S = 1$ and after each iteration it is decremented by a value of e.g. $\Delta_{PS} = 0.1$. In this case an ant has a maximum lifespan of $\Delta_{PS}^{-1} = 10$ iterations. Additionally selected ants are allowed to reproduce. A formula was proposed in [2] for determining the reproduction probability P_R in relation to the amount of pixels in the neighborhood n , the amount of ants m in the same area and the signal intensity $\Delta[i]$ in voxel i and maximum intensity Δ_{max} in the volume. The formula adapted to our three dimensional volume environment is as follows

$$P_{R,3D} = \left(1 - \frac{|m - \frac{n}{2}|}{\frac{n}{2}} \right) \cdot \frac{\Delta[i]}{\Delta_{max}} \quad (1)$$

Because ants are following transition rules to optimize a cost or fitness function, they should accumulate in valuable regions. The first multiplier of the formula simulates that single ants in an empty neighborhood cannot be at a valuable place and should not reproduce. Ants in an overfilled region should not reproduce too, because of the missing space for descendants. The optimal neighborhood occupancy for reproduction is 50%. The second multiplier is due to the grey values in the volume. Ants near the maximum signal intensity have a higher probability for self reproduction.

4 The Proposed Method

The proposed ACO simulation is build on the ACO metaheuristic with some modifications to simplify the ant movement behavior. The probability P_{ik} of an ant to move from location k to location i is expressed by weighting of voxel properties expressed by

$$P_{ik} = \frac{W(i)}{\sum_{j/k} W(j)} \quad (2)$$

This formula is a simplified version of the originally proposed formula from [1]. The expression $\Sigma_{j/k}$ describes the fact that all voxels j in the neighbourhood of the voxel k are summed up. The value of $W(i)$ depends on the task of the ant. In our simulation ants are divided into two castes: scout ants, responsible for volume exploration consider the activity in the voxel i , $W(i) = A(i)$, and search for regions with high signal intensity. Worker ants follow the pheromone gradient, $W(i) = \tau(i)$, to exploit the valuable regions. Both castes are allowed to reproduce, but they always produce worker ants as descendants. Scout ants are seeded at random locations to 1.4% of the voxels of the volume at the beginning of every iteration.

During the simulation a pheromone field is generated showing the target object with higher contrast than in the original data set. Of course this effect is limited to data sets evincing a contrast which is high enough. From phantom measurements we learned that the contrast-to-noise-ratio must be higher than 11. After the simulation the pheromone field and the worker ant distribution are analyzed using a histogram based thresholding approach. The main principle is shown in figure 1. The interested reader is referred to [3] for a detailed report on generating volume delineations from worker ant distribution and pheromone field.

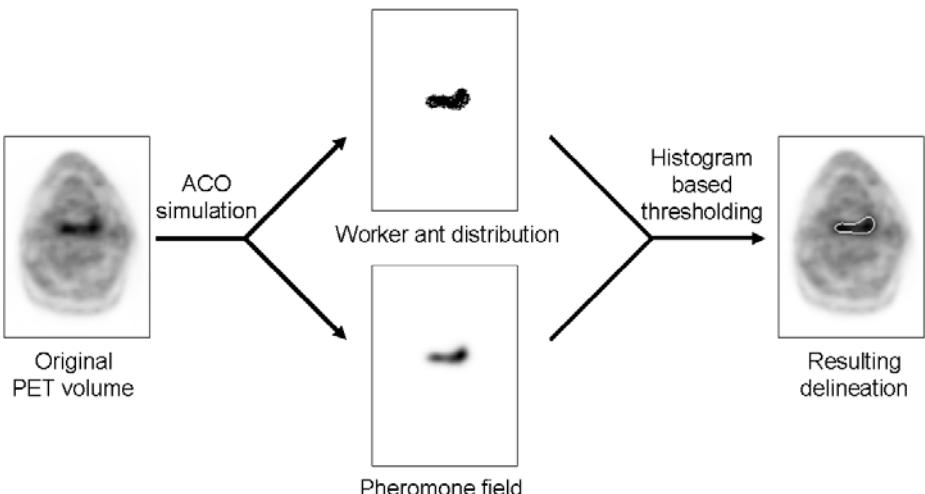


Fig. 1. The proposed volume segmentation strategy uses an ACO simulation for volume inspection and target selection. The results from this simulation are stored as worker ant distribution and pheromone field. These both volume data sets are further processed to generate object delineation.

5 Regulating Population Size

The aim of the ant simulation is delineating objects with signal intensity above average. This is the reason why our system adapts the population size using a savage

reproduction decision rule (SRDR) depending on two thresholds: The first, A_{\min} , is related to the activity intensities A of voxels in the volume. The other threshold τ_{\min} is related to normalized pheromone intensities. Only if an ant is located in a voxel exceeding both thresholds, it is allowed to reproduce ($P_R = 1$). In that way it is ensured that ants reproduce in regions where activity and surrounding conditions indicate the location to be valuable. The reproduction probability P_R of an ant is defined by the pheromone intensity τ_x , its maximum in the pheromone field τ_{\max} and the activity A_x at the ants location x . The equation for the P_R calculation using SRDR is

$$P_{R,SRDR} = \begin{cases} 1 & \text{if } A_x > A_{\min} \wedge \frac{\tau_x}{\tau_{\max}} > \tau_{\min}, \\ 0 & \text{else.} \end{cases} \quad (3)$$

Both thresholds are recalculated after every iteration, because at the beginning of the simulation the colony doesn't have any knowledge about the volume yet. The first threshold A_{\min} ensures that only ants in regions with activity A above average are allowed to reproduce. But the average activity \bar{A} is measured after each iteration, it is the mean activity of all voxels being occupied by ants after the last iteration. In that way the threshold can only increase from iteration to iteration until the mean activity \bar{A} stops increasing. A typical plot of the mean activity \bar{A} and A_{\min} is plotted in figure 2. The mean activity \bar{A} describes the satisfaction of the colony members. It is automatically maximized during the first iterations. Ants in promising regions self reproduce, while ants in poor regions are not allowed to do so. This results in a growing ant population in regions with signal intensity above average. Thus from iteration to iteration the measured average increases.

The other threshold τ_{\min} refers to pheromone intensity τ . It increases with time until the threshold A_{\min} reaches its maximum. After every iteration, when A_{\min} is increased, τ_{\min} is calculated using the iteration count i and a constant factor $P = 0.015$:

$$\tau_{\min} = P \cdot i \quad (4)$$

The intention is, when A_{\min} no longer increases, the colony is overpopulated. There are some ants occupying voxels with poor signal intensity and thus decreasing the mean activity of occupied voxels. After A_{\min} reached its maximum the colony population decreases due to a high τ_{\min} value. But the decreasing population of ants is still mostly located inside the target volumes and the resulting pheromone field still shows the target objects with higher pheromone contrast than contrast in the original PET data set.

To compare the proposed SRDR with a probabilistic equation as proposed in [2] in equation (1) we combined (1) and (3) to define a probabilistic reproduction decision rule (PRDR):

$$P_{R,PRDR} = \left(1 - \frac{|m - \frac{n}{2}|}{\frac{n}{2}} \right) \cdot \begin{cases} 1 & \text{if } A_x > A_{\min} \wedge \frac{\tau_x}{\tau_{\max}} > \tau_{\min}, \\ 0 & \text{else.} \end{cases} \quad (5)$$

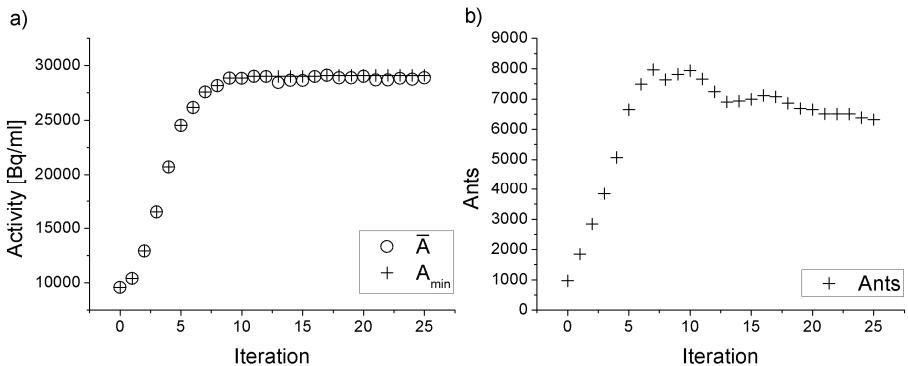


Fig. 2. Diagram a) shows the progress of mean activity \bar{A} of voxels occupied by ants measured during the iterations of the ACO simulation. The threshold A_{min} allowing ants to reproduce is always the maximum of \bar{A} in the past iterations. Corresponding diagram b) shows population size during the same simulation. The increasing threshold decelerates further self reproduction of the ants.

6 Experimental Setup

To examine the contribution of the self reproducing behavior to segmentation results the proposed methods SRDR and PRDR for calculating P_R were applied to clinical PET data sets of 23 patients from an ongoing prospective study on head and neck cancer. This study was approved by the institutional ethics committee and by the Federal Office for Radiation Protection of Germany. All patients were informed about the aims of the study and the further processing of the data for scientific analysis and gave their written informed consent. The data sets were acquired using the radiotracer [¹⁸F]-Fluoromisonidazole (FMISO), which is known to produce data sets with low contrast visualizing hypoxia.

All data sets were segmented for five times each using the same parameter configuration. Investigation focused on how the two different methods for determining the reproduction probability P_R perform. Secondly, it was evaluated how different configurations of the Δ_{PS} parameter influence the result. Six values for this parameter were tested (0.025, 0.05, 0.1, 0.2, 0.4 and 1) resulting in simulation of ants with a maximum live span of 40, 20, 10, 5, 2.5 and 1 iteration(s) respectively. Application of ants with a maximum live span of one iteration (resulting from $\Delta_{PS} = 1$) is a way to minimize the effect of the ant movement. To eliminate the contribution of ant movement completely and investigate the results, the SRDR simulation with $\Delta_{PS} = 1$ was repeated with ants which were not allowed to move.

To validate segmentation results we analyzed the measured positive volume and the mean Jaccard Index J of corresponding segmentation results. After each five iterations of the same data set and the same parameter configuration the segmentation results are expected as sets R_j ($j = 1..5$). The Jaccard Index of each pair is calculated by dividing intersection through union of two sets R :

$$J(R_i, R_j) = \frac{R_i \cap R_j}{R_i \cup R_j}, i, j = \{1, \dots, 5\}, i \neq j. \quad (6)$$

This method results in 10 J values and the resulting mean J is interpreted as indicator for reproducibility of the volume delineation.

7 Results

After execution of the 1495 segmentations (7x SRDR + 6x PRDR, 23 patients, 5 times repeated, $13 \cdot 23 \cdot 5 = 1495$) mean J and V measures for each data set were plotted against the Δ_{PS} parameter, as shown in figure 3. The diagrams also show the mean standard deviation resulting from the five equally configured segmentation processes.

The simulation results of the PRDR are obviously dependent on the Δ_{PS} parameter. The proposed SRDR method seems to be less dependent on it. On the one hand, the volume measures decrease, if the Δ_{PS} parameter is increased using the PRDR method. The influence is not that dramatic using SRDR approach. On the other hand the J values behave similar suggesting the segmentation results are not reproducible using PRDR method and higher Δ_{PS} values. The case $\Delta_{PS} = 1$ and using PRDR method resulted in a colony with very limited reproductive behavior: All ants are eliminated after their first iteration, because the maximum lifespan is 1. The resulting pheromone fields and ant distribution were not analyzable using our histogram analysis method and therefore this data point was not plotted in the diagram. Another 8 combinations of data sets and parameter configurations could not be analyzed for the same reason. Table 1 shows the list of experiments and the number of data sets which were segmented all five times successfully. Thus in figure 3 the data points $\Delta_{PS} = 0.33$ and $\Delta_{PS} = 0.4$ using PRDR should be regarded carefully.

The evaluation of the additional simulation with $\Delta_{PS} = 1$ and not moving ants is also shown in figure 3. The mean segmented volume is higher than segmented by the SRDR approach with moving ants and the J value is lower. This indicates a positive effect of the explicit movement approach to the reproducibility of the segmentation results.

Table 1. List of experiments with number of data sets which were segmented successfully all five times

| Δ_{PS} | 0.025 | 0.05 | 0.1 | 0.2 | 0.33 | 0.4 | 1 | 1 (with not moving ants) |
|---------------|-------|------|-----|-----|------|-----|----|--------------------------|
| SRDR | 23 | 23 | 23 | 23 | 23 | 23 | 23 | 23 |
| PRDR | 23 | 23 | 23 | 23 | 20 | 18 | 0 | - test not performed - |

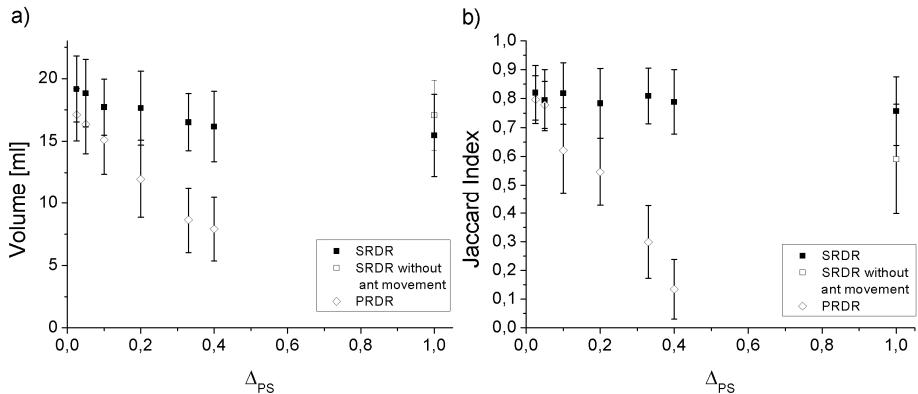


Fig. 3. a) Plots of the mean measured volume using the SRDR and PRDR approach. The Δ_{PS} parameter appears to influence the results of PRDR much more than in SRDR. b) The corresponding Jaccard Index indicates reproducibility differences applying a) SRDR and b) PRDR approach to the PET data sets. The results of the simulation using SRDR evince high reproducibility even if the maximum lifespan of the ants is 1. Using PRDR the reproducibility decreases when applying higher Δ_{PS} values.

To visualize the different colony reproduction behavior, an exemplary data set is shown in figure 4. While the SRDR method performs well almost independent from the Δ_{PS} configuration, the PRDR method fails to fill the target object with worker ants. Therefore the final delineation results in a smaller volume, as shown in figure 3. Figure 5 emphasizes the fact that PRDR method also allows to generate reproducible worker ant distributions but only if the Δ_{PS} parameter is high enough. This is the reason why the reproducibility, indicated by low J values in figure 3, is lower using the PRDR approach and higher Δ_{PS} values.

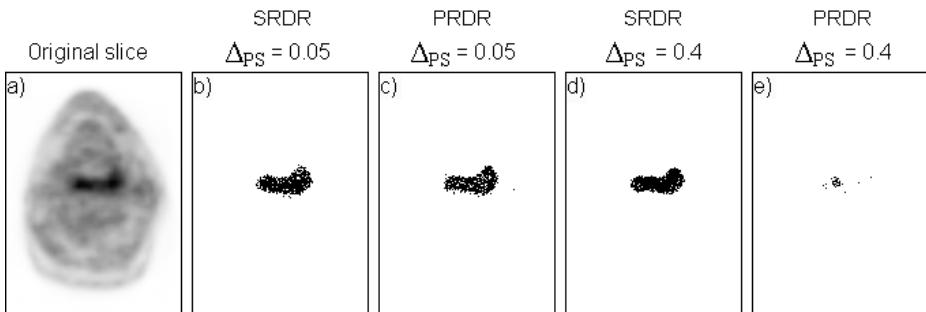


Fig. 4. a) Original FMISO PET axial slice of the jawbone region from a patient suffering from hypopharyngeal carcinoma. The increased signal intensity (black) at the center results from a hypoxic target volume, of which the delineation is in question. Images b) – e) show worker ant distributions after 15 iterations of the simulation. Black pixels correspond to an ant occupying a voxel in this slice. In b) and d) different Δ_{PS} parameter configurations using SRDR are comparable. Due to probabilistic movement, the ants are not distributed identical, but the worker ant distribution appears similar. In c) and e) the same parameter configurations were applied using PRDR approach, the resulting difference of the worker ant distribution is obvious.

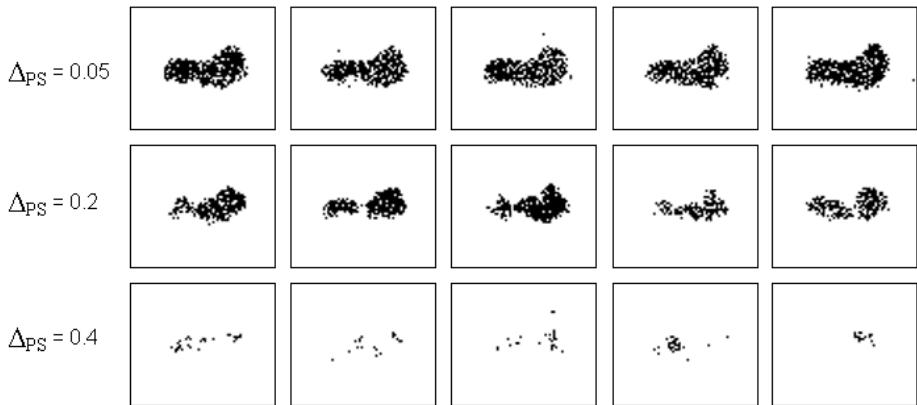


Fig. 5. Comparison of reproducibility of the worker ant distribution after 15 iterations using PRDR. All 5 simulations in a row were executed using the same configuration. The worker ant distribution is always different, but appears less reproducible at higher Δ_{PS} values. All shown simulations deliver worker ants moving inside the presumptive target volume, but the population size and location of the worker ants appears more stable in the top row.

8 Discussion

In our ACO simulation ants are able to locate and delineate regions of signal intensities above average in low contrast FMISO PET volume data. Two different methods for determination of the reproduction probability were presented and experimental simulations showed that both perform differently when applied to clinical PET data sets. The SRDR approach should be preferred due to its higher J values indicate better reproducibility.

The SRDR approach causes that all ants inside the target objects are allowed to reproduce, independent from occupancy of the neighborhood. This might be the reason why changing Δ_{PS} shows almost no remarkable effect on the volume delineations. The permanently eliminated ants are immediately replaced by descendants of surrounding ants inside the target objects. If $\Delta_{PS} = 1$, every ant is allowed to do one step, reproduce and after that it is eliminated. This raises the question as to whether the explicit movement of the ants can be suppressed, not affecting the results. If the ants are not allowed to move, but are still self reproducing at valuable locations, an implicit movement behavior might still appear, because descendants are seeded in the neighborhood of an ant. The complete elimination of the ant movement behavior resulted in segmentation results with lower J values, indicating the reproducibility of the algorithm is affected negatively. Thus both forms of movement, the explicit and the implicit, are needed for generating best reproducible segmentation results.

For some standard theoretical problems, like the Traveling Salesman Problem (TSP) it has been proven that the ACO algorithm converges in at least one optimal solution with an infinitesimal error after a sufficient number of iterations [15]. Several image segmentation approaches built on the ACO metaheuristic were published, but none has been proven in a similar way. Especially in the field of medical imaging a similar prove seems exorbitant effort expensive due to the complexity of this

distributed, parallelized and randomized algorithm. Additionally it doesn't seem promising due to the imponderables all tomography techniques comprise. The grey value of a voxel cannot be a guarantee for the healthiness of the volume it represents. At first scanning a volume and transforming the measurements to one grey value of a voxel always means summarizing different sub volumes. Tissue boundaries going through a voxel result in a grey value calculated as mean average of grey values from both tissues. This effect is called the Partial Volume Effect (PVE) and it is expected to influence all medical imaging techniques. A segmentation algorithm, which is based on grey value analysis, cannot deliver an accurate definition of a diseased volume, just because of PVE. The aim of segmentation techniques in medical imaging thus cannot be definition of healthiness or not, but it is able to support a physician in delineating a volume he expects to be of interest.

9 Conclusion

Both compared methods for selection of ants for reproduction, SRDR and PRDR, perform well for volume delineation, but only if the maximum life span of the ants is long enough ($\Delta_{PS} \leq 0.05$). In the case $\Delta_{PS} = 1$ only the savage reproduction decision rule is able to deliver ant distributions applicable for accurate volume delineation. This indicates that the contribution of the self reproducing paradigm to the final segmentation results is greater than the contribution of the ant movement. The reason must be seen in the fact that in our simulation the ant colony moves not only through explicit movement by transition rules, but also through reproduction. But disabling explicit ant movement affects the reproducibility of the algorithm negatively. Thus the combination of explicit movement and self reproduction delivers best reproducible volume delineations.

Due to the lack of a reference standard for FMISO PET imaging, the correctness of the resulting delineations cannot be proven. Comparison with other algorithms is not possible because to our knowledge, there are no other segmentation algorithms delineating FMISO PET data accurately to any degree. Thus further investigations, e.g. comparison with volume delineations generated by experienced physicians are currently in preparation for further evaluation of the proposed algorithm.

References

1. Colomi, A., Dorigo, M., Maniezzo, V.: Distributed Optimization by Ant Colonies. In: European Conference on Artificial Life (1992)
2. Fernandes, C., Ramos, V., Rosa, A.C.: Varying the Population Size of Artificial Foraging Swarms on Time Varying Landscapes. In: Duch, W., Kacprzyk, J., Oja, E., Zadrożny, S. (eds.) ICANN 2005. LNCS, vol. 3696, pp. 311–316. Springer, Heidelberg (2005)
3. Haase, R., et al.: A New Segmentation Algorithm for Low Contrast Positron Emission Tomography based on Ant Colony Optimization. In: 55th IWK - Crossing Borders within the ABC, Automation, Biomedical Engineering and Computer Science. TU Ilmenau, Ilmenau (2010)

4. Biehl, K.J., et al.: 18F-FDG PET definition of gross tumor volume for radiotherapy of non-small cell lung cancer: is a single standardized uptake value threshold approach appropriate? *J. Nucl. Med.* 47(11), 1808–1812 (2006)
5. Nehmeh, S.A., et al.: An iterative technique to segment PET lesions using a Monte Carlo based mathematical model. *Med. Phys.* 36(10), 4803–4809 (2009)
6. Swanson, K.R., et al.: Complementary but distinct roles for MRI and 18F-fluoromisonidazole PET in the assessment of human glioblastomas. *J. Nucl. Med.* 50(1), 36–44 (2009)
7. Eschmann, S.M., et al.: Prognostic impact of hypoxia imaging with 18F-misonidazole PET in non-small cell lung cancer and head and neck cancer before radiotherapy. *J. Nucl. Med.* 46(2), 253–260 (2005)
8. Abolmaali, N., et al.: Two or four hour [18F]FMISO-PET in HNSCC. When is the contrast best? *Nuklearmedizin*, 50(1) (2010)
9. Lee, J.A.: Segmentation of positron emission tomography images: some recommendations for target delineation in radiation oncology. *Radiother. Oncol.* 96(3), 302–307 (2010)
10. Dorigo, M.: Ant Algorithms Solve Difficult Optimization Problems. In: Kelemen, J., Sosík, P. (eds.) ECAL 2001. LNCS (LNAI), vol. 2159, pp. 11–22. Springer, Heidelberg (2001)
11. Chialvo, D., Millonas, M.M.: How Swarms Build Cognitive Maps. In: Steels, L. (ed.) The Biology and Technology of Intelligent Autonomous Agents, pp. 439–450 (1995)
12. Huang, P., Cao, H., Luo, S.: An artificial ant colonies approach to medical image segmentation. *Comput. Methods Programs Biomed.* 92(3), 267–273 (2008)
13. Cerello, P., et al.: 3-D object segmentation using ant colonies. *Pattern Recognition* 43(4), 1476–1490 (2010)
14. Myung-Eun, L., et al.: Segmentation of Brain MR Images Using an Ant Colony Optimization Algorithm. In: Ninth IEEE International Conference on Bioinformatics and BioEngineering, BIBE (2009)
15. Stützle, T., Dorigo, M.: A short convergence proof for a class of ant colony optimization algorithms. *IEEE Trans. Evolutionary Computation*, 358–365 (2002)

Efficient Sequential Clamping for Lifted Message Passing

Fabian Hadiji, Babak Ahmadi, and Kristian Kersting

Knowledge Discovery Department, Fraunhofer IAIS, 53754 Sankt Augustin, Germany
`{firstname.lastname}@iais.fraunhofer.de`

Abstract. Lifted message passing approaches can be extremely fast at computing approximate marginal probability distributions over single variables and neighboring ones in the underlying graphical model. They do, however, not prescribe a way to solve more complex inference tasks such as computing joint marginals for k -tuples of distant random variables or satisfying assignments of CNFs. A popular solution in these cases is the idea of turning the complex inference task into a sequence of simpler ones by selecting and clamping variables one at a time and running lifted message passing again after each selection. This naive solution, however, recomputes the lifted network in each step from scratch, therefore often canceling the benefits of lifted inference. We show how to avoid this by efficiently computing the lifted network for each conditioning directly from the one already known for the single node marginals. Our experiments show that significant efficiency gains are possible for lifted message passing guided decimation for SAT and sampling.

Keywords: Relational Probabilistic Models, Relational Learning, Probabilistic Inference, Satisfiability.

1 Introduction

Recently, there has been much interest in methods for performing lifted probabilistic inference, handling whole sets of indistinguishable objects together, see e.g. [9][15] and references in there. Most of these lifted inference approaches are extremely complex, so far do not easily scale to realistic domains and hence have only been applied to rather small artificial problems. An exception are lifted versions of belief propagation (BP) [16][6]. They group together random variables that have identical computation trees and now run a modified BP on the resulting lifted (compressed) network. Being instances of BP, they can be extremely fast at computing approximate marginal probability distributions over single variable nodes and neighboring ones in the underlying graphical model. Still, lifted message passing approaches leave space for improvement. For instance, in BP-guided decimation for satisfiability and sampling [10][8], which are common methods for two important AI tasks, one is essentially interested in probabilities of the same network but with changing evidence. Both tasks are crucial to AI in general and have important applications. For example, sampling is often used in

parameter learning and the idea of "reduction to SAT" is a powerful paradigm for solving problems in different areas. A popular solution in these cases is the idea of turning the complex inference task into a sequence of simpler ones by selecting and clamping variables one at a time and running lifted BP again after each selection. However, the naive solution recomputes the lifted network in each step from scratch, therefore often canceling the benefits of lifted inference.

This paper makes a number of important and novel contributions to both the WP and lifted BP literature. We present *Shortest-Paths-Sequence Lifting* for BP (SPS-LBP), a scalable lifted inference algorithm for approximate solutions of complex inference tasks. SPS-LBP avoids the lifting from scratch for each sub-task by efficiently computing the corresponding lifted network directly from the one already known for the single node marginals. We demonstrate the powerful application of these techniques to two novel lifted inference tasks: lifted message passing guided sampling and SAT solving. The experimental results demonstrate that significant efficiency gains are obtainable compared to ground inference and naive lifting in each iteration.

Indeed, there has been some prior work for related problems. Delcher *et al.* [3] propose a data structure that allows efficient queries when new evidence is incorporated in singly connected Bayesian networks and Acar *et al.* [1] present an algorithm to adapt the model to structural changes using an extension of Rake-and-Compress Trees. The only lifted inference approach we are aware of is the work by Nath and Domingos [11]. They essentially memorize the intermediate results of previous liftings. For new evidence they make a warm start from the first valid intermediate lifting. So far their approach has not been used for lifted sampling and lifted satisfiability. Additionally, the algorithm is only defined for Markov Logic Networks (MLNs) whereas our approach applies to any factor graph. Furthermore, the approach presented in the present paper gives a clear characterization of the core information required for sequential clamping for lifted message passing: the shortest-paths connecting the variables in the network. We proceed as follows. We start off by briefly reviewing LBP and show how it can be carried over to WP. Then, we introduce SPS-LBP. Before concluding, we present our experimental results.

2 Lifted Message Passing

Let $\mathbf{X} = (X_1, \dots, X_n)$ be a set of n discrete-valued random variables each having d states, and let x_i represent the possible realizations of random variable X_i . Graphical models compactly represent a joint distribution over \mathbf{X} as a product of factors [12], i.e., $P(\mathbf{X} = \mathbf{x}) = Z^{-1} \prod_k f_k(\mathbf{x}_k)$. Each factor f_k is a non-negative function of a subset of the variables \mathbf{x}_k , and Z is a normalization constant. Each graphical model can be represented as a factor graph, a bipartite graph that expresses the factorization structure of the joint distribution. It has a variable node (denoted as a circle) for each variable X_i , a factor node (denoted as a square) for each f_k , with an edge connecting variable node i to factor node k if and only if X_i is an argument of f_k . An important ($\#P$ -complete) inference

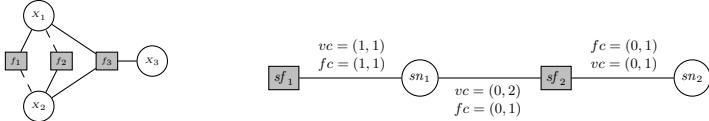


Fig. 1. (Left) $(X_1 \vee \neg X_2) \wedge (\neg X_1 \vee X_2) \wedge (X_1 \vee X_2 \vee X_3)$ represented as a factor graph. Circles denote variables, squares denote factors. A dashed line indicates that the variable appears negated in a clause **(Right)** Lifted factor graph after running CP.

task is to compute the conditional probability of variables given the values of some others, the evidence, by summing out the remaining variables. BP is an efficient way to solve this problem that is exact when the factor graph is a tree, but only approximate when the factor graph has cycles. Although loopy BP has no guarantees of convergence or of giving the correct result, in practice it often does, and can be much more efficient than other methods. BP can be elegantly described in terms of sending messages within a factor graph. The message from a variable X to a factor f is $\mu_{X \rightarrow f}(x) = \prod_{h \in \text{nb}(X) \setminus \{f\}} \mu_{h \rightarrow X}(x)$ where $\text{nb}(X)$ is the set of factors X appears in. The message from a factor to a variable is $\mu_{f \rightarrow X}(x) = \sum_{\neg\{X\}} \left(f(\mathbf{x}) \prod_{Y \in \text{nb}(f) \setminus \{X\}} \mu_{Y \rightarrow f}(y) \right)$ where $\text{nb}(f)$ are the arguments of f , and the sum is over all of these except X , denoted as $\neg\{X\}$. The unnormalized belief of each variable X_i can be computed from the equation $b_i(x_i) = \prod_{f \in \text{nb}(X_i)} \mu_{f \rightarrow X_i}(x_i)$. Evidence is incorporated by setting $f(\mathbf{x}) = 0$ for states \mathbf{x} that are incompatible with it.

Although already quite efficient, many graphical models produce factor graphs with a lot of symmetries not reflected in the structure. Lifted BP (LBP) can make use of this fact by essentially performing two steps: Given a factor graph G , it first computes a compressed factor graph \mathfrak{G} and then runs a modified BP on \mathfrak{G} . We will now briefly review the first step of LBP. For more details we refer to [6].

The lifting step can be viewed as a color-passing (CP) approach. This view abstracts from the type of messages sent and, hence, highlights one of the main insights underlying the present paper: *CP can be used to lift other message-passing approaches besides BP as well*. Since a large part of our experiments focuses on Boolean satisfiability problems, we will here explain some of the specifics when lifting Boolean formulas. We assume that a Boolean formula is represented in Conjunctive Normal Form (CNF). Every CNF can be represented as a factor graph, see e.g. [7]. More importantly, as we will show now, CP can actually be simplified in the case of CNFs. A CNF is a conjunction of disjunctions of Boolean literals. A literal is either a negated or unnegated propositional variable. Specifically, a CNF consists of n variables with $x_i \in \{0, 1\}$ and m clauses constraining the variables. Every clause is represented by a factor f_k . A solution to a CNF is an assignment to all variables satisfying all constraints f_k . As an example, consider the following CNF: $(X_1 \vee \neg X_2) \wedge (\neg X_1 \vee X_2) \wedge (X_1 \vee X_2 \vee X_3)$. The factor graph representing this CNF is shown in Fig. 1 (left). We use a dashed edge between a variable and a clause whenever the variable appears negated in a clause, otherwise a full line.

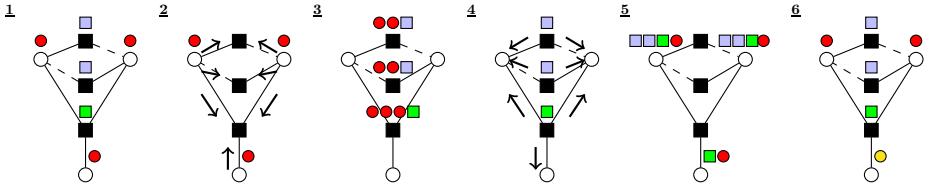


Fig. 2. From left to right, the steps of running CP on the factor graph in Fig. I (left) (assuming no evidence). The colored small circles and squares denote the groups and signatures produced running CP. (Best viewed in color.)

Let G be an arbitrary factor graph with variable and factor nodes. Initially, all variable nodes fall into $d + 1$ groups (one or more of these may be empty) — known states s_1, \dots, s_d , and *unknown* — represented by colors. In the case of CNFs, $d = 2$. All factor nodes with the same associated potentials also fall into one group represented by a shade. For CNFs, two clauses fall into the same group if both have the same number of positive and negative literals. For the factor graph in Fig. I the CP steps are depicted in Fig. 2. As shown on the left-hand side, assuming no evidence, all variable nodes are unknown, here: red. Now, each variable node sends a message to its neighboring factor nodes (step 2). A factor node sorts the incoming colors into a vector according to the order the variables appear in its arguments. The last entry of the vector is the factor node's own color, represented as light blue, respectively green, squares in Fig. 2 (step 3). Based on these signatures, the colors of the factors are newly determined and sent back to the neighboring variable nodes (step 4). The variable nodes stack the incoming signatures and, hence, form unique signatures of their one-step message history (step 5). Variable nodes with the same stacked signatures are grouped together. To indicate this, we assign a new color to each group (step 6). In our example, only variable node X_3 changes its color from red to yellow. This process is iterated until no new colors are created anymore. The final lifted graph \mathfrak{G} is constructed by grouping nodes (factors) with the same color into *supernodes* (*superfactors*). Supernodes (resp. superfactors) are sets of nodes (resp. factors) that send and receive the same messages at each step of carrying out message-passing on G . In our case, variable nodes X_1, X_2 and factor nodes f_1, f_2 are grouped together into supernode sn_1 and superfactor sf_1 . (Fig. I (right)). On this lifted network, LBP runs an efficient modified message passing. We refer to [I6.6] for details. However, in the case of CNFs, we can employ a more efficient simplified color signature coding scheme. The factor nodes do not have to sort incoming colors according to the positions of the variables, instead only the sign of a variable matters, i.e. only two positions exist.

3 Lifted Decimation

Many complex inference tasks, such as finding a satisfying assignment to a Boolean formula or computing joint marginals of random variables, can be cast into a sequence of simpler ones by selecting and clamping variables one at a time

Algorithm 1. Lifted Decimation using SPS

```

input: A factor graph  $fg$ , list of query vars  $L$ 
output: List of clamped variables and values

1 cfg  $\leftarrow \text{compress}(fg);$ 
2 distances  $\leftarrow \text{calcDistances}(fg, cfg);$ 
3 varList  $= \emptyset;$ 
4 while  $L \neq \emptyset$  do
5   marginals  $\leftarrow \text{runInference}(cfg);$ 
6   var, val  $= \text{pickVarToClamp}(\text{marginals});$ 
7   clampVariable( $cfg$ , var, val);
8   adaptLifiting( $cfg$ , dist);
9   varList  $+ = (\text{var}, \text{val});$ 
10  remove var from  $L;$ 
11 return varList

```

and running lifted inference after each selection. This is sometimes called decimation and is essentially summarized in Alg. 1. SAT problems can be solved using a decimation procedure based on BP (Montanari *et al.* 2007). Here, decimation is the process of iteratively assigning truth values to variables of formula F . This results in a factor graph which has these variables clamped to a specific truth value. Now, we repeatedly decimate the formula in this manner, until all variables have been clamped. In fact, when we are interested in solving CNFs, we make use of a second message passing algorithm in every iteration because some of the variables may be implied directly by unit clauses. Unit clauses are clauses consisting of a single literal only, i.e. factors with an edge to exactly one variable X . A unit clause essentially fixes the truth value of X . The process of fixing all variables appearing in unit clauses and simplifying the CNF correspondingly is called *Unit Propagation* (UP). Cast into the message passing framework, it is known under the name of *Warning Propagation* (WP) [2]. Intuitively, a message $\mu_{f \rightarrow X} = 1$ sent from a factor to a variable says: "Warning! The variable X must take on the value satisfying the clause represented by the factor f ."

In the lifted case, however, we need to extend the WP equations by counts, to ensure that we simulate the ground messages. First, there is the *factor count* fc . It represents the number of equal ground factors that send messages to the variable at a given position. Because there are only two positions — a variable may occur at any position in *negated* and *unnegated* form — we store two values: counts for the negated position and for the unnegated position. Second, there is the *variable count* vc . It corresponds to the number of ground variables that send messages to a factor at each position. Reconsider Fig. 1 (right). Here, the factor count associated with the edge between sn_1 and sf_1 is $(1, 1)$ because sn_1 represents ground variables that appeared positive and negative in ground factors represented by sf_1 . In the ground network, the clause f_3 is connected to three positive variables, but two of them are represented by a single supernode in the lifted network. Hence, we have variable count $vc = (0, 2)$ for

¹ Presented already using SPS that is devised in the next section.

the edge between sn_1 and sf_2 . The lifted equations are as follows:³: $\mu_{f \rightarrow X, p} = \prod_{Y \in \text{nb}(f)} \prod_{p' \in P(f, Y)} \theta(n_{Y \rightarrow f, p'} s_{p'})^{\text{vc}(f, Y, p') - \delta_{XY} \delta_{pp'}}$ with $\theta(x) = 0$ if $x \leq 0$ and 1 otherwise, $s_{p'} = 1$ if $p' = 0$ and $s_{p'} = -1$ if $p' = 1$, $\delta_{ij} = 1$ if $i = j$ and 0 otherwise. Here, $P(f, Y)$ only runs over positions with a variable count greater than zero. Reconsidering the factor graph in Fig. □ (right), P contains both positions for the edge $sn_1 - sf_1$, while for $sn_2 - sf_2$ it only contains the unnegated position. The Kronecker deltas reduce counts when a message is sent to the node itself that means it prevents double counting of messages. Due to space limitations we are not stating the proof that this is correct. Instead we illustrate using an example. Applying the formulas on the ground network shown in Fig. □ (left) shows that the lifted formula indeed simulates WP on the ground network. In this case, there is exactly one position with a variable count greater than zero for each neighbor Y of factor f . In other words, $P(f, Y)$ is a singleton. In turn, the second product can be dropped, and both formulas coincide. Similarly, the lifted message from a supervariable to a superfactor is: $\mu_{X \rightarrow f, p} = (\sum_{\substack{h \in \text{nb}(X), \\ \text{fc}(h, X, 1) > 0}} \mu_{h \rightarrow X}(\text{fc}(h, X, 1) - \delta_{fh} \delta_{p1})) - (\sum_{\substack{h \in \text{nb}(X), \\ \text{fc}(h, X, 0) > 0}} \mu_{h \rightarrow X}(\text{fc}(h, X, 0) - \delta_{fh} \delta_{p0}))$. We can prove that lifted warning propagation³ (LWP) gives the same results as WP applied to the ground network similar to the proof for LPB [16].

In our sequential setting, first (L)WP is run on the factor graph to clamp directly implied variables. Based on these implications the lifting is updated and (L)BP is used to fix the next variable. Additionally, we use (L)WP to detect possible contradictions. When (L)WP finds a contradiction, the algorithm stops and does not return a satisfying configuration. Indeed, lifted inference can already speed up the decimation procedure. The naive solution, however, recomputes the lifted network in each step from scratch, therefore often canceling the benefits of lifted inference. As we will show, we can do better.

4 Lifted Sequential Inference

When we turn a complex inference task into a sequence of simpler tasks, we are repeatedly answering slightly modified queries on the same graph. Because lifted BP/WP generally lacks the opportunity of adaptively changing the lifted graph and using the updated lifted graph for efficient inference, it is doomed to lift the original model in each of the k iterations again from scratch. Each CP run scales $\mathcal{O}(n \cdot m)$ where n is the number of nodes and m is the length of the longest path without a loop. Hence, we essentially spend $\mathcal{O}(k \cdot n \cdot m)$ time just on lifting. Consider now BP-guided sampling (although the same argument applies to BP-guided decimation). When we want to sample from the joint distribution over k

² We define $0^0 = 1$ in cases where $\theta(x) = 0$ and $\text{vc}(a, j, p') - \delta_{ij, pp'} = 0$.

³ For the sake of simplicity, we focussed on the evidence-free updates. In the experiments, we used similar looking updates involving clamped variables, i.e., evidence. Clamping variables has effects on the WP equations, since variables can possibly not satisfy clauses anymore due to their clamped value. Additionally, we can skip messages for variables that have already been set.

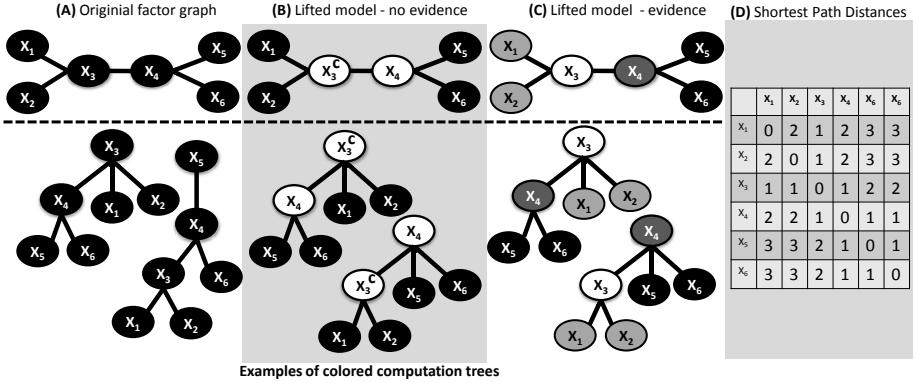


Fig. 3. (A): Original factor graph. (B): Prior lifted network, i.e., lifted factor graph with no evidence. (C): Lifted factor graph when X_3 is set to some evidence. Factor graphs are shown (**top**) with corresponding colored computation trees (**bottom**). For simplicity, we assume identical factors (omitted here). Ovals denote variables/nodes. The shades in (B) and (C) encode the supernodes. (D): Shortest-path distances of the nodes. The i -th row will be denoted d_i .

variables, this can be reduced to a sequence of one-variable samples conditioned on a subset of the other variables [8]. Thus, to get a sample for $X = X_1, \dots, X_k$, we first compute $P(X_1)$, then $P(X_2|X_1), \dots, P(X_k|X_1, \dots, X_{k-1})$ (Alg. II).

Assume we want to sample from the joint distribution $P(X_1, X_2, X_3)$, given the network in Fig. 3(A). We first compute $P(X_3)$ from the prior lifted network, i.e., the lifted network when no evidence has been set (B) and sample a state x_3 . Now, we want to compute $P(X|x_3)$ as shown in (C). To do so, it is useful to describe BP in terms of its *computation tree* (CT), see e.g. [5]. The CT is the unrolling of the graph structure where each level i corresponds to the i -th iteration of message passing. Similarly we can view CP as a *colored computation tree* (CCT). More precisely, one considers for every node X the computation tree rooted in X but now each node in the tree is colored according to the nodes' initial colors, cf. Fig. 3(bottom). Each CCT encodes the root nodes' local communication patterns that show all the colored paths along which node X communicates in the network. Consequently, CP groups nodes with respect to their CCTs: nodes having the same set of rooted paths of colors (node and factor names neglected) are clustered together. For instance, Fig. 3(A) shows the CCTs for X_3 and X_5 . Because their set of paths are different, X_3 and X_5 are clustered into different supernodes as shown in Fig. 3(B). Now, when we clamp the node X_3 to a value x_3 we change the communication pattern of every node having a path to X_3 . Specifically, we change X_3 's (and only X_3 's) color in all CCTs X_3 is involved, as indicated by the "c" in Fig. 3(B). This effects nodes X_1 and X_2 differently than X_4 respectively X_5 and X_6 for two reasons: (1) they have different communication patterns as they belong to different supernodes in the prior network; more importantly, (2) they have different paths connecting them to X_3 in their CCTs. The shortest path is the shortest sequence of factor colors

connecting two nodes. Since we are not interested in the paths but whether the paths are identical or not, these sets might as well be represented as colors. Note that in Fig. 3 we assume identical factors for simplicity. Thus in this case path colors reduce to distances. In the general case, however, we compare the paths, i.e. the sequence of factor colors.

The prior lifted network can be encoded as the vector $l = (0, 0, 1, 1, 0, 0)$ of node colors. Thus, to get the lifted network for $P(X|x_3)$ as shown in Fig. 3(C), we only have to consider the vector d_3 of shortest-paths distances to X_3 , cf. Fig. 3(D), and refine the initial supernodes correspondingly. This is done by (1) $l \oplus d_3$, the element-wise concatenation of two vectors, and (2) viewing each resulting number as a new color. $(0, 0, 1, 1, 0, 0) \oplus (1, 1, 0, 1, 2, 2) =_{(1)} (01, 01, 10, 11, 02, 02) =_{(2)} (3, 3, 4, 5, 6, 6)$. Thus, we can directly update the prior lifted network in linear time without taking the detour through running CP on the ground network. Now, we sample a state $X_4 = x_4$ and compute the lifted network for $P(X|x_4, x_3)$, to draw a sample for $P(X_1|x_4, x_3)$. Essentially, we proceed as before: compute $l \oplus (d_3 \oplus d_4)$. However, the resulting network might be suboptimal. It assumes $x_3 \neq x_4$ and, hence, X_3 and X_4 cannot be in the same supernode. For $x_4 = x_3$, they could be placed in the same supernode, if they are in the same supernode in the prior network. This can be checked by $d_3 \odot d_4$, the element-wise sort of two vectors. In our case, this yields $l \oplus (d_3 \odot d_4) = l \oplus l = l$: the prior lifted network. In general, we compute $l \oplus (\bigoplus_s (\bigoplus_v d_{s,v}))$ where $d_{s,v} = \bigoplus_{i \in s: x_i=v} d_i$, s and v are the supernodes and the truth value respectively. For an arbitrary network, however, the shortest paths might be identical although the nodes have to be split, i.e. they differ in a longer path, or in other words, the shortest paths of other nodes to the evidence node are different. Consequently we iteratively apply the shortest paths lifting. Let SN_S denote the supernodes given the set S as evidence. By applying the shortest path procedure we compute $SN_{\{X_1\}}$ from SN_\emptyset . This step might cause initial supernodes to be split into newly formed supernodes. To incorporate these changes in the network structure the shortest paths lifting procedure has to be iteratively applied. Thus in the next step we compute $SN_{\{X_1\} \cup \Gamma_{X_1}}$ from $SN_{\{X_1\}}$, where Γ_{X_1} denotes the changed supernodes of the previous step. This procedure is iteratively applied until no new supernodes are created. This essentially sketches the proof of the following theorem.

Theorem 1. *If the shortest-path colors among all nodes and the prior lifted network are given, computing the lifted network for $P(X|X_i, \dots, X_1)$, $i > 0$, takes $\mathcal{O}(i \cdot n \cdot s)$, where n is the number of nodes, s is the number of supernodes. Running MBP produces the same results as running BP.*

5 Experiments

Our intention here is to investigate whether lifting improves sequential inference approaches (Q1) and if SPS can be even more beneficial (Q2). Therefore, we run experiments on two AI tasks in which sequential clamping is essential, namely BP guided decimation for satisfiability problems and sampling in MLNs [13]. Both tasks essentially follow the decimation strategy shown in Alg. II.

Table 1. Total messages sent (millions) in SAT experiments and number of average flips needed by Walksat

| CNF Name | Iters | Ground | Naive | SPS | Walksat |
|-----------------|-------|--------|-------|-------|-----------|
| ls8-normalized | 26 | 3.17 | 1.12 | 0.95 | 540 |
| ls9-normalized | 13 | 5.47 | 1.65 | 1.47 | 1,139 |
| ls10-normalized | 14 | 10.27 | 1.84 | 1.59 | 1,994 |
| ls11-normalized | 26 | 38.82 | 11.51 | 10.64 | 4,500 |
| ls12-normalized | 35 | 60.83 | 13.15 | 11.57 | 10,351 |
| ls13-normalized | 21 | 55.39 | 9.99 | 8.21 | 30,061 |
| ls14-normalized | 22 | 83.30 | 10.22 | 8.30 | 104,326 |
| 2bitmax_6 | 55 | 2.35 | 1.25 | 1.05 | 379 |
| 5_100_sd_schur | 53 | 111.19 | 75.98 | 64.91 | 1,573,208 |
| wff.3.100.150 | 54 | 0.19 | 0.26 | 0.22 | 17 |
| wff.4.100.500 | 78 | 1.73 | 2.04 | 1.89 | 33 |
| wff.3.150.525 | 126 | 6.36 | 6.76 | 6.56 | 284 |

Lifted Satisfiability: We compared the performance of lifted message passing approaches with the corresponding ground versions on a CNF benchmark from [4]. We use decimation as described above to measure the efficiency of the algorithms. To assess performance, we report the number of messages sent. For the typical message sizes, e.g., for binary random variables with low degree, computing color messages is essentially as expensive as computing the actual messages. Therefore, we report both color and (modified) BP messages, treating individual message updates as atomic unit time operations. We used the “flooding” message protocol for (L)BP and (L)WP where messages are passed from each variable to all corresponding factors and back at each step. The convergence threshold was 10^{-8} for (L)BP, all messages were initialized to one (zero for (L)WP). As mentioned above, it is usually necessary to iteratively apply the SPS-lifting. The number of required iterations, however, can be high if long paths occur in the network. Therefore, we use the SPS-lifting only once but then continue with standard CP to determine the new lifting. This can still save several passes of color passing.

We evaluated (lifted) WP+BP decimation on different CNFs, ranging from problems with about 450 up to 78,510 edges. The CNFs contain structured problems as well as random instances. The statistics of the runs are shown in Tab. II. As one can see, naive lifting already yields significant improvement. When applying the SPS-lifting we can do even better by saving additional messages in the compression phases. The savings in messages are visible in running times as well. Looking at the experiment in Fig. 4 and comparing ground decimation with its lifted counterpart, we only send 33% of the total ground messages. When using the SPS-lifting, we can save up to an additional 10% messages in the compression. In the decimation we always clamp the most magnetized (largest difference between negated and unnegated marginals) variable. We also applied the lifted message passing algorithms to random CNFs (last three rows in Tab. II). As expected, no lifting was possible because random instances do usually not contain symmetries. In our experiments we were able to find satisfying solutions for all problems.

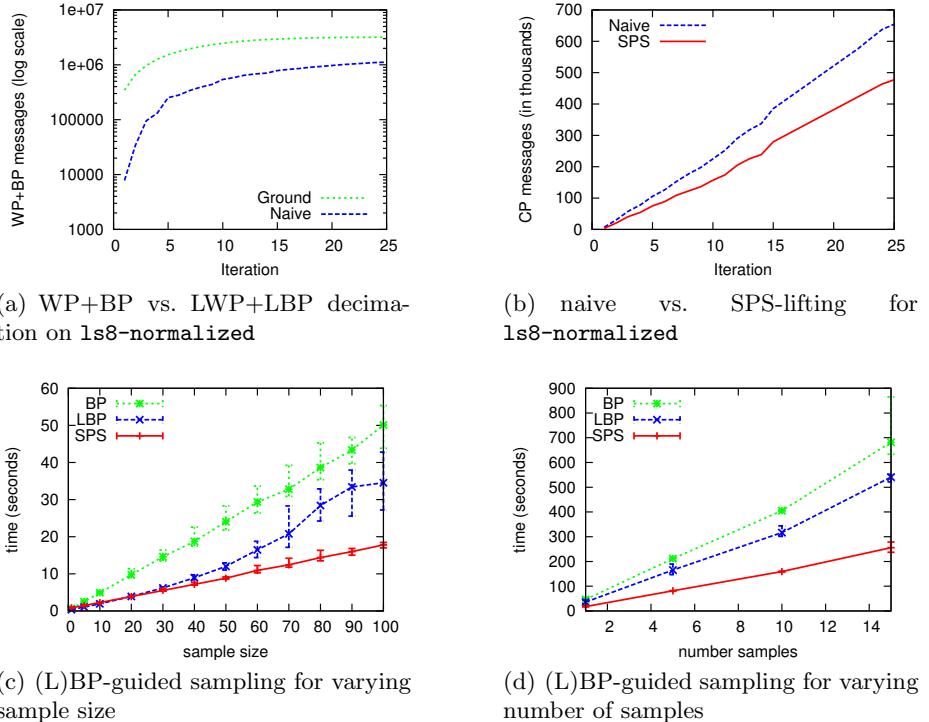


Fig. 4. Experimental results for complex sequential inference tasks

Although we are not aiming at presenting a state-of-the-art SAT solver, we solved all problems using Walksat [14] as well and we report results in Tab. I measured in variable flips. Even though Walksat requires fewer flips than we send messages, one can see that our lifted decimation strategy still scales well. In Fig. 5 (right) we have compared the computational effort on increasing problem sizes for Walksat and our lifted decimation. The results indicate that our approach can handle large problem instances without employing complex heuristics and code optimization but exploiting symmetries in the problems.

Lifted Sampling: We investigated BP, LBP and SPS-LBP for sampling a joint configuration over a set of variables sequentially, i.e. to a sequence of one-variable samples conditioned on a subset. Thus, to get a sample for $X = X_1, \dots, X_k$, we first compute $P(X_1)$, then $P(X_2|X_1), \dots, P(X_k|X_1, \dots, X_{k-1})$ as shown in Alg. II. For the "Friends-and-Smokers" dynamic MLN with 10 people over 10 time steps [6], Fig. 4 summarizes the results.

In our first experiment, we randomly chose 1, 5, 10, 20, 30, ..., 100 "cancer" nodes over all time steps, and sampled from the joint distribution. As one can see, LBP already provides significant improvement compared to BP, however, as the sample size increases, the speed-up is lower. The more evidence we have in the network, the less lifting is possible. SPS-LBP has the additional gain in runtime as we do not need to perform the lifting in each step from scratch.

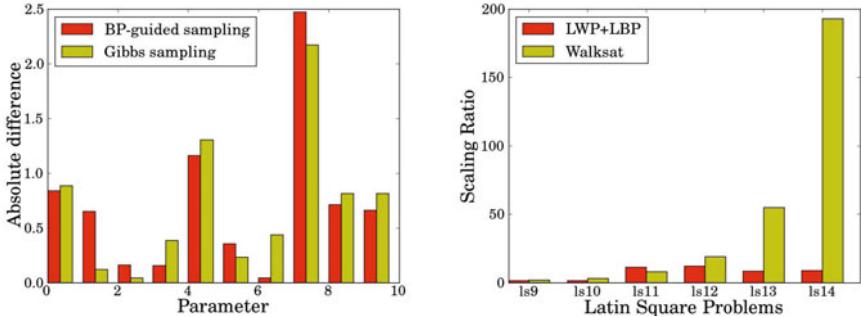


Fig. 5. left: Absolute difference of the learned parameter from the parameters of the original distribution the samples were drawn. **right:** Growth of computational costs on increasing problem sizes measured relative to the smallest problem.

In our second experiment we fixed the sample size to 100, i.e. we sampled from the joint distribution of all $cancer(X, t)$ for all persons X in the domain and all time steps t . We drew 1, 5, 10 and 15 samples and the timings are averaged over 5 runs. Here, we see that LBP is only slightly advantageous compared to BP, as the sample size is 100, especially in the later iterations we have lots of evidence and long chains to propagate the evidence. Repeatedly running CP almost cancels the benefits. SPS-LBP on the other hand, shows significant speed-ups.

To evaluate the quality of the samples, we drew 100 samples of the joint distribution of all variables using the BP-guided approach and Gibbs sampling respectively. We learned the parameters of the model maximizing the conditional marginal log-likelihood (CMLL) using scaled conjugate gradient (SCG). Fig. 5 (left) shows the absolute difference of the learnt weights from the model the data-cases were drawn. As one can see parameter learning with BP-guided samples performs as good as with samples drawn by Gibbs sampling. The root-mean-square error (RMSE) for the bp parameters was 0.31 and for Gibbs parameters 0.3.

6 Conclusion

In this paper, we proposed the first decimation framework guided by lifted message-passing algorithms. To avoid the lifting from scratch in each iteration of a naive realization, we employed an efficient sequential clamping approach and gave a novel characterization of the main information required in terms of shorted-paths in a given network. The experimental results on two novel tasks for lifted inference, namely Boolean satisfiability and sampling from Markov logic networks validate the correctness of the proposed lifted decimation framework and demonstrate that instantiations can actually be faster than just using lifting.

Indeed, much remains to be done. Since lifting SAT solvers itself and the exploitation of efficient sequential lifting for it is a major advance, the most interesting avenue for future work is the tight integration of lifted SAT and lifted probabilistic inference. In many real-world applications, the problem formulation

does not fall neatly into one of them. The problem may have a component that can be well-modeled as a SAT problem. Our work suggests to partition a problem into corresponding subnetworks, run the corresponding type of lifted message passing algorithm on each subnetwork, and to combine the information from the different sub-networks. Another interesting avenue is to explore lifted inference for (stochastic) planning.

Acknowledgements. This work was supported by the Fraunhofer ATTRACT fellowship STREAM and by the European Commission under contract number FP7-248258-First-MM.

References

1. Acar, U., Ihler, A., Mettu, R., Sumer, O.: Adaptive inference on general graphical models. In: Proc. of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI 2008). AUAI Press, Corvallis (2008)
2. Braunstein, A., Mézard, M., Zecchina, R.: Survey propagation: An algorithm for satisfiability. *Random Structures and Algorithms* 27(2), 201–226 (2005)
3. Delcher, A.L., Grove, A.J., Kasif, S., Pearl, J.: Logarithmic-time updates and queries in probabilistic networks. *JAIR* 4, 37–59 (1996)
4. Gomes, C.P., Hoffmann, J., Sabharwal, A., Selman, B.: From sampling to model counting. In: 20th IJCAI, Hyderabad, India, pp. 2293–2299 (January 2007)
5. Ihler, A.T., Fisher III, J.W., Willsky, A.S.: Loopy belief propagation: Convergence and effects of message errors. *JMLR* 6, 905–936 (2005)
6. Kersting, K., Ahmadi, B., Natarajan, S.: Counting belief propagation. In: Proc. of the 25th Conf. on Uncertainty in AI (UAI 2009), Montreal, Canada (2009)
7. Kschischang, F.R., Frey, B.J., Loeliger, H.-A.: Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory* 47 (2001)
8. Mezard, M., Montanari, A.: *Information, Physics, and Computation*. Oxford University Press, Inc., New York (2009)
9. Milch, B., Zettlemoyer, L., Kersting, K., Haimes, M., Pack Kaelbling, L.: Lifted Probabilistic Inference with Counting Formulas. In: Proc. of the 23rd AAAI Conf. on Artificial Intelligence, AAAI 2008 (July 13–17, 2008)
10. Montanari, A., Ricci-Tersenghi, F., Semerjian, G.: Solving constraint satisfaction problems through belief propagation-guided decimation. In: Proc. of the 45th Allerton Conference on Communications, Control and Computing (2007)
11. Nath, A., Domingos, P.: Efficient lifting for online probabilistic inference. In: Proceedings of the Twenty-Fourth AAAI Conference on AI, AAAI 2010 (2010)
12. Pearl, J.: *Reasoning in Intelligent Systems: Networks of Plausible Inference*, 2nd edn. Morgan Kaufmann, San Francisco (1991)
13. Richardson, M., Domingos, P.: Markov Logic Networks. *MLJ* 62, 107–136 (2006)
14. Selman, B., Kautz, H., Cohen, B.: Local search strategies for satisfiability testing. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pp. 521–532 (1995)
15. Sen, P., Deshpande, A., Getoor, L.: Bisimulation-based approximate lifted inference. In: Proc. of the 25th Conf. on Uncertainty in AI, UAI 2009 (2009)
16. Singla, P., Domingos, P.: Lifted First-Order Belief Propagation. In: Proc. of the 23rd AAAI Conf. on AI (AAAI 2008), pp. 1094–1099 (2008)

BetterRelations: Using a Game to Rate Linked Data Triples

Jörn Hees^{1,2}, Thomas Roth-Berghofer^{2,3}, Ralf Biedert²,
Benjamin Adrian², and Andreas Dengel^{1,2}

¹ Computer Science Department, University of Kaiserslautern, Germany

² Knowledge Management Department, DFKI GmbH, Kaiserslautern, Germany

³ Institute of Computer Science, University of Hildesheim, Germany

{firstname.lastname}@dfki.de

Abstract. While associations between concepts in our memory have different strengths, explicit strengths of links (edge weights) are missing in Linked Data. In order to build a collection of such edge weights, we created a web-game prototype that ranks triples by importance. In this paper we briefly describe the game, Linked Data preprocessing aspects, and the promising results of an evaluation of the game.

1 Introduction

Since its introduction in 2001 the Semantic Web [1] has gained much attention. In recent years, especially the Linking Open Data (LOD) project contributed many large, interlinked and publicly accessible RDF datasets, generating one of the world’s largest, decentralized knowledge bases. The accumulated amount of Linked Data has many applications and can already be used to answer structured questions (e.g., the DBpedia [2] dataset can easily be used to compile a list of musicians who were born in Berlin).

Currently it is impossible to rank result sets—not even those of simplistic (descriptive) queries—by importance as considered by an average human. For example, asked to describe (“What/Who is ...?”) *Facebook*, nearly all humans will explain that it *is an online social network*, but only few will tell us that *Chris Hughes is one of its co-founders*.¹ In the remainder of this paper, we will hence call the fact “Facebook has subject online social networking” to be more *important* than “Facebook has key person Chris Hughes”.

In order to overcome the knowledge acquisition bottleneck, which involves the manual generation of a dataset of such explicit importance ratings for many facts, we sketched the idea for a web-game in [3]. In this paper we present our experiences gathered from a web-game prototype called *BetterRelations* following the “Games With A Purpose” approach by von Ahn and Dabbish [4].

¹ In this paper we focus on an “average human’s” view, leaving the application of user and context models to future work.

2 Related Work

In terms of game design, BetterRelations is related to *Matchin* [5], which confronts both players with two pictures (taken from the WWW), asking them which one they prefer. In contrast, BetterRelations presents two textual facts about one topic to its players. Whereas Matchin returns a globally ranked list of images, BetterRelations creates a ranking for each topic and its related facts. In order to avoid forced decisions in cases of unknown or noisy facts, the GUI had to be extended, also causing the need to modify Matchin’s reward function in order to counter obvious cheating strategies.

OntoGame [6] was the first and most prominent game with a purpose focusing on Linked Data. Nevertheless, it collects another type of information than BetterRelations: Players are asked to decide if a Wikipedia topic is a class or an instance, aiming at creating a taxonomy of Wikipedia.

WhoKnows? [7], a *single player* game, judges whether an existing Linked Data triple is known by testing players with (amongst others) a multiple choice test or a hangman game. In contrast to our approach, WhoKnows only uses a limited fraction of the DBpedia dataset and excludes triples not matched by a predefined domain ontology in a preprocessing step. This greatly reduces noise issues, but eliminates the possibility to collect user feedback about triple qualities and problems in the extraction process. Also, WhoKnows intends to rank triples by degree of familiarity. However, the used measurement only relies on the ratio of correctly recognized facts divided by number of times a fact was tested. The quality of this ratio is doubtful as it does not distinguish whether a fact has been tested few or many times.

3 The Game

A straightforward approach to collect association strengths for Linked Data triples is this: First, we select a Linked Data resource of interest (e.g., `dbpedia:Facebook` or `dbpedia:Wiki`). We call this a *topic of interest* or simply *topic*. We then show randomly shuffled lists of all related triples to test persons and ask them to order the triples by decreasing importance. In the context of this work, given a topic, we define *related triples* to be the collection of (subject, predicate, object)-triples where the topic is the subject.²

The aforementioned approach suffers from the problem that the outcome of each of these experiments, which is a user centric ranking, is not only highly subjective, but sometimes even unstable for one person over time. In order to overcome difficulties for humans when sorting lengthy lists, we could ask for the atomic relative comparisons of two facts about one topic and then use an objective rating algorithm to generate an absolute ranking of the topic’s related facts. This leads us to the idea behind BetterRelations.

² Extending the list by triples where the topic is the object (incoming links) typically imports a large number of unimportant facts for the topic (e.g., in Wikipedia and thus in DBpedia one would expect to learn about Facebook by visiting the page about it, not by reading through all the pages linking to its page).



Fig. 1. In a game round, choosing phase

3.1 BetterRelations

*BetterRelations*³ is a symmetric two player output (decision) agreement game in terms of von Ahn and Dabbish's design principles for Games With A Purpose [4]:

A player starting to play the game is randomly matched with some other player for a predefined timespan (e.g., 2 minutes). In every round (see Figure 1) both players are presented with a *topic*, which actually is a Linked Data resource's symbol (e.g., *Facebook*, the symbol for [dbpedia:Facebook](#)), and two *items*, which are symbolic forms of facts about the topic (e.g., *key person Chris Hughes (Facebook)* and *has subject Online social networking*).

Both players are asked to select the fact that their partner will have thought of first. In case a player does not know the topic, a quick info can be requested by clicking on the question mark appended to the topic. Doing so will internally mark the player's decision as influenced and the partner's as unvalidated. To decide, each player can either click on the more important fact's button or on two additional buttons in case the player can't decide between the alternatives or thinks that both alternatives are nonsense / noise.

On the server side the game records a large amount of relative decisions between pairs of items, filtered by a partner and uses them to upgrade ratings in case of agreements. Internally, BetterRelations uses a TrueSkill [8] based algorithm to update fact ratings after each agreement, selects next fact pairs for a topic in a way to minimize the overall needed amount of decisions and stops sorting lists with n facts after $n \cdot \log_2(n)$ updates, determined to be a good threshold by simulations.

After rewarding the players with points, the next round starts until the game runs out of time. The next topic is chosen by selecting the topic least often played by both players from a list of topics currently opened for playing, which

³ BetterRelations can be played online: <http://lodgames.k1.dfki.de>

is based on the topmost accessed Wikipedia articles. In the end, both players see a summary of their performance showing the amount of points gained in this game, the longest streak and their total game score in BetterRelations.

In case no partner can be found or the partner leaves the Game, BetterRelations also provides a single player mode.

3.2 Game Data Acquisition and Preprocessing

In order to provide players with popular topics, BetterRelations selects topics (URI references, e.g., <http://dbpedia.org/resource/Facebook>) corresponding to the most often accessed Wikipedia pages⁴.

Each time the game needs a new game topic and its related triples (e.g., because an existing topic's facts were sorted), it loads the corresponding triples for the next topmost Wikipedia topic from a local DBpedia mirror, which also was pre-loaded with standard vocabularies such as `rdf`, `rdfs`, `foaf`.

As showing URIs to the end-users is of limited use, the users will always see `rdfs:labels` of such references. Triples having the same labels are merged from a game's point of view and such with missing labels for predicate or object excluded from the game.

Finally labels and corresponding triples are excluded, which (due to long string length) don't fit into the game's window, end with suspicious file endings (e.g., .jpeg) or which have an object label equal to the topic's label ("Facebook label Facebook").

4 Evaluation

BetterRelations was tested in an 18 day period in January 2011. In this time 1041 games were played by 359 users, resulting in over 4700 matches within an overall playtime of 42 human hours. From this we can estimate an *average lifetime play* of 7 minutes per player, a *throughput* of 112 matches per human hour of gaming, and an *expected contribution* of 13 matches per player⁵. Furthermore, with our current approach, we can estimate, that in order to sort the facts known about the top 1000 Wikipedia topics we would need about $313K$ matches or $23.9K$ players, so 24 players per Wikipedia topic.

We also compared the resulting ordering of facts with a manually created gold standard and found out that the rankings generated by BetterRelations can compete with those generated by human beings: In half of the cases (6/12) our approach won against the average single human's error. In three more it was approximately equal.

5 Conclusion

In this paper we presented results from implementing and testing BetterRelations, a game with a purpose which rates Linked Data triples by importance.

⁴ Stats aggregated from raw access logs, available at <http://dom.as/wikistats/>

⁵ Throughput, average lifetime play and expected contribution as in [4].

Our evaluation shows very promising results in terms of the desired and achieved high quality of the generated collection of importance ratings. However, the low average lifetime play indicates a problem with the game's fun factor. Based on a questionnaire we identified the high amount of noise in the underlying Linked Data triples to be the main problem (i.e., nonsense, unknown, and irrelevant facts).

As even slight improvements of the low average lifetime play could already drastically reduce the number of players needed to sort the facts known about a popular Wikipedia topic, our future work will focus on ways to reduce the amount of noise included in BetterRelations and other ways to increase the player's fun, such as including user accounts and high scores. We also plan to provide the game's output (ranked lists with rating scores) as Linked Data, allowing others to rank result sets of queries by importance for humans.

This work was financed in part by the University of Kaiserslautern PhD scholarship program and the BMBF project Perspecting (Grant 01IW08002).

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American* 284(5), 34–43 (2001)
2. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web* 7(3), 154–165 (2009)
3. Hees, J., Roth-Berghofer, T., Dengel, A.: Linked Data Games: Simulating Human Association with Linked Data. In: Atzmüller, M., Benz, D., Hotho, A., Stumme, G. (eds.) *LWA 2010*, Kassel, Germany (2010)
4. von Ahn, L., Dabbish, L.: Designing games with a purpose. *Communications of the ACM* 51(8), 58–67 (2008)
5. Hacker, S., von Ahn, L.: Matchin: Eliciting User Preferences with an Online Game. In: Proc. of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1207–1216. ACM, Boston (2009)
6. Siorpaes, K., Hepp, M.: OntoGame: Towards Overcoming the Incentive Bottleneck in Ontology Building. In: Chung, S., Herrero, P. (eds.) *OTM-WS 2007, Part II*. LNCS, vol. 4806, pp. 1222–1232. Springer, Heidelberg (2007)
7. Kny, E., Kölle, S., Töpper, G., Wittmers, E.: WhoKnows? (October 2010)
8. Herbrich, R., Minka, T., Graepel, T.: TrueSkill(TM): A Bayesian Skill Rating System. In: Schölkopf, B., Platt, J., Hoffmann, T. (eds.) *Advances in Neural Information Processing Systems*, vol. 19, pp. 569–576. MIT Press, Cambridge (2007)

Generic Performance Metrics for Continuous Activity Recognition

Albert Hein and Thomas Kirste

Dept. of Computer Science, University of Rostock, Germany

{albert.hein,thomas.kirste}@uni-rostock.de,

<http://mmis.informatik.uni-rostock.de>

Abstract. For evaluating activity recognition results still classical error metrics like *Accuracy*, *Precision*, and *Recall* are being used. They are well understood and widely accepted but entail fundamental problems: They can not handle fuzzy event boundaries, or parallel activities, and they over-emphasize decision boundaries. We introduce more generic performance metrics as replacement, allowing for soft classification and annotation while being backward compatible. We argue that they can increase the expressiveness and still allow more sophisticated methods like event and segment analysis.

Keywords: Performance Metrics, Activity Recognition.

1 Introduction

For evaluating the performance of an Activity Recognition system, the predicted activities have to be compared to some kind of ground truth, usually given by manual annotation. Although new performance measures like event- or segment-based analysis especially developed for temporally continuous Activity Recognition have been proposed years ago [1], most researchers still use the classical error metrics *Accuracy*, *Precision*, and *Recall*, complemented by example plots for visual analysis [2].

These metrics have been established more than 30 years ago in the field information retrieval [3]. As they are well known and understood their usage is widely accepted by researchers and they have been adapted to the field of activity recognition from its beginnings, despite they entail fundamental problems. As they are only applicable to frame by frame analysis, relevant errors remain invisible (see [4] for a more comprehensive list): variability in event lengths, fragmented events, and event merging. A second problem is, that they can not handle fuzzy event boundaries at transitions, co-occurring or parallel activities and the lack of rational crisp decision boundaries in prediction¹.

The latter are problems of crisp annotation and classification, which will be addressed in this paper. First, we will define generic performance metrics allowing

¹ A small difference in the probabilities of two possible activities leads to a weak or even instable decision disregarding uncertainty. This is wasting valuable information for analysis and may even produce new errors like virtual fragmentation.

both, soft and crisp classification and truth, or all together. We will then give two examples of visual analysis plots of truth and prediction over time for these cases. The first problems described above are intrinsic problems of frame by frame analysis and only solvable by event or segment analysis. We will demonstrate, that it is still possible to apply these methods on soft predictions and truth.

2 Performance Metrics

For our approach both, truth and prediction are represented as the *Categorical Probability Distribution* over all activity classes C , given by the probability of belonging to a certain activity class $c \in C$ at a certain frame t in time T . The sum of all probabilities at one frame is $\sum_{c \in C} \text{truth}_t^c = \sum_{c \in C} \text{pred}_t^c = 1$. This is the natural representation for soft classification and annotation. However, crisp trajectories of prediction and truth can easily be converted by letting the corresponding value be 1 and set the rest to 0:

$$\forall c \in C, \text{truth}_t^c = \begin{cases} 1 & \text{if } \text{truth}_t^{crisp} = c \\ 0 & \text{otherwise} \end{cases} \quad \forall c \in C, \text{pred}_t^c = \begin{cases} 1 & \text{if } \text{pred}_t^{crisp} = c \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Derived from the confusion matrix of binary classification problems the counted numbers of true positives (tp), false positives (fp), true negatives (tn), and false negatives (fn) are the basis for calculating a variety of performance metrics. Instead of counting, we now define these values class-wise in a generic way:

Definition 1. *The class-wise ratios true positives (tp^c), false positives (fp^c), true negatives (tn^c), and false negatives (fn^c) are given by*

$$tp^c = \sum_{t=1}^T \text{truth}_t^c \times \text{pred}_t^c \quad (2)$$

$$fp^c = \sum_{t=1}^T (1 - \text{truth}_t^c) \times \text{pred}_t^c \quad (3)$$

$$tn^c = \sum_{t=1}^T (1 - \text{truth}_t^c) \times (1 - \text{pred}_t^c) \quad (4)$$

$$fn^c = \sum_{t=1}^T \text{truth}_t^c \times (1 - \text{pred}_t^c) \quad (5)$$

For multi-class situations these class-wise values can be aggregated to a weighted average using a weighting factor w^c :

Definition 2. *The aggregated ratios true positives (tp), false positives (fp), true negatives (tn), false negatives (fn) are given by*

$$tp = \sum_{c \in C} \frac{tp^c}{w^c} \quad fp = \sum_{c \in C} \frac{fp^c}{w^c} \quad tn = \sum_{c \in C} \frac{tn^c}{w^c} \quad fn = \sum_{c \in C} \frac{fn^c}{w^c} \quad (6)$$

where $w^c = \sum_{t=1}^T \text{truth}_t^c$.

Now all classical summary statistics can be computed from these values, as there are Accuracy, Precision, Recall, Specificity, Negative Predictive Value (*NPV*), F-Measure (*f*), and Likelihood Ratio (*LR+* and *LR-*) as the most common ones presented below²:

$$\text{accuracy} = \frac{tp + tn}{tp + fp + tn + fn} \quad (7)$$

$$\text{precision} = \frac{tp}{tp + fp} \quad (8)$$

$$\text{recall} = \frac{tp}{tp + fn} \quad (9)$$

$$\text{specificity} = \frac{tn}{fp + tn} \quad (10)$$

$$NPV = \frac{tn}{tn + fn} \quad (11)$$

$$f = \frac{(\beta^2 + 1) \times \text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}} \quad (12)$$

$$LR+ = \frac{tp + tn}{tp + fp + tn + fn} \quad (13)$$

$$LR- = \frac{tp + tn}{tp + fp + tn + fn} \quad (14)$$

3 Event and Segment Analysis

To address the problems of a frame by frame analysis of the activity recognition results, none of the statistics above is applicable. Fortunately the more sophisticated evaluation methods developed by Ward, Minnen and others^{2,4,11} based on events and segments can still be utilized on smooth predictions and annotations after converting them back to crisp values. This conversion can be done in a voting preprocess using the following equations:

$$\text{votetruth}_t = c \in C | \forall x \in C, x \neq c : \text{truth}_t^c \geq \text{truth}_t^x. \quad (15)$$

$$\text{votepred}_t = c \in C | \forall x \in C, x \neq c : \text{pred}_t^c \geq \text{pred}_t^x. \quad (16)$$

Of course, this step may reintroduce problems like virtual fragmentation again as it does not make use of soft decision and event boundaries or allow co-occurring or parallel activities.

4 Visual Analysis

As the visual comparison of the ground truth with the predicted activities is very common, it is important that it is still possible to create descriptive plots with soft annotations and/or classification results. We suggest a heatmap representation and divergence plots as shown in fig. 11. While the visual detection of errors is simplified, crisp trajectories may be overplotted for better understanding. In both plots color schemes established in bioinformatics have been utilized.

² For a more detailed overview see [1].

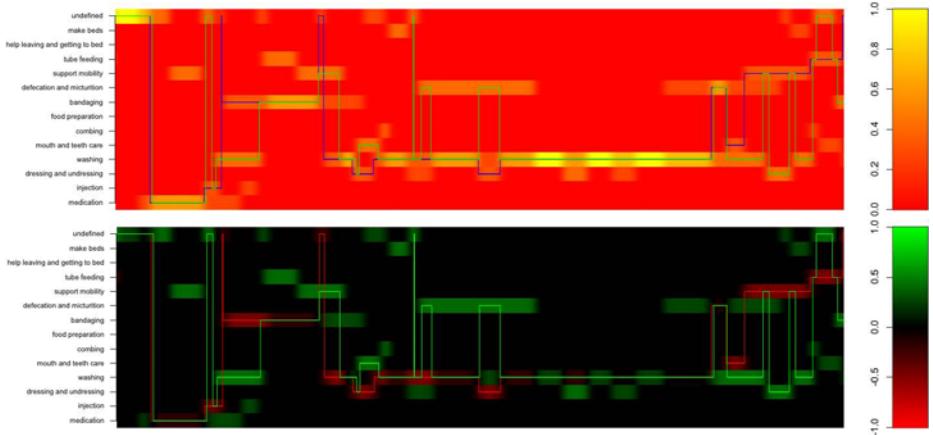


Fig. 1. *Heatmap* (top) and *Divergence Plot* (bottom) combining soft and overplotted crisp annotations and prediction. In the heatmap high class probabilities are shown in yellow, low in red. Crisp values are overplotted as lines - blue for truth and green for prediction. At the bottom correct values are shown in black, false positives in green, and false negatives in red. Again, crisp values are plotted as green (prediction) and red (truth) lines.

The heatmap in the upper image shows the class predictions for a multi-class Activities of Daily Living recognition example. High probabilities are plotted in yellow, low in red. Crisp truth and prediction have been calculated using Eqn. I5 and I6. They are overplotted as blue (truth) and green (prediction) line, vertical connections do not show interpolated values but have been left in for the better orientation of the eye and for emphasizing outliers.

For comparing soft truth and prediction values a divergence plot is used in the lower image. Black is used whenever the predicted class probability matches the annotation while green areas are showing false positive and red false negative values. Again calculated crisp truth (red) and prediction (green) have been overplotted. The divergence plot directly resembles the commonly used line based plots: As can be seen in the figure normally the red line only gets visible in false negative areas, while false positives are still emphasized in bright green. We argue that using these types of plots for crisp, mixed, and soft annotation and classification task, it is easier to get an understanding of the data and to localize prediction errors.

5 Conclusion

In this paper we introduced a set of generic performance metrics which allow the evaluation of activity recognition results with suboptimal but very common and well understood performance measures like accuracy or precision and recall using soft classification and annotations. The new metrics avoid intrinsic fuzzy event and decision boundary issues of crisp trajectories and allow for co-occurring

and parallel activities. Although they are designed for fuzzy annotation and classification they are perfectly backward compatible with crisp cases, so that they can act as drop in replacements and even work in mixed occurrences. Also the expressiveness of example plots over time gets increased by the use of soft values. Finally, we showed that it is still possible to utilize event and segment analysis methods by applying a voting proprocess, although a natural possible next step would be to extend the suggested metrics for native support of event and sequence evaluation.

References

1. Minnen, D., Westeyn, T., Starner, T.: Performance metrics and evaluation issues for continuous activity recognition. In: Proceedings of Performance Metrics for Intelligent Systems (PerMis 2006), Gaithersburg, MD (2006)
2. Ward, J.A., Lukowicz, P., Gellersen, H.W.: Performance metrics for activity recognition. *Transactions on Information Systems and Technology (TIST)* 2(1) (2011)
3. van Rijsbergen, C.J.: Information retrieval, 2nd edn. Butterworths, London (1979)
4. Ward, J.A., Lukowicz, P., Tröster, G.: Evaluating performance in continuous context recognition using event-driven error characterisation. In: Hazas, M., Krumm, J., Strang, T. (eds.) LoCA 2006. LNCS, vol. 3987, pp. 239–255. Springer, Heidelberg (2006)

Bayesian Logic Networks and the Search for Samples with Backward Simulation and Abstract Constraint Learning

Dominik Jain, Klaus von Gleissenthall, and Michael Beetz

Intelligent Autonomous Systems, Department of Informatics, Technische Universität München

Abstract. With Bayesian logic networks (BLNs), we present a practical representation formalism for statistical relational knowledge. Based on the concept of mixed networks with probabilistic and deterministic constraints, BLNs combine the probabilistic semantics of (relational) Bayesian networks with constraints in first-order logic. In practical applications, efficient inference in statistical relational models such as BLNs is a key concern. Motivated by the inherently mixed nature of models instantiated from BLNs, we investigate two novel importance sampling methods: The first combines backward simulation, i.e. sampling backward from the evidence, with systematic search, while the second explores the possibility of recording abstract constraints during the search for samples.

1 Introduction

When modelling real-world domains in the context of high-level AI applications, where both expressivity and tractability are key, we typically need to be able to cope with manifold relations concerning varying entities that are subject to uncertainty. Representation formalisms must therefore combine probabilistic semantics with ways of abstractly specifying rules that generalize across domains of relevant objects. In the field that has emerged as statistical relational learning and reasoning, a number of such formalisms have been proposed in recent years.

Among the most expressive such formalisms are Markov logic networks (MLNs), which elegantly extend first-order logic to a probabilistic setting by attaching weights to formulas [1]. The weighted formulas collectively represent a template for the construction of undirected graphical models (i.e. Markov random fields). Unfortunately, parameter learning in MLNs is an ill-posed problem [2] and approximate inference is typically expensive even for conceptually simple queries. Many alternative approaches [3][4][5] are based on Bayesian networks, which are often easier to deal with – both in terms of learning and inference. However, these formalisms tend to sacrifice expressivity¹ for tractability. Typically, one can easily express only local probabilistic constraints, while even simple relational properties required on a global level, such as the transitivity or symmetry of an uncertain relation, cannot easily be modelled. While, in theory, we are able to represent most such properties using most representation formalisms [6],

¹ We here understand *expressivity* not merely as the ability to express but rather as the ability to express *in concise terms*.

we cannot, unfortunately, do so in practice without substantially increasing the complexity of the first-order model (which, notably, does not necessarily coincide with the complexity of ground models instantiated from it, however). From a knowledge engineering perspective, model simplicity and conciseness are key.

The representation formalism we describe in this work, Bayesian logic networks (BLNs), is a reasonable compromise in this regard. Its probabilistic components are based on conditional probability distribution templates (for the construction of a Bayesian network). Global constraints are supported by another model component, a template for the construction of a constraint network, in which we represent deterministic constraints using first-order logic.

Since learning can usually be done offline, especially the question of efficient inference in statistical relational models such as BLNs is a key concern. Exact methods being inapplicable in many larger, more complex models, sampling-based approaches are particularly widely used methods that allow to obtain any-time approximations. However, in models with determinism, sampling has constraint satisfaction as a subproblem and performance can be significantly limited due to the *rejection problem* (i.e. the problem of having to discard samples because they have zero probability). In this paper, we investigate two novel importance sampling methods that explicitly consider determinism: The first combines backward simulation, i.e. sampling backward from the evidence, with systematic search; the second explores the possibility of recording abstract constraints (*nogoods*) during the search for samples. The consideration of abstract constraints that are applicable to more than one context seems natural given the large number of repeated substructures typically found in instantiations of relational models.

The remainder of this work is organized as follows: In the following section, we review the fundamental graphical models. In Section 3 we provide details on Bayesian logic networks. In Section 4 we review standard sampling methods and introduce the two novel methods outlined above. We subsequently report on the performance of these methods in comparison to other methods on a variety of problem instances in Section 5. We conclude with an outlook on future work.

2 From Bayesian to Mixed Networks

Bayesian Networks. A Bayesian network is a tuple $B = \langle X, D, G, P \rangle$, where $X = \{X_1, \dots, X_n\}$ is an ordered set of random variables, $D = \{D_1, \dots, D_n\}$ is the corresponding set of domains, $G = \langle X, E \rangle$ is a directed acyclic graph representing a dependency structure over the variables X , and $P = \{P_1, \dots, P_n\}$ is a set of (conditional) probability distributions with $P_i = P(X_i \mid \text{Par}_{X_i})$, where Par_{X_i} denotes the set of parents of X_i in G . Let $\mathcal{X} = \text{dom}(X) = \prod_i D_i$ be the set of possible worlds. B represents a probability distribution over \mathcal{X} as a product of entries in P : For all $x \in \mathcal{X}$, $P(x) = \prod_i P(x_i \mid \text{par}_{X_i}^x)$, where $\text{par}_{X_i}^x$ is the assignment of X_i 's parents in x . Note that we write x as shorthand for $X = x$ and similarly for other (sets of) variables.

Mixed Networks. Mixed networks [7] extend Bayesian networks with explicit representations of deterministic constraints – simply by coupling them with constraint networks. Formally, a *mixed network* M is a pair $\langle B, R \rangle$, where $B = \langle X, D, G, P \rangle$ is a Bayesian

network representing the joint probability distribution $P_B(x)$ and R is a constraint network. R is a tuple $\langle X, D, C \rangle$ where X and D are shared with B , and $C = \{C_i\}$ is a set of constraints, each constraint C_i being a pair $\langle S_i, R_i \rangle$. $S_i \subseteq X$ is the scope of the constraint, and $R_i \subseteq \prod_{X_j \in S_i} D_j$ is a relation denoting the allowed combinations of values. R is a representation of the set of assignments satisfying all constraints, denoted as ρ . The mixed network M specifies a probability distribution over \mathcal{X} as follows:

$$P_M(x) = \begin{cases} P_B(x) / \sum_{x' \in \rho} P_B(x') & \text{if } x \in \rho \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Assignments that do not satisfy the constraints in R are thus phased out and the remainder of the distribution represented by B is renormalized.

3 Bayesian Logic Networks

Bayesian logic networks (BLNs) extend the notion of mixed networks to a relational setting. Thus, a BLN can be thought of as a meta-model that represents a template for the construction of a mixed network. The model as such defines general principles that are universally applicable (akin to universal quantification in first-order logic) and that determine, for an arbitrary number of concrete entities, a concrete mixed network. The random variables appearing in a BLN are thus abstract, parametrized random variables, and the model defines which principles to apply in order to construct concrete random variables.

Formally, a BLN is a tuple $\mathcal{B} = \langle \mathcal{D}, \mathcal{F}, \mathcal{L} \rangle$, where

- $\mathcal{D} = \langle \mathcal{T}, \mathcal{S}, \mathcal{E}, t \rangle$ comprises the model's fundamental *declarations*. \mathcal{T} is a taxonomy of types, which is represented as a directed forest (T, I) , where T is the actual set of types and $I \subset T \times T$ is the *generalizes* relation (inverse is-a), i.e. $(T_i, T_j) \in I$ iff T_i is a generalization of T_j . \mathcal{S} is a set of signatures of functions, and \mathcal{E} is a set of (abstract) entities that are to exist in all instantiations, whose types are given by the function $t : \mathcal{E} \rightarrow 2^T \setminus \{\emptyset\}$ which maps every entity to the non-empty subset of types $T = \{T_1, \dots, T_{|T|}\}$ it belongs to. The function t thus induces a cover of the set of entities with sets $E_{T_i} = \{e \in \mathcal{E} \mid T_i \in t(e)\}$. (We assume that t is consistent with \mathcal{T} , i.e. if $(T_i, T_j) \in I$, then $e \in E_{T_j}$ implies $e \in E_{T_i}$.)

The set \mathcal{S} contains the signature of every function f , defining the domain and the range of the function in terms of types, i.e.

$$(f, (T_{i_1}, \dots, T_{i_n}), T_r) \in \mathcal{S} \Leftrightarrow f : E_{T_{i_1}} \times \dots \times E_{T_{i_n}} \rightarrow E_{T_r}$$

Logical predicates are simply Boolean functions, i.e. functions that map to $E_{T_r} = \mathbb{B}$, and we implicitly assume that the corresponding type symbol *Boolean* is always contained in T and that $\mathbb{B} = \{\text{True}, \text{False}\} \subseteq E$.

We regard a set E_{T_r} that corresponds to a type $T_r \in T$ which appears as the return type of a function as a (fixed) *domain*, i.e. as a fixed set of entities that must be fully contained in E , whereas the extensions of other types may vary from instantiation to instantiation (and the corresponding subsets of E may even be empty).

- \mathcal{F} is a set of *fragments* of conditional probability distributions. Every fragment defines a dependency of an abstract random variable $f(p_1, \dots, p_n)$ (the *fragment variable*) on a set of other abstract random variables (the *parents*), where f is one of the functions defined in S , and the parameters p_1, \dots, p_n are either variables typed according to f 's signature or entities in E belonging to the respective type. The dependencies are encoded in a *conditional probability function* (CPF), which defines, for every setting of the parent variables (i.e. every element in the domain product of the parent variables, as specified by the ranges in the functions' signatures), a probability distribution over the elements in the domain of the fragment variable (i.e. the range of f).

Additionally, a fragment may define preconditions for its applicability, which may involve arbitrary logical statements about the parameters p_1, \dots, p_n (or parameters that can be functionally determined by these).

- The set \mathcal{L} consists of formulas in first-order logic (with equality) over the functions/predicates defined in S , which represent hard deterministic dependencies. Such formulas may help us to model global constraints that cannot concisely be represented by the conditional probability fragments in the set \mathcal{F} .

Instantiation. For any given set of entities E' , whose types are given by a function $t' : E' \rightarrow 2^T \setminus \emptyset$, a Bayesian logic network \mathcal{B} defines a *ground mixed network* $M_{\mathcal{B}, E'} = \langle \langle X, D, G, P \rangle, \langle X, D, C \rangle \rangle$ as follows:

- E and t in \mathcal{B} are augmented to include E', t' .
- The set of random variables X contains, for each function $(f, (T_{i_1}, \dots, T_{i_n}), T_r) \in S$ and each tuple of applicable entities $(e_1, \dots, e_n) \in E_{T_{i_1}} \times \dots \times E_{T_{i_n}}$, one element $X_i = f(e_1, \dots, e_n)$. The corresponding domain $D_i \in D$ is simply E_{T_r} .
- The conditional probability function $P_i \in P$ that is applicable to a random variable $X_i = f(e_1, \dots, e_n)$ is determined by \mathcal{F} , which must either contain exactly one fragment for f whose preconditions are met given the actual parameters or must specify a *combining rule* [4] (e.g. noisy-or) that defines how to combine several fragments into a single conditional distribution. The connectivity of the directed graph G is such that there is a directed edge from every parent to the fragment variable – as indicated by the applicable fragments.
- For every grounding of every formula in \mathcal{L} (obtained by substituting quantified variables by the applicable elements of E accordingly), the set C contains one constraint $C_i = \langle S_i, R_i \rangle$, where S_i , the scope of the constraint, is the set of random variables mentioned in the ground formula, and R_i is the relation indicating the combinations of values for which the formula is satisfied.

In the special case where $\mathcal{L} = \emptyset$, the mixed network contains no explicit constraints and is thus equivalent to the Bayesian network $\langle X, D, G, P \rangle$.

Some algorithms can essentially operate directly on mixed networks [7, 8]. For purposes of inference, we may also convert any mixed network $M_{\mathcal{B}, E'}$ into an *auxiliary Bayesian network* $B_{\mathcal{B}, E'}$, allowing us to leverage the large body of inference methods available for Bayesian networks. $B_{\mathcal{B}, E'}$ is constructed from $M_{\mathcal{B}, E'}$ by adding to X for every constraint $C_i = \langle S_i, R_i \rangle \in C$ a Boolean auxiliary variable A_i that represents the

corresponding constraint and has as its parents in G the set of nodes S_i . The probability function associated with A_i is to return a probability of 1 as the value for *True* for every parent configuration contained in R_i and 0 otherwise. Since all constraints are required to be satisfied, when we perform inference in the auxiliary Bayesian network, we condition on the auxiliary variables, requiring that they all take on a value of *True*. Therefore, if $|C| = k$, we have

$$P_{M_{B,E'}}(X = x) = P_{B_{B,E'}}(X = x \mid A_1 = \text{True}, \dots, A_k = \text{True}) \quad (2)$$

BLNs are implemented in the open-source toolbox PROBCOG. In particular, the implementation uses a graphical representation for the set of fragments \mathcal{F} (see Fig. 1) and allows to conveniently augment evidence based on Prolog rules, which, given seed evidence, can compute additional pieces of evidence. For further details, we refer to the project homepage².

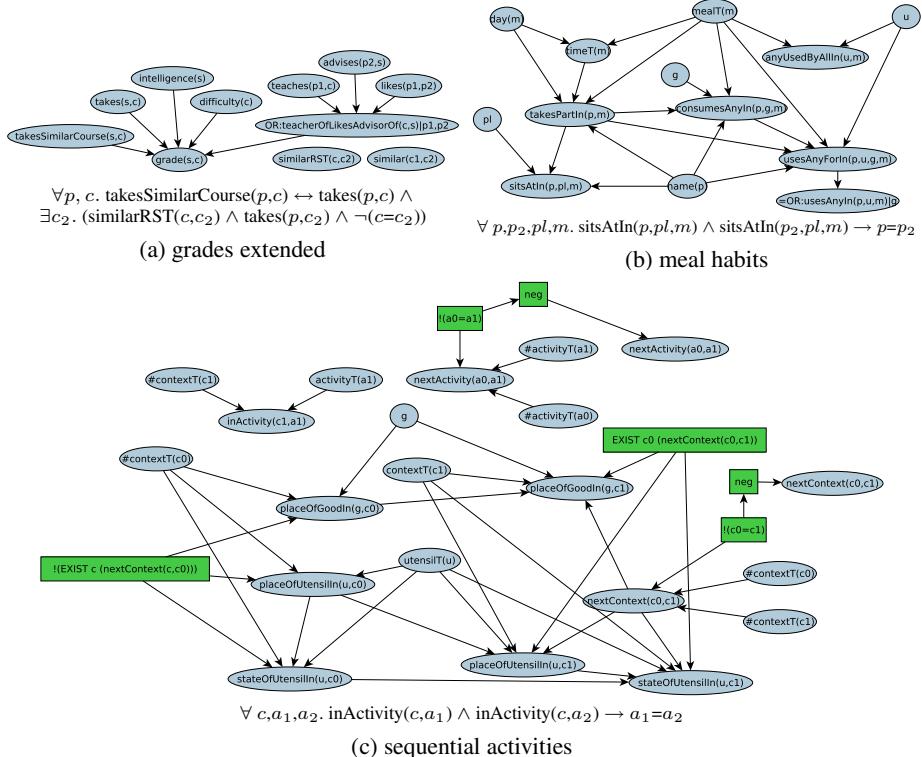


Fig. 1. Excerpts of three BLN models: A graphical representation of the set of fragments \mathcal{F} is shown (elliptical nodes are associated with conditional probability functions, rectangular nodes indicate preconditions for fragments to be applicable) along with one exemplary logical formula from the set \mathcal{L} for each model. These models are used in the experiments of Section 5 where they are briefly described.

² <https://ias.in.tum.de/probcog-wiki>

4 Sampling Techniques

In this section, we propose inference algorithms that are specifically tailored towards applicability in probabilistic models with determinism as we obtain them by instantiating a BLN. We consider the inference task of computing the distribution of one or more query variables $Q \subset X$ given an assignment to evidence variables $E \subset X$, i.e. to compute the posterior distribution $P(Q | E = e)$. First, we review the fundamental sampling techniques.

4.1 Fundamental Sampling Techniques

Rejection Sampling. In Bayesian networks, sampling from the prior $P(x)$ is particularly simple: If random variables are ordered topologically with respect to G , we sample from each (conditional) distribution in order to generate a full assignment to all variables in X . Let $S = \{x^{(1)}, \dots, x^{(N)}\}$ be a set of samples taken from $P(x)$. For sufficiently large N , the relative frequency of an assignment will be an approximation of its probability, i.e. for all $x \in \mathcal{X}$, $n(S, x)/N \approx P(x)$, where $n(S, x)$ is the number of samples in S satisfying the assignment x . We can therefore compute a query as $P(q | e) \approx n(S, q \wedge e)/n(S, e)$. Since all samples that do not satisfy e are irrelevant to our query, they can be ignored (and are consequently rejected by the sampler).

Importance Sampling. Importance sampling [9] allows to reduce the problem of rejections by sampling from some *importance function* $Q(x)$ instead of the prior $P(x)$, the idea being that $Q(x)$ should somehow use the evidence and the structure of the model to avoid sampling irrelevant assignments. If Q is a probability distribution, the only additional constraint that needs to be placed on Q is that it must not assign zero probability to any $x \in \mathcal{X}$ for which $P(x | e) \neq 0$. Given a set of samples S (as above but now taken from $Q(x)$), we have that for all $x \in \mathcal{X}$, $n(S, x)/N \approx Q(x)$, and therefore $n(S, x)/N \cdot P(x)/Q(x) \approx P(x)$ for all x satisfying $E = e$. Thus a query can be computed as

$$P(q | e) \approx \frac{\sum_{x \in \mathcal{X}, x \models q \wedge e} n(S, x)/N \cdot P(x)/Q(x)}{\sum_{x \in \mathcal{X}, x \models e} n(S, x)/N \cdot P(x)/Q(x)} = \frac{\sum_{x^{(i)} \in S, x^{(i)} \models q \wedge e} w(x^{(i)})}{\sum_{x^{(i)} \in S, x^{(i)} \models e} w(x^{(i)})} \quad (3)$$

where $w(x) := P(x)/Q(x)$. Therefore, by assigning a weight of $P(x)/Q(x)$ to each sample, we compensate having sampled from Q instead of P and can approximate the desired posterior.

Backward Simulation. Fung and Del Favero [10] presented one way of focusing the importance function Q on the evidence. Instead of sampling forward in a topological ordering, we can sample backward from the evidence, i.e. we can assign the parents of a node X_i whose value is already given by sampling an assignment to Par_{X_i} according to the conditional distribution of X_i : We sample $\text{Par}_{X_i} = \text{par}_{X_i}$ with probability proportional to $P(x_i | \text{par}_{X_i})$. The backward sampling process is repeated until all

ancestors of evidence nodes have been instantiated. Any remaining nodes to which no values have been assigned are forward sampled to obtain a complete assignment to all variables.

The algorithm first determines a sampling order \mathcal{O} . This step partitions the set of nodes into three sets: backward sampled nodes \mathcal{O}_B , forward sampled nodes \mathcal{O}_F and unsampled nodes \mathcal{O}_O that are outside the sampling order. To compute an ordering, we manage a list of backward sampling candidates – initially the list of evidence nodes. For each candidate X_i , we determine whether any of its parents are yet uninstantiated and if so, we add it to \mathcal{O} as a backward sampled node and add its parents to the list of candidates (as during sampling, X_i 's value will be used to *instantiate* the previously uninstantiated parents). Otherwise the node is outside the actual sampling order and is part of \mathcal{O}_O . All remaining uninstantiated nodes are forward sampled (make up the set \mathcal{O}_F) and are added to the sampling order in topological order (a forward sampled node instantiates itself).

Given the semantics of forward and backward sampling, the importance distribution of backward simulation is

$$Q(x) = \prod_{X_i \in \mathcal{O}_B} \frac{P(X_i = x_i \mid \text{par}_{X_i}^x)}{Z_i} \cdot \prod_{X_i \in \mathcal{O}_F} P(X_i = x_i \mid \text{par}_{X_i}^x) \quad (4)$$

where Z_i is a normalization constant. Thus sample weights are computed using

$$w(x) = \frac{P(x)}{Q(x)} = \prod_{X_i \in \mathcal{O}_B} Z_i \prod_{X_i \in \mathcal{O}_O} P(X_i = x_i \mid \text{par}_{X_i}^x) \quad (5)$$

SampleSearch. Even though importance sampling techniques such as backward simulation can considerably reduce the rejection problem, there may still be too many rejections to obtain a sufficient number of samples within a reasonable time frame. To address this problem, Gogate and Dechter [11] proposed the SampleSearch scheme, which systematically searches for samples. Rather than discarding a sample as soon as its weight becomes zero and restarting, we can go back and reassign variables until we find an assignment that has a non-zero weight.

The simplest version of SampleSearch samples from the prior $P(x)$, consecutively assigning values to each random variable in topological order. If it encounters an evidence variable where the probability of the evidence is 0 given the values previously assigned to the variable's parents, we backtrack to the previous variable in the topological ordering, excluding the value that was previously assigned, renormalizing the conditional distribution and choosing a different value. Should all the values within a domain have been excluded, we backtrack further – and so on, until all variables have been assigned a value that is compatible with the evidence. Of course, more elaborate versions of SampleSearch can make use of advanced techniques known from satisfiability testing [10] or constraint satisfaction problems (e.g. arc consistency and conflict-driven backjumping [12]), and, instead of the prior $P(x)$, we could use an importance function computed via a message-passing algorithm such as loopy belief propagation.

It is important to note that the probability $Q(x)$ with which a sample is selected in SampleSearch cannot directly be used to compute weights that will yield an asymptotically unbiased estimator for the desired posterior. We may select the same sample with

different probabilities depending on the exclusions we had to make in order to obtain it. In effect, the algorithm samples from the backtrack-free distribution and hence the highest possible probability (maximum number of exclusions) in each sampling step will yield an unbiased estimator (see [11] for proof). This probability can be approximated using the set of samples collected.

4.2 Backward SampleSearch

In the following, we combine the notion of backward simulation with a sample search scheme to obtain *Backward SampleSearch* (BSS), i.e. we perform backward simulation and apply backtracking whenever the sample weight becomes zero – until we find an assignment that represents a solution to the induced constraint satisfaction problem.

Since chronological backtracking is typically highly inefficient, we use conflict-directed backjumping [12] in the variant BSS-BJ. Whenever we encounter a partial assignment that cannot be completed to a full assignment, we jump back to the node with the highest index in the sampling order that has bearing on the conflict that we encountered. With Algorithm 1 we provide pseudocode for the generation of a sample with BSS-BJ (some of the symbols used are introduced further on). Line 11 performs, for $X_i \in \mathcal{O}_B \cup \mathcal{O}_F$, either backward sampling or forward sampling (using the given exclusions and returning a new state), and, for $X_i \in \mathcal{O}_O$, it checks whether the current assignment is valid given the previously sampled parents of X_i . In `sampledIndices[i]`, we store the index of the value (node assignment) that we sampled from the respective distribution. Lines 14–22 realize conflict-directed backjumping for the specific semantics of backward simulation. The set B_i is used to collect order indices of candidate nodes for a backjump from X_i .

In our implementation, we augment the distribution we use for backward sampling using a more informed estimate of the probability of a parent assignment. The original backward simulation algorithm goes backward “blindly” in the sense that it does not explicitly consider any information from evidence that is further up – beyond the parents of the variable X_i that is backward sampled. As a computationally simple improvement, we use, for backward sampling, a crude approximate belief $\tilde{P}(X_j = x_j)$ for parents X_j of X_i in addition to the conditional distribution $P(x_i \mid \text{par}_{X_i})$. \tilde{P} propagates evidence only forward. If $X_i \in E$, then $\tilde{P}(X_i = x_i) = 1$ if $E = e \models X_i = x_i$ and 0 otherwise. If $X_i \notin E$, we compute it as

$$\tilde{P}(X_i = x_i) := \sum_{\bar{x} \in \text{dom}(\bar{X})} P(X_i = x_i \mid \bar{x}, \text{par}_{X_i}^e) \cdot \prod_{X_j \in \bar{X}} \tilde{P}(X_j = \bar{x}_j) \quad (6)$$

where $\bar{X} = \text{Par}_{X_i} \setminus E$ and $\text{par}_{X_i}^e$ is the assignment of $\text{Par}_{X_i} \cap E$ in e .

When backward sampling from X_i , some of X_i ’s parents are yet uninstantiated while others may already be instantiated (either because they are evidence variables or because they were instantiated by a previous backward sampling step, as determined by the sampling order). Par_{X_i} is partitioned into two sets U^{X_i} and I^{X_i} accordingly. We sample the yet uninstantiated parents U^{X_i} of X_i given previous assignments via

$$Q(U^{X_i} = u^{X_i} \mid \cdot) = \frac{1}{Z_i} P(X_i = x_i \mid U^{X_i} = u^{X_i}, I^{X_i} = i^{X_i}) \prod_{X_j \in U^{X_i}} \tilde{P}(X_j = u_j^{X_i}) \quad (7)$$

where Z_i is a normalization constant and $u_j^{X_i}$ is X_j 's value in u^{X_i} . Forward sampled nodes are treated as in backward simulation. Therefore, we can compute sample weights as follows:

$$w(x) = \prod_{X_i \in \mathcal{O}_B} \frac{Z_i}{\prod_{X_j \in U^{X_i}} \tilde{P}(X_j = x_j)} \prod_{X_i \in \mathcal{O}_O} P(X_i = x_i \mid \text{par}_{X_i}^x) \quad (8)$$

As in SampleSearch, directly using these weights will yield a biased estimator. Fortunately, we can obtain an unbiased estimator by keeping track of the highest probability values with which each particular partial assignment was sampled (cf. ), which we obtain for the smallest Z_i s in Eq. 7 we observed in the sampling process.

A similar sample searching method has not been previously investigated. However, the backward approach is loosely related to formula-based inference , as the backward sampling of a node can be viewed as treating parts of the sampling problem at a formula level – since backward sampling assigns several variables (the yet uninstantiated parents) at once according to the constraint imposed by the child's setting.

Algorithm 1. SAMPLE-BSS-BJ

```

1: state ← empty assignment
2:  $i \leftarrow 0$ 
3: backtracking ← false
4: while  $i < |X|$  do
5:    $X_i \leftarrow \text{nodes}[\text{samplingOrder}[i]]$ 
6:   if backtracking then
7:     exclusions[i].append(sampledIndices[i])
8:   else
9:     exclusions[i] ← empty list
10:     $B_i \leftarrow \emptyset$ 
11:    sampledIndices[i], state ← sample(state,  $X_i$ , exclusions[i])
12:    if sampledIndices[i] is null then
13:      backtracking ← true
14:      if  $X_i \in \mathcal{O}_B$  then
15:         $B_i \leftarrow B_i \cup \{j \mid \text{The } j\text{-th node in the order instantiated } X_i \text{ or a node in } I^{X_i}\}$ 
16:      else
17:         $B_i \leftarrow B_i \cup \{j \mid \text{The } j\text{-th node in the order instantiated a node in } \text{Par}_{X_i}\}$ 
18:        if  $X_i \in \mathcal{O}_O$  then  $B_i \leftarrow B_i \cup \{j \mid \text{The } j\text{-th node in the order instantiated } X_i\}$ 
19:        if  $B_i = \emptyset$  then return “inconsistent evidence”
20:         $i_{\text{prev}} \leftarrow i$ 
21:         $i \leftarrow \max B_i$ 
22:         $B_i \leftarrow B_i \cup B_{i_{\text{prev}}} \setminus \{i\}$ 
23:    else
24:      backtracking ← false
25:       $i \leftarrow i + 1$ 
26: return state

```

4.3 SampleSearch with Abstract Constraint Learning

When sampling, we are repeatedly faced with the same problem – generating a sample from a distribution – and when applying SampleSearch, we thus repeatedly encounter the same dead-ends and are forced to backtrack in precisely the same way. Even during the generation of a single sample, we may encounter the same local dead-end several times. In the constraint solving community, this has given rise to the notion of constraint (or *nogood*) learning [12] – the natural principle of learning from one’s past mistakes. The combination of SampleSearch with constraint learning is essentially straightforward: Whenever we backtrack/backjump, we record a constraint that forbids the configuration that caused the dead-end, and whenever an assignment that is relevant to the constraint is made in the future, we check whether the constraint is applicable and if so, we add a domain exclusion. Should all domain values have been exhausted, we backtrack, recursively generating further constraints.

In instances of relational models such as BLNs, we not only encounter specific dead-ends repeatedly as described above but also dead-ends that pertain to different variables but are otherwise completely analogous. This is due to the repeated substructures found within instantiations of a relational model. Thus, the recording of abstract constraints that allow to represent such analogous constraints in a single form suggests itself. The use of abstract constraints potentially has several advantages:

- There are fewer abstract constraints and fewer constraints are more quickly found. Therefore, the search for samples can potentially be speeded up.
- In the best case, the recording of constraints successfully trades speed for memory usage. With fewer constraints, an unrestricted recording of abstract constraints may still be feasible, while the number of propositional constraints may already be too large to fit in memory.
- Because they are not specific to a particular instantiation, abstract constraints can be saved and stored for future use in order to apply them across different instantiations of the same relational model.

For purposes of abstraction, we consider an equivalence relation over the random variables X . In BLNs, we can uniquely identify the CPF that was generated from the fragments for any variable X_i and use this information – along with information on evidence – to define the relation. To obtain an equivalence relation that is guaranteed to be correct with respect to the domain exclusions it will later result in, we can use a colour-passing scheme analogous to the one described in [14]. In practice, simpler equivalence relations based, for example, on the CPF identifiers of all the variables in a variable’s Markov blanket often suffice, because information on relevant evidence will also be represented within the constraints themselves.

We represent an abstract constraint as a graph whose nodes are equivalence classes and whose edges indicate the relationship that is to exist between the two nodes against which the edge is matched. Our relationships are based on the topology of the ground network, and we thus consider the i -th-parent relation and its inverse. To represent the actual constraint, any node in the graph may involve a check for a particular assignment of the node it is matched against. We implemented SampleSearch with abstract learning (SS-AL) based on this definition of abstract constraints.

It should be clear that abstraction comes at a price. Abstract constraints are larger, because the graph representation requires internal nodes that would not appear in a propositional constraint which does not need to concern itself with relationships between variables. Moreover, checking an abstract constraint incurs an unavoidable overhead, since it requires matching a relational structure. To evaluate whether the positive or the negative aspects of abstraction prevail, we implemented SampleSearch with propositional learning (SS-L) as a point of reference.

5 Experiments

In our experiments, we compare the inference methods described above on several instantiations of BLN models (see Figures 1 and 2): The “grades” models are adaptations of the university model from [3]; the “meals habits” model captures the consumption and utensil usage habits of people during their daily meals; the “kitchen organization” model associates the spatial configuration of storage locations and devices in a kitchen with consumable objects and tools; and the “sequential activities” model is concerned with object placements and states as a result of activities of daily life being performed over time. For each model, we selected an inference task from our pool of tasks at random. In Figures 2a[2e], we plot the mean squared error in the posterior marginals of non-evidence variables against runtime, averaged across five trials. The algorithms used are: likelihood weighting (LW), SampleSearch with chronological backtracking (SS) and backjumping (SS-BJ), backward simulation (BS), backward SampleSearch (BSS, BSS-BJ), and learning-based search methods with backjumping (SS-L, SS-AL)³.

We observe that, in three of our scenarios, BSS methods are among the best, clearly outperforming their forward SS counterparts in two scenarios. In the two other scenarios, the roles are reversed. However, it is the learning-based methods that perform best in these scenarios. In “sequential activities”, this is due to the sequential nature of this particular model, which causes frequent backtracking that can be prevented by learning constraints. The abstract learning method SS-AL never surpasses its propositional counterpart SS-L. Apparently, the overhead incurred by abstraction cannot be amortized by the reduced number of constraints in the instances we tested. Because constraint learning in general constitutes additional overhead, it is a disadvantage in cases where it is essentially unnecessary. Conceptually simple methods such as LW and BS are unable to compute approximations for any of the challenging scenarios. The quality of the samples generated by the algorithms does not seem to vary widely, as approximation quality appears to be strongly correlated with the number of samples drawn (cf. Fig. 2f).

We also experimented with a Markov chain Monte Carlo method, MC-SAT [8], which is frequently the method of choice for Markov logic networks. In most runs, we were, however, unable to draw a single sample due to the size of the SAT problems that resulted from the large number of weighted formulas that is generated for each of the problem instances.

³ All methods were implemented in Java. For all search-based methods, asymptotically unbiased estimators were used.

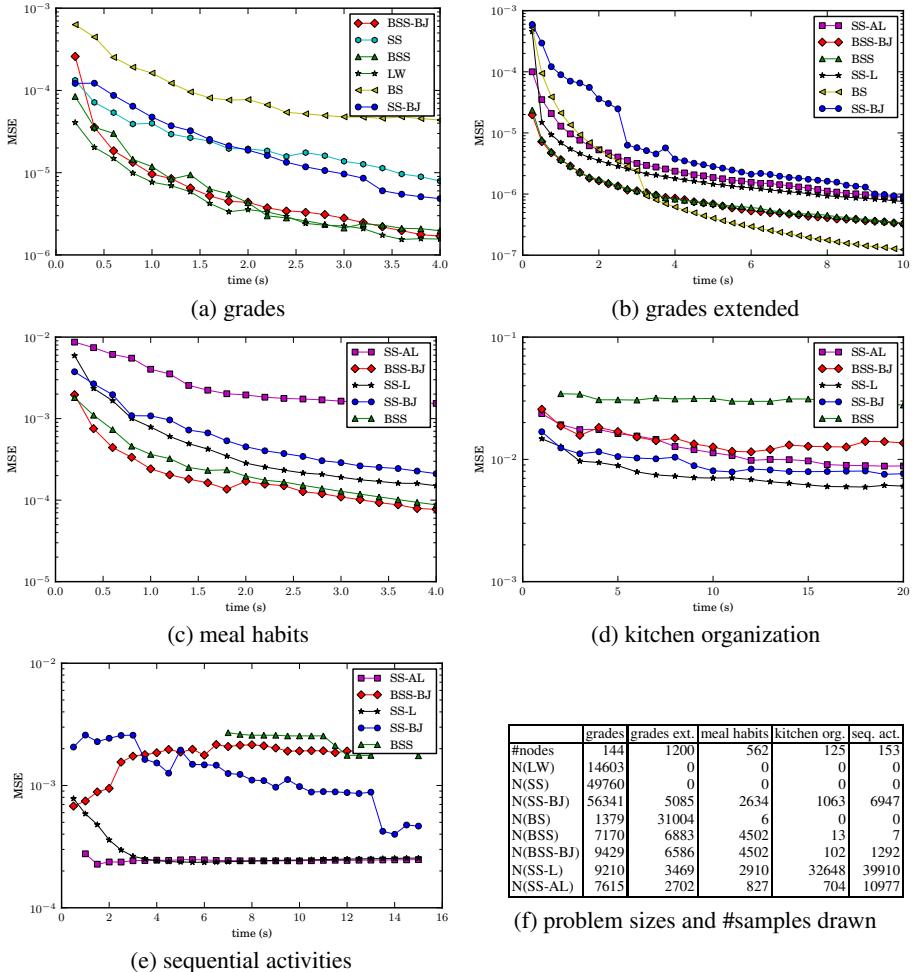


Fig. 2. Evaluation of the algorithms on instances of different BLNs. Mean squared errors in non-evidence variables are plotted in 2a-2e. The table 2f provides data on problem sizes and the number of samples drawn by the end of the time limit.

6 Conclusion

In this work, we presented Bayesian logic networks as a practical representation language for statistical relational knowledge that bases its semantics on mixed networks. Efficient inference being a key concern in practical applications, we investigated two importance sampling techniques that explicitly address the problem of dealing with a combination of deterministic and probabilistic knowledge as it is typically found in instances of BLNs. Our initial results are encouraging and do indicate that both backward simulation and constraint learning are appropriate options for the search for samples

(under the right circumstances). The recording of abstract constraints, however, incurs too much of an overhead, which cannot be compensated by gains in compactness – at least in the instances we tried. Experience in lifted satisfiability solving, however, suggests that abstraction can lead to significant speed-ups [15]. Therefore, directions for future work include finding better representations for abstract constraints that may alleviate the respective problems.

References

1. Richardson, M., Domingos, P.: Markov Logic Networks. *Mach. Learn.* 62, 107–136 (2006)
2. Jain, D., Kirchlechner, B., Beetz, M.: Extending Markov Logic to Model Probability Distributions in Relational Domains. In: Hertzberg, J., Beetz, M., Englert, R. (eds.) KI 2007. LNCS (LNAI), vol. 4667, pp. 129–143. Springer, Heidelberg (2007)
3. Getoor, L., Friedman, N., Koller, D., Pfeffer, A., Taskar, B.: Probabilistic Relational Models. In: Getoor, L., Taskar, B. (eds.) An Introduction to Statistical Relational Learning. MIT Press, Cambridge (2007)
4. Kersting, K., Raedt, L.D.: Bayesian Logic Programming: Theory and Tool. In: Getoor, L., Taskar, B. (eds.) Introduction to Statistical Relational Learning. MIT Press, Cambridge (2007)
5. Laskey, K.B.: MEBN: A Language for First-Order Bayesian Knowledge Bases. *Artif. Intell.* 172, 140–178 (2008)
6. Jaeger, M.: Model-Theoretic Expressivity Analysis. In: Raedt, L.D., Frasconi, P., Kersting, K., Muggleton, S. (eds.) Probabilistic Inductive Logic Programming. LNCS (LNAI), vol. 4911, pp. 325–339. Springer, Heidelberg (2008)
7. Mateescu, R., Dechter, R.: Mixed Deterministic and Probabilistic Networks. *Ann. Math. Artif. Intel.* (2008)
8. Poon, H., Domingos, P.: Sound and Efficient Inference with Probabilistic and Deterministic Dependencies. In: AAAI. AAAI Press, Menlo Park (2006)
9. Rubinstein, R.: Simulation and the Monte Carlo Method. John Wiley & Sons, Inc., Chichester (1981)
10. Fung, R.M., Favero, B.D.: Backward Simulation in Bayesian Networks. In: UAI, pp. 227–234 (1994)
11. Gogate, V., Dechter, R.: SampleSearch: A Scheme that Searches for Consistent Samples. In: AISTATS (2007)
12. Dechter, R., Frost, D.: Backjump-Based Backtracking for Constraint Satisfaction Problems. *Artif. Intell.* 136, 147–188 (2002)
13. Gogate, V., Domingos, P.: Formula-Based Probabilistic Inference. In: UAI (2010)
14. Kersting, K., Ahmadi, B., Natarajan, S.: Counting Belief Propagation. In: UAI (2009)
15. Parkes, A.J.: Lifted Search Engines for Satisfiability. PhD thesis (1999)

Transformation Rules for First-Order Probabilistic Conditional Logic Yielding Parametric Uniformity

Ruth Janning and Christoph Beierle

Fak. für Mathematik und Informatik, FernUniversität in Hagen, 58084 Hagen, Germany

Abstract. A major challenge in knowledge representation is to express uncertain knowledge. One possibility is to combine logic and probability. In this paper, we investigate the logic FO-PCL that uses first-order probabilistic conditionals to formulate uncertain knowledge. Reasoning in FO-PCL employs the principle of maximum entropy which in this context refers to the set of all ground instances of the conditionals in a knowledge base \mathcal{R} . We formalize the syntactic criterion of FO-PCL interactions in \mathcal{R} prohibiting the maximum entropy model computation on the level of conditionals instead of their instances. A set of rules is developed transforming \mathcal{R} into an equivalent knowledge base \mathcal{R}' without FO-PCL interactions.

1 Introduction

Many real-world problems require the representation of uncertain knowledge. One method to express uncertain knowledge is the combination of logic and probability as it is done for instance in the well-known Bayes and Markov nets. Probabilistic Conditional Logic (PCL) [13] is a logic which assigns to conditionals [110], i.e. if-then rules, a certain probability. Various extensions to a relational setting have been proposed (see [4] for an overview), among them Bayesian logic programs and Markov logic networks. In [3], a first-order extension of PCL called FO-PCL is developed. In FO-PCL, a rule like "*If it rains in X and X is located near to another place Y, then it rains in Y too with a probability of 0.8.*" can be formalized by the conditional $\langle(rains(Y)|rains(X) \wedge nearby(X, Y))[0.8], X \neq Y\rangle$ where $X \neq Y$ is a constraint prohibiting to instantiate X and Y with the same element (which would yield an immediate inconsistency). Reasoning in FO-PCL is done using the concept of maximum entropy which inductively completes the knowledge given by a set of conditionals in the most unbiased way [117]. However, the specification of the maximum entropy model of an FO-PCL knowledge base \mathcal{R} refers to the set of all ground instances of the conditionals in \mathcal{R} , yielding an unfeasible computation task for large universes [3]. Therefore, [3] introduces the notion of *parametric uniformity*. If \mathcal{R} is parametrically uniform, then the computation of the maximum entropy model only has to consider the conditionals in \mathcal{R} and not their instances, effectively reducing the complexity of the computation for \mathcal{R} to the complexity of the propositional case. Whereas [3] only indicates how certain causes prohibiting parametric uniformity of \mathcal{R} can be avoided, in this paper we formalize these causes by the syntactic criterion of FO-PCL interactions and develop a set of transformation rules \mathcal{T}_{PU} that transform any consistent knowledge base \mathcal{R} into an equivalent \mathcal{R}' that does not have FO-PCL interactions.

After briefly recalling the basic concepts of FO-PCL (Sec. 2), we develop a formal definition of interactions that lead to parametric non-uniformity (Sec. 3). In Sec. 4, transformation rules for interaction removing and simplification are defined, and their properties are investigated in Sec. 5. In Sec. 6, we conclude and point out future work.

2 Background: FO-PCL in a Nutshell

FO-PCL uses function-free signatures of the form $\Sigma = (S, D, Pred)$ where S is a set of sorts, $D = \bigcup_{s \in S} D^{(s)}$ is a finite set of (disjoint) sets of sorted constant symbols, and $Pred$ is a set of predicate symbols, each coming with an arity of the form $s_1 \times \dots \times s_n \in S^n$ indicating the required sorts for the arguments. Variables \mathcal{V} also have a unique sort, and all formulas and variable substitutions must obey the obvious sort restrictions. An FO-PCL *conditional* $R_x = \langle (\phi_{R_x} | \psi_{R_x})[\xi_{R_x}], C_{R_x} \rangle$ is composed of a *premise* ψ_{R_x} and a *conclusion* ϕ_{R_x} , which are quantifier and function free first-order formulas (over Σ and \mathcal{V}) without equality, a probability value $\xi_{R_x} \in [0, 1]$, and a *constraint* C_{R_x} which is a quantifier-free first-order formula using only the equality predicate. For $\neg(V = X)$ we also write $(V \neq X)$, and \top resp. \perp denote a tautology resp. a contradiction. An FO-PCL knowledge base \mathcal{R} consists of a set of FO-PCL conditionals.

When the constraint of a ground instance of R_x evaluates to *true*, it is called *admissible*, and $gnd(R_x)$ denotes the set of all admissible instances of R_x (over Σ). The Herbrand base $\mathcal{H}(\mathcal{R})$ is the set of all atoms in all $gnd(R_x)$ with $R_x \in \mathcal{R}$, and every subset of $\mathcal{H}(\mathcal{R})$ is a Herbrand interpretation, defining a logical semantics for \mathcal{R} . The probabilistic semantics of \mathcal{R} is a possible world semantics [5] where the ground atoms in $\mathcal{H}(\mathcal{R})$ are binary random variables. An FO-PCL *interpretation* $p_{X(\mathcal{R})}$ of \mathcal{R} is thus a joint probability function over $\mathcal{H}(\mathcal{R})$, and $p_{X(\mathcal{R})}$ is a *model* of \mathcal{R} if it satisfies every $R_x \in \mathcal{R}$, where $p_{X(\mathcal{R})}$ satisfies R_x iff for every admissible instance $\langle (\phi | \psi)[\xi_{R_x}], \top \rangle$ of R_x it holds that $p_{X(\mathcal{R})}(\phi \wedge \psi) = \xi_{R_x} \cdot p_{X(\mathcal{R})}(\psi)$.

A knowledge base $\mathcal{R} = \{R_1, \dots, R_m\}$ may have many different models, and the principle of maximum entropy [11] provides a method to select a model that is optimal in the sense that it is the most unbiased one. The computation of the uniquely determined maximum entropy model $p_{X(\mathcal{R})}^*$ is an optimization problem that can be represented by a Gibbs distribution

$$p_{X(\mathcal{R})}^*(x) = \frac{1}{Z} \exp \left(\sum_{k=1}^m \sum_{g_{R_k} \in gnd(R_k)} \lambda_{g_{R_k}} f_{g_{R_k}}(x) \right) \quad (1)$$

where $f_{g_{R_k}}$ is the feature function determined by g_{R_k} , $\lambda_{g_{R_k}}$ is a Lagrange multiplier and Z is a normalization constant (see [3] for more details). With this formula, one entropy-optimal parameter $\lambda_{g_{R_k}}$ for every ground instance g_{R_k} of a conditional R_k has to be determined. This calculation is computationally intensive respectively infeasible for larger sets of ground instances. However, there are FO-PCL knowledge bases for which the ground instances of a conditional share the same entropy-optimal parameter. Parameter sharing [3] means that for all conditionals all their ground instances share the same entropy-optimal parameter value. The advantage of parametric uniformity is

that just one entropy-optimal parameter $\lambda_{R_k}^*$ per conditional R_k has to be computed instead of one parameter per ground instance, yielding a usually computationally feasible problem:

$$p_{X(\mathcal{R})}^*(x) = \frac{1}{Z} \exp \left(\sum_{k=1}^m \lambda_{R_k}^* \sum_{g_{R_k} \in gnd(R_k)} f_{g_{R_k}}(x) \right) \quad (2)$$

Whereas parametric uniformity is a semantic notion, in [3] a syntactic criterion using so-called *involution*s sufficient to ensure it is presented. This syntactic criterion is based on the observation that parameter uniformity indicates identical knowledge about all ground instances of the same conditional for an FO-PCL knowledge base \mathcal{R} . Due to this, one should be able to transpose two ground instances g_{R_k}, g'_{R_k} of a conditional in \mathcal{R} without changing the joint probability function with maximum entropy. In this case the transposed ground instances must possess the same entropy optimal parameter, as the Gibbs distribution in (1) is determined by a unique set of Lagrange multipliers. A *probabilistic constraint involution* transposes instances of conditionals with an involution $\pi_{F(\mathcal{R})}$ and it transposes ground atoms with an involution $\pi_{X(\mathcal{R})}$ (see Example 3 or [3] for more details). An *involution covering* for \mathcal{R} is a set

$$\Pi := \left\{ (\pi_{F(\mathcal{R})}^{(1)}, \pi_{X(\mathcal{R})}^{(1)}), \dots, (\pi_{F(\mathcal{R})}^{(|\Pi|)}, \pi_{X(\mathcal{R})}^{(|\Pi|)}) \right\}$$

of pairs of $\pi_{F(\mathcal{R})}$ and $\pi_{X(\mathcal{R})}$, so that for any two instances $g_{R_k}, g'_{R_k} \in gnd(R_k)$ with $R_k \in \mathcal{R}$, there exists a sequence $\langle i_1, \dots, i_n \rangle$ of indices $i_j \in \{1, \dots, |\Pi|\}$, such that

$$\pi_{F(\mathcal{R})}^{(i_j)}(g_k^{i_{j-1}}) = g_k^{i_j}, 1 \leq j \leq n$$

holds, with $g_k^{i_j} \in gnd(R_k)$ for all $1 \leq j \leq n$, $g_k^{i_0} := g_{R_k}$ and $g_k^{i_n} := g'_{R_k}$ (Def. 7.4.2 in [3]). Corollary 7.4.4 in [3] then states:

Theorem 1 (Involution covering implies parametric uniformity). *If there is an involution covering for \mathcal{R} , then \mathcal{R} is parametrically uniform.*

While in general, FO-PCL constraints may contain conjunctions and disjunctions, for the rest of the paper we will consider only FO-PCL conditionals with a constraint formula which is a conjunction of equations and inequations. This is not a principle restriction, since conditionals of the form $\langle (\phi_{R_x} | \psi_{R_x})[\xi_{R_x}], C_1 \vee C_2 \rangle$ may be replaced by the two conditionals $\langle (\phi_{R_x} | \psi_{R_x})[\xi_{R_x}], C_1 \rangle$ and $\langle (\phi_{R_x} | \psi_{R_x})[\xi_{R_x}], C_2 \rangle$.

3 Parametric Non-uniformity and FO-PCL Interactions

There are various causes for parametric non-uniformity: interactions between two different conditionals (inter-rule interactions), and interactions within a single conditional (intra-rule interactions); such interactions prohibit involution covering.

Example 1. Let $R_1 = \langle (P(a))[\xi_1], \top \rangle$ and $R_2 = \langle (P(U)|Q(V))[\xi_2], U \neq V \rangle$ and let $\Sigma = \{\{s\}, \{D^{(s)} = \{a, b, c\}\}, \{P/\langle s \rangle, Q/\langle s \rangle\}\}$ be the corresponding signature. The ground instances of R_2 are:

$$\begin{array}{ll} R_{2-1} = \langle (P(a)|Q(b))[\xi_2], \top \rangle & R_{2-4} = \langle (P(b)|Q(c))[\xi_2], \top \rangle \\ R_{2-2} = \langle (P(a)|Q(c))[\xi_2], \top \rangle & R_{2-5} = \langle (P(c)|Q(a))[\xi_2], \top \rangle \\ R_{2-3} = \langle (P(b)|Q(a))[\xi_2], \top \rangle & R_{2-6} = \langle (P(c)|Q(b))[\xi_2], \top \rangle \end{array}$$

Making an attempt to find a probabilistic constraint involution, which transposes R_{2-1} or R_{2-2} with one of R_{2-3}, \dots, R_{2-6} , does not yield an admissible probabilistic constraint involution, because R_1 possesses no ground instances with the atoms $P(b)$ or $P(c)$ to transpose it with ground instances with $P(a)$. So R_1 and R_2 cause this *inter-rule interaction*. \square

Example 2. Let $R_3 = \langle (P(U)|Q(V))[\xi_3], U \neq V \wedge U \neq a \rangle$ and let Σ as in Ex. 1 be the corresponding signature. The ground instances of R_3 are:

$$\begin{array}{ll} R_{3-1} = \langle (P(b)|Q(a))[\xi_3], \top \rangle & R_{3-3} = \langle (P(c)|Q(a))[\xi_2], \top \rangle \\ R_{3-2} = \langle (P(b)|Q(c))[\xi_2], \top \rangle & R_{3-4} = \langle (P(c)|Q(b))[\xi_2], \top \rangle \end{array}$$

As one can see, $Q(a)$ occurs twice but $Q(b)$ and $Q(c)$ each once only. So there is no probabilistic constraint involution which transposes $Q(a)$, as for the second $Q(a)$ there is no second $Q(b)$ or $Q(c)$ to transpose. So R_3 causes this *intra-rule interactions*. \square

In general, the reasons for inter-rule or intra-rule interactions are the sharing of ground atoms (see Ex. 1) or an imbalance in the frequency of occurrence of ground atoms (see Ex. 2) after substituting the variables by constants; both types of interactions prohibit probabilistic constraint involutions and cause parametric non-uniformity. While 3 only gives an informal illustration of these phenomena, we will now formally define them.

Definition 1 (FO-PCL Interactions). Let $R_x = \langle (\phi_{R_x}|\psi_{R_x})[\xi_{R_x}], C_{R_x} \rangle$ and $R_y = \langle (\phi_{R_y}|\psi_{R_y})[\xi_{R_y}], C_{R_y} \rangle$, and let U, V, X be variables, P a predicate symbol and a a constant symbol.

1. If $R_x = \langle (\dots P(\dots, x_{n-1}, Arg_n, x_{n+1}, \dots) \dots)[\xi_{R_x}], C_{R_x} \rangle$ and $R_y = \langle (\dots P(\dots, x_{n-1}, V, x_{n+1}, \dots) \dots)[\xi_{R_y}], C_{R_y} \rangle$ with $C_{R_y} \not\models (V \neq a)$ then there is an inter-rule interaction between R_x and R_y iff $Arg_n = a$ or ($Arg_n = X$ and $(C_{R_x} \models (X = a) \text{ or } C_{R_x} \models (X \neq a))$).
2. If there are different atoms $A, B \in atoms(\phi_{R_x}) \cup atoms(\psi_{R_x})$ with $U \in vars(A)$, $U \notin vars(B)$, $V \in vars(B)$, then there is an intra-rule interaction within R_x iff $C_{R_x} \models (U \neq a) \wedge (U \neq V)$ and $C_{R_x} \not\models (V \neq a)$.

An involution covering for \mathcal{R} and corresponding probabilistic constraint involutions exist if each of the conditionals in \mathcal{R} contains identical knowledge about its ground instances so that one is able to transpose these ground instances and still obtain the same joint probability function with maximum entropy (cf. 3). The occurrence of a constant symbol a in an FO-PCL conditional R_x indicates exceptional knowledge about a real-world object, prohibiting constraint involutions and an involution covering:

1. The occurrence of a in an atom with the predicate symbol P in the premise or conclusion of R_x prohibits probabilistic constraint involutions if there is a second conditional R_y with a variable V in the same position n in P and the constraint

- formula of R_y does not imply $(V \neq a)$. In this case the ground instances of R_x and R_y share ground atoms. If we now transpose the ground atoms of R_y , we are not able to transpose the corresponding ground atoms of R_x , as the ground instances of R_x lack some ground atoms because they just contain ground atoms with a in position n in P (cf. Ex. 1). A dual observation applies if a is forbidden in the n -th argument of P in R_x , but may or may not occur in that position in R_y . These cases correspond to the conditions of inter-rule interactions in Def. 1.
2. The occurrence of a in the constraint formula of R_x prohibits probabilistic constraint involutions, if the constraint formula of R_x implies $(U \neq a) \wedge (U \neq V)$ and does not imply $(V \neq a)$ for variables U, V occurring in different atoms. In this case there is an imbalance in the frequency of occurrence of ground atoms with the same predicate symbol in the ground instances of R_x . Because of this imbalance, for the ground atoms with a there are not enough other ground atoms with other constant symbols in the place of a to get a transposition (cf. Example 2). This case corresponds to the conditions of intra-rule interactions in Def. 1.

Thus, the formal notion of FO-PCL interaction gives a syntactic criterion required for parametric uniformity since any FO-PCL interaction in \mathcal{R} implies that \mathcal{R} can not be parametrically uniform.

4 FO-PCL Transformation Rules

In order to eliminate causes for parametric non-uniformity in an FO-PCL knowledge base \mathcal{R} , we will detect conditionals causing an FO-PCL interaction and replace them by other conditionals avoiding that interaction, while not changing the entropy-optimal model of \mathcal{R} (see also [6]).

Definition 2 (Non-substitution). Let $\sigma = \{V/a\}$ be a substitution with a domain containing exactly one variable and let $R_i = \langle (\phi_{R_i} | \psi_{R_i})[\xi_{R_i}], C_{R_i} \rangle$ be a conditional with $V \in \text{vars}(R_i)$. Then $\bar{\sigma}(R_i)$ is called non-substitution application and denotes the conditional $\langle (\phi_{R_i} | \psi_{R_i})[\xi_{R_i}], C_{R_i} \wedge V \neq a \rangle$.

Thus, $\bar{\sigma}$ maps the FO-PCL conditional R_i onto the FO-PCL conditional $\bar{\sigma}(R_i)$ obtained from R_i by adding a constraint that prohibits the application of σ .

For the rest of this paper, we will assume that \mathcal{R} is an FO-PCL knowledge base, U, V, W are variables, a, b, c, d are constant symbols, and that for any index i , $R_i = \langle (\phi_{R_i} | \psi_{R_i})[\xi_{R_i}], C_{R_i} \rangle$ is an FO-PCL conditional.

4.1 Transformation Rules for Interaction Removing

In the following, we present the transformation rules PC_{inter} for FO-PCL interactions between two different conditionals and S_{intra} for such within a single conditional.

Definition 3 (PC_{inter} – Inter-rule substitution). The inter-rule substitution PC_{inter} is the following transformation rule:

Conditions:

$$R_1 = \langle (\phi_{R_1} | \psi_{R_1})[\xi_{R_1}], C_{R_1} \rangle = \langle (\dots P(\dots, x_{n-1}, x_n, x_{n+1}, \dots) \dots)[\xi_{R_1}], C_{R_1} \rangle \in \mathcal{R}$$

$$R_2 = \langle (\phi_{R_2} | \psi_{R_2})[\xi_{R_2}], C_{R_2} \rangle = \langle (\dots P(\dots, x_{n-1}, V, x_{n+1}, \dots) \dots)[\xi_{R_2}], C_{R_2} \rangle$$

$$x_n = a \text{ or } (x_n = U \text{ and } C_{R_1} \models (U \neq a))$$

$$C_{R_2} \not\models (V \neq a)$$

Transformation: \Box

$$\frac{\mathcal{R} \cup \{R_2\}}{\mathcal{R} \cup \{\sigma(R_2), \bar{\sigma}(R_2)\}}, \sigma = \{V/a\}$$

The conditions of PC_{inter} state that there must be an FO-PCL conditional R_1 with a constant symbol a (or a variable U which may not be substituted by a) as an argument in one of its atoms in the premise or in the conclusion. Additionally, there must be another conditional R_2 , which contains in its premise or conclusion an atom with the same predicate symbol but with a variable V in the position of a (or U , respectively). If further $V \neq a$ is not implied by the constraint formula of R_2 , then R_2 is replaced by $\sigma(R_2)$ and $\bar{\sigma}(R_2)$ with $\sigma = \{V/a\}$.

Definition 4 (S_{intra} – Intra-rule substitution). The intra-rule substitution S_{intra} is the following transformation rule:

Conditions:

$$R_1 = \langle (\phi_{R_1} | \psi_{R_1})[\xi_{R_1}], C_{R_1} \rangle, C_{R_1} \models (U \neq a) \wedge (U \neq V), C_{R_1} \not\models (V \neq a)$$

$$A, B \in atoms(\phi_{R_1}) \cup atoms(\psi_{R_1}), A \neq B, U \in vars(A), U \notin vars(B), V \in vars(B)$$

Transformation:

$$\frac{\mathcal{R} \cup \{R_1\}}{\mathcal{R} \cup \{\sigma(R_1), \bar{\sigma}(R_1)\}}, \sigma = \{V/a\}$$

The conditions of S_{intra} say that there must be an FO-PCL conditional R_1 whose constraint formula implies $(U \neq a) \wedge (U \neq V)$ and does not imply $V \neq a$. U must occur in an atom A , and V must occur in an another atom B which does not contain U , where A and B occur in the premise or the conclusion of R_1 .

Example 3 (Application of PC_{inter} and S_{intra}). Let $R_1 = \langle (P(a))[\xi_1], \top \rangle$ and $R_2 = \langle (P(U)|Q(V))[\xi_2], U \neq V \rangle$ be the FO-PCL conditionals from Example \Box. Due to the inter-rule interaction between R_1 and R_2 , using $\sigma = \{U/a\}$, PC_{inter} replaces R_2 by:

$$\sigma(R_2) = \langle (P(a)|Q(V))[\xi_2], a \neq V \rangle$$

$$\bar{\sigma}(R_2) = \langle (P(U)|Q(V))[\xi_2], U \neq V \wedge U \neq a \rangle$$

The ground instances of $\sigma(R_2)$ are:

$$\langle (P(a)|Q(b))[\xi_2], \top \rangle = R_{2-1}$$

$$\langle (P(a)|Q(c))[\xi_2], \top \rangle = R_{2-2}$$

The ground instances of $\bar{\sigma}(R_2)$ are:

$$\langle (P(b)|Q(a))[\xi_2], \top \rangle = R_{2-3}$$

$$\langle (P(b)|Q(c))[\xi_2], \top \rangle = R_{2-4}$$

$$\langle (P(c)|Q(a))[\xi_2], \top \rangle = R_{2-5}$$

$$\langle (P(c)|Q(b))[\xi_2], \top \rangle = R_{2-6}$$

There are still intra-rule interactions within $\bar{\sigma}(R_2)$. Hence, using $\tau = \{V/a\}$, S_{intra} replaces $\bar{\sigma}(R_2)$ by

¹ As usual, when using a set union like $S_1 \cup S_2$ in the premise of a transformation rule, we always assume $S_1 \cap S_2 = \emptyset$.

$$\tau(\bar{\sigma}(R_2)) = \langle (P(U)|Q(a))[\xi_2], U \neq a \rangle \quad \text{with the ground instances} \\ \langle (P(b)|Q(a))[\xi_2], \top \rangle = R_{2-3} \\ \langle (P(c)|Q(a))[\xi_2], \top \rangle = R_{2-5}$$

and

$$\bar{\tau}(\bar{\sigma}(R_2)) = \langle (P(U)|Q(V))[\xi_2], \\ U \neq V \wedge U \neq a \wedge V \neq a \rangle \quad \text{with the ground instances} \\ \langle (P(b)|Q(c))[\xi_2], \top \rangle = R_{2-4} \\ \langle (P(c)|Q(b))[\xi_2], \top \rangle = R_{2-6}$$

Now there exists a probabilistic constraint involution π and an involution covering Π :

$$\pi_{F(R)}^{(1)} := (R_{2-3} \ R_{2-5}) \quad \pi_{X(R)}^{(1)} := (P(b) \ P(c)) \\ (R_{2-4} \ R_{2-6}) \quad (Q(c) \ Q(b)) \quad \Pi := \{(\pi_{F(R)}^{(1)}, \pi_{X(R)}^{(1)})\}$$

This implies $\lambda_{R_{2-3}}^* = \lambda_{R_{2-5}}^*$ and $\lambda_{R_{2-4}}^* = \lambda_{R_{2-6}}^*$. \square

An equation ($V = a$) implied by the constraint formula of an FO-PCL conditional R_w yields the substitution $\sigma = \{V/a\}$ as the only possible substitution for V , but PC_{inter} may not recognise this.

Example 4. Let $R_1 = \langle (P(X))[\xi_1], X = a \rangle$ and $R_2 = \langle (P(U)|Q(V))[\xi_2], U \neq V \rangle$. PC_{inter} is not applicable although there is an inter-rule interaction between R_1 and R_2 .

Therefore, we need a transformation applying a substitution of variables.

Definition 5 (V_{subs} – Substitution of variables in equations). *The substitution of variables in equations V_{subs} is the following transformation rule:*

$$\begin{array}{ll} \text{Condition:} & C_{R_w} \models (V = a) \\ \text{Transformation:} & \frac{\mathcal{R} \cup \{R_w\}}{\mathcal{R} \cup \{\sigma(R_w)\}}, \sigma = \{V/a\}. \end{array}$$

4.2 Simplification Rules

Conditionals R_x and R_y whose ground instances are satisfied by exactly the same probability distributions are called *FO-PCL equivalent*, denoted by $C_{R_x} \equiv_{FO-PCL} C_{R_y}$. Thus, if \mathcal{R} contains two FO-PCL equivalent conditionals, removing one of them from \mathcal{R} does not change the entropy-maximal model. We could therefore define a simplification rule based on factorization by replacing $\mathcal{R} \cup \{R_x, R_y\}$ by $\mathcal{R} \cup \{R_x\}$ under some condition ensuring that $R_x \equiv_{FO-PCL} R_y$. Instead, we will use a more specific approach by preferring conditionals with non-negated conclusions and with weak premises and weak constraints. The following transformation rule replaces a conditional with a negated formula $\neg\phi$ as conclusion by an FO-PCL equivalent conditional with conclusion ϕ .

Definition 6 (H_{pos} – Prefer positive heads)

$$\text{Transformation:} \quad \frac{\mathcal{R} \cup \{(\neg\phi|\psi)[\xi], C\}}{\mathcal{R} \cup \{(\phi|\psi)[1-\xi], C\}}$$

If there are two conditionals such that the premise of the first one is a logical consequence of the second one and the other parts of the two conditionals are identical, the second conditional can be removed since it does not carry any additional information not already contained in the first one. A similar argumentation applies to the constraint parts of the conditionals. For the latter, we have to take into account that different constants denote different elements. For any constraint C , let $\rho_{rc}(C)$ be the constraint obtained from C by replacing any $a \neq b$ and $a = a$ by \top and any $a \neq a$ and $a = b$ by \perp , where a, b are different constants.

Definition 7 (W_{weak} – Prefer weak premises and constraints)

Conditions: $R_1 = \langle (\phi_{R_1} | \psi_{R_1})[\xi_{R_1}], C_{R_1} \rangle$
 $R_2 = \langle (\phi_{R_2} | \psi_{R_2})[\xi_{R_2}], C_{R_2} \rangle$
 $\xi_{R_1} = \xi_{R_2}$ and there is a variable renaming σ such that
 $\sigma(\phi_{R_2}) \equiv \phi_{R_1}, \sigma(\psi_{R_2}) \models \psi_{R_1}, \sigma(\rho_{rc}(C_{R_2})) \models \rho_{rc}(C_{R_1})$

$$\text{Transformation: } \frac{\mathcal{R} \cup \{R_1, R_2\}}{\mathcal{R} \cup \{R_1\}}$$

Furthermore, FO-PCL conditionals with an inconsistent constraint formula yield no admissible ground instances and can be removed from \mathcal{R} .

Definition 8 (F_{remove} – Contradiction removing)

Condition: $C_{R_v} \equiv \perp$

$$\text{Transformation: } \frac{\mathcal{R} \cup \{R_v\}}{\mathcal{R}}$$

5 Properties of FO-PCL Transformation Rules

Applying the set of obtained FO-PCL transformation rules $\mathcal{T}_{PU} = \{PC_{inter}, S_{intra}, V_{subs}, H_{pos}, W_{weak}, F_{remove}\}$ exhaustively to an FO-PCL knowledge base \mathcal{R} , yields an algorithm transforming \mathcal{R} into an FO-PCL knowledge base \mathcal{R}' such that \mathcal{R}' has the same entropy-maximal model as \mathcal{R} , but does not contain any FO-PCL interactions as given in Def. 1. In the following, we will first show that the transformation rules are correct and that the algorithm always terminates. Afterwards, we will address its completeness and confluence properties.

5.1 Correctness

Correctness means that before and after the application of each of the transformation rules we get the same maximum-entropy model.

Theorem 2 (Correctness of PC_{inter}). Let \mathcal{R}, R_1, R_2 and σ be as in Def. 3. Then $\mathcal{R}' := \mathcal{R} \cup \{\sigma(R_2), \bar{\sigma}(R_2)\}$ possesses the same maximum-entropy model as $\mathcal{R} \cup \{R_2\}$.

Proof. The sets of ground instances of $\sigma(R_2)$ and $\bar{\sigma}(R_2)$ are disjoint because in $\sigma(R_2)$ all occurrences of V are substituted by a , and in $\bar{\sigma}(R_2)$ V must not be substituted by a . However, the union of these sets is equal to the set of ground instances of R_2 since in R_2 , V can be substituted both by a and by the other constants which can be applied in the position of V . So ground instances of R_2 with a in the position of V correspond to the

ground instances of $\sigma(R_2)$, and ground instances of R_2 with other possible constants in the position of V correspond to the ground instances of $\bar{\sigma}(R_2)$. Thus, \mathcal{R}' with $\sigma(R_2)$ and $\bar{\sigma}(R_2)$ instead of R_2 possesses the same maximum-entropy model as \mathcal{R} with R_2 , because the set of ground instances of conditionals is equal for both. \square

Using an analogous argumentation, we can also proof the following theorem.

Theorem 3 (Correctness of S_{intra}). *Let \mathcal{R} , R_1 , and σ be as in Def. 4. Then $\mathcal{R}' := \mathcal{R} \cup \{\sigma(R_1), \bar{\sigma}(R_1)\}$ possesses the same maximum-entropy model as $\mathcal{R} \cup \{R_1\}$.*

Theorem 4 (Correctness of V_{subs}). *Let \mathcal{R} , R_w , and σ be as in Def. 5. Then $\mathcal{R}' := \mathcal{R} \cup \{\sigma(R_w)\}$ possesses the same maximum-entropy model as $\mathcal{R} \cup \{R_w\}$.*

Proof. V_{subs} substitutes the variable V by the constant a in R_w if the constraint formula of R_w implies $V = a$. This has no influence to the maximum-entropy model, as it does not change the set of admissible ground instances of R_w . \square

Theorem 5 (Correctness of simplification rules). *Applying any of the three simplification rules H_{pos} , W_{weak} or F_{remove} to an FO-PCL knowledge base \mathcal{R} yields a knowledge base \mathcal{R}' possessing the same maximum-entropy model as \mathcal{R} .*

Proof. F_{remove} removes conditionals with inconsistent constraint formulas. Such formulas do not lead to admissible ground instances, hence such conditionals do not affect the maximum-entropy model. H_{pos} is correct since $\langle(\neg\phi|\psi)[\xi], C\rangle \equiv_{FO\text{-PCL}} \langle(\phi|\psi)[1 - \xi], C\rangle$. W_{weak} removes an FO-PCL conditional R_2 if the considered knowledge base contains another conditional R_1 with the same probability value and there is a variable renaming so that under this renaming, the conclusions are equivalent, the premise of R_2 implies the premise of R_1 and the constraint formula of R_2 implies the constraint formula of R_1 , taking into account that different constants are not equal. Thus, every probability distribution satisfying all admissible ground instances of R_2 also satisfies all ground instances of R_1 , implying that removing R_2 from \mathcal{R} does not change the maximum-entropy model. \square

Putting the previous theorems together, we get:

Theorem 6 (Correctness of T_{PU}). *Applying T_{PU} to an FO-PCL knowledge base \mathcal{R} yields a knowledge base \mathcal{R}' possessing the same maximum-entropy model as \mathcal{R} .*

5.2 Termination

For proving termination we have to show that the transformation rules in T_{PU} are not infinitely often applicable to an FO-PCL knowledge base \mathcal{R} . We first observe that PC_{inter} and S_{intra} both reduce the number of admissible instantiations, V_{subs} reduces the number of variables in a constraint, H_{pos} reduces the number of negation symbols in the conclusion, and W_{weak} and F_{remove} reduce the number of conditionals. We can exploit these observations by applying techniques used for proving termination of rewriting systems [2]. In the following, we will sketch a possible formalization of the proof, while assuming some familiarity with these techniques, in particular with well-founded sets, multisets, and multiset orderings (cf. [2]).

For each conditional R , let $\delta(R) = (k, m, n)$ where k is the number of negation symbols in the conclusion of R , m the number of variables in the constraint of R , and n the number of admissible ground instantiations of R . Let \leq^3 be the ordering relation on \mathbb{N}^3 given by $(k, m, n) \leq^3 (k', m', n')$ iff $k \leq k'$, $m \leq m'$ and $n \leq m'$, and let $<^3$ be the irreflexive component of \leq^3 . Then $(\mathbb{N}^3, <^3)$ is a well-founded set, and the induced multiset $(\mathcal{M}(\mathbb{N}^3), \ll^3)$ of tripels from \mathbb{N}^3 with the induced multiset ordering \ll^3 is well-founded as well (see e.g. [2] for details). Let $\Delta(\mathcal{R})$ denote the multiset of tripels $\delta(R)$ for all $R \in \mathcal{R}$. We can then show that for any rule application from T_{PU} transforming \mathcal{R} into \mathcal{R}' it holds that $\Delta(\mathcal{R}') \ll^3 \Delta(\mathcal{R})$, yielding the following theorem.

Theorem 7 (Termination of T_{PU}). *Exhaustively applying T_{PU} to an FO-PCL knowledge base \mathcal{R} always terminates.*

5.3 Completeness

In Def. 1 we formalized FO-PCL interactions that prohibit parametric uniformity of \mathcal{R} . T_{PU} completely removes these interactions.

Theorem 8 (T_{PU} not applicable implies no interactions). *If none of the transformation rules in T_{PU} is applicable to any conditional in \mathcal{R} then there are no FO-PCL interactions in \mathcal{R} .*

Proof. Assume that there is an FO-PCL interaction in \mathcal{R} as defined in Def. 1. In the first case (inter-rule interactions) PC_{inter} or V_{subs} is applicable as either the conditions of PC_{inter} (cf. Def. 3) or the conditions of V_{subs} (cf. Def. 5) are satisfied. The conditions of the second case (intra-rule interactions) are the same as the conditions of S_{intra} (cf. Def. 4); hence in this case S_{intra} is applicable. Thus, at least one of the three rules PC_{inter} , S_{intra} , or V_{subs} is applicable to \mathcal{R} . \square

While a full investigation of the general case is still to be done, for all FO-PCL knowledge bases \mathcal{R} studied in [3] and [6], the non-existence of FO-PCL interactions in \mathcal{R}' obtained from \mathcal{R} by exhaustively applying T_{PU} ensured the existence of an involution covering and thus, due to Theorem 1 parametric uniformity of \mathcal{R}' .

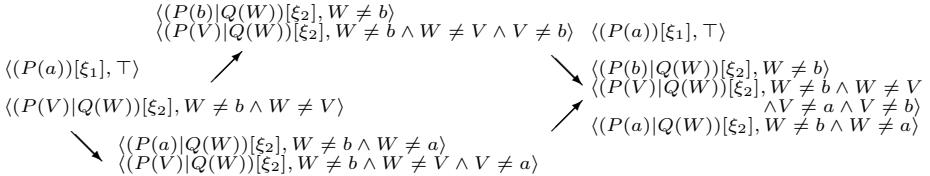
5.4 Confluence

Confluence is often a very desirable property for a transformation system (cf. [2]). While we do not yet have a full result regarding the behaviour of T_{PU} with respect to confluence, the following examples illustrate confluence properties of T_{PU} in important situations. The first one arises when two different transformation rules of T_{PU} are applicable to an FO-PCL conditional R_x , cf. Ex. 5. The second situation arises when one transformation rule is applicable to two different positions in R_x , cf. Ex. 6.

Example 5 (Two different transformation rules are applicable). Because of the inter-rule interactions between

$$R_1 = \langle (P(a))[\xi_1], \top \rangle \text{ and } R_2 = \langle (P(V)|Q(W))[\xi_2], W \neq b \wedge W \neq V \rangle$$

R_2 has to be replaced. However, there is also an intra-rule interaction within R_2 . Hence, R_2 can be replaced first by S_{intra} or by PC_{inter} :

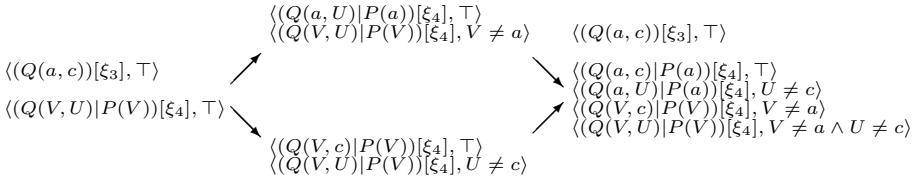


In the case of Example 5 it does not matter in which order the two rules are applied, as the interaction which is not treated first persists in the second conditional of the new conditionals and is treated subsequently.

Example 6 (One transformation rule is applicable to two different positions). There are inter-rule interactions between

$$R_3 = \langle (Q(a, c))[\xi_3], \top \rangle \text{ and } R_4 = \langle (Q(V, U)|P(V))[\xi_4], \top \rangle$$

both because of the occurrence of a and because of the occurrence of c . Hence, PC_{inter} can replace R_2 in two different ways:



Also in the case of Ex. 6, it does not matter in which order the rules are applied. One transformation rule just considers the FO-PCL interaction in one position in a conditional and does not affect FO-PCL interactions in other positions. If one FO-PCL interaction is treated first, the other FO-PCL interaction persists in the new conditionals and is treated subsequently, leading to the same result in both application orders.

6 Conclusions and Further Work

For FO-PCL knowledge bases with a large set of constants in the universe, the maximum entropy model computation is infeasible without parametric uniformity. The set of transformation rules T_{PU} presented in this paper, for which we showed correctness and termination, transforms any FO-PCL knowledge base \mathcal{R} into an FO-PCL knowledge base \mathcal{R}' without FO-PCL interactions, which are reasons for non-uniformity, but with the same maximum entropy model. Paramtric uniformity of a knowledge base allows lifted inference instead of inference on the level of instances.

Besides [3] where FO-PCL was introduced, there are some other approaches to probabilistic relational modeling also using the principle of maximum entropy, see [9][8][14]. In these approaches, a semantics that is also instantiation-based and alternative semantics that do not require instantiations are investigated in detail, but none of them uses constraints as in FO-PCL to restrict the number of admissible instantiations. The notion of *prototypical indifference* in [8] is closely related to the concept of involutions used here. Another approach to inference from probabilistic knowledge is the inference in parameterized belief networks as described in [12]. In this work, methods similar to

our approach are applied. The knowledge is modeled by *probabilistic assertions*, which correspond to FO-PCL conditionals, and the *splitting operation* is similar to our use of σ in combination with $\bar{\sigma}$. However, [12] is based on Bayesian networks and does not use the principle of maximum entropy.

Our current work on FO-PCL and the transformation of FO-PCL knowledge bases includes continuing the investigation of the formal properties of T_{PU} , the treatment of disjunctive constraint formulas, and the implementation of T_{PU} in a software system in order to facilitate experiments and further evaluations.

Acknowledgements. We thank Annika Krämer for her helpful suggestions and comments on a previous version of this paper. The research reported here was partially supported by the DFG (grant BE 1700/7-2).

References

1. Adams, E.W.: *The Logic of Conditionals*. D. Reidel Publishing Company, Dordrecht-Holland (1975)
2. Baader, F., Nipkow, T.: *Term Rewriting and All That*. Cambridge University Press, Berlin (1999)
3. Fisseler, J.: Learning and Modeling with Probabilistic Conditional Logic. *Dissertations in Artificial Intelligence*, vol. 328. IOS Press, Amsterdam (2010)
4. Getoor, L., Taskar, B. (eds.): *Introduction to Statistical Relational Learning*. MIT Press, Cambridge (2007)
5. Halpern, J.Y.: *Reasoning About Uncertainty*. MIT Press, Cambridge (2005)
6. Janning, R.: Transforming first-order probabilistic conditional logic knowledge bases to facilitate the maximum entropy model computation. Master's thesis, FernUniversität in Hagen (to appear, 2011)
7. Kern-Isberner, G.: *Conditionals in Nonmonotonic Reasoning and Belief Revision*. LNCS (LNAI), vol. 2087. Springer, Heidelberg (2001)
8. Kern-Isberner, G., Thimm, M.: Novel Semantical Approaches to Relational Probabilistic Conditionals. In: *Proceedings of the Twelfth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2010)*, pp. 382–392 (May 2010)
9. Loh, S., Thimm, M., Kern-Isberner, G.: On the Problem of Grounding a Relational Probabilistic Conditional Knowledge Base. In: *Proceedings of the 14th International Workshop on Non-Monotonic Reasoning (NMR 2010)*, Toronto, Canada (May 2010)
10. Nute, D., Cross, C.B.: Conditional logic. In: Gabbay, D.M., Guenther, F. (eds.) *Handbook of Philosophical Logic*, vol. 4, pp. 1–98. Kluwer Academic Publishers, Dordrecht (2002)
11. Paris, J.B.: *The uncertain reasoner's companion - A mathematical perspective*. Cambridge University Press, Cambridge (1994)
12. Poole, D.: First-order probabilistic inference. In: Gottlob, G., Walsh, T. (eds.) *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI 2003)*, pp. 985–991. Morgan Kaufmann, San Francisco (2003)
13. Rödder, W., Kern-Isberner, G.: Representation and extraction of information by probabilistic logic. *Information Systems* 21(8), 637–652 (1996)
14. Thimm, M., Kern-Isberner, G., Fisseler, J.: Relational probabilistic conditional reasoning at maximum entropy. In: Liu, W. (ed.) *ECSQARU 2011*. LNCS, vol. 6717, pp. 447–458. Springer, Heidelberg (2011)

Variance Scaling for EDAs Revisited

Oliver Kramer and Fabian Gieseke

Department Informatik
Carl von Ossietzky Universität Oldenburg
26111 Oldenburg

Abstract. Estimation of distribution algorithms (EDAs) are derivative-free optimization approaches based on the successive estimation of the probability density function of the best solutions, and their subsequent sampling. It turns out that the success of EDAs in numerical optimization strongly depends on scaling of the variance. The contribution of this paper is a comparison of various adaptive and self-adaptive variance scaling techniques for a Gaussian EDA. The analysis includes: (1) the Gaussian EDA without scaling, but different selection pressures and population sizes, (2) the variance adaptation technique known as Silverman's rule-of-thumb, (3) σ -self-adaptation known from evolution strategies, and (4) transformation of the solution space by estimation of the Hessian. We discuss the results for the sphere function, and its constrained counterpart.

1 Introduction

EDAs are a class of evolutionary optimizers that are closely related to machine learning techniques as they are based on the estimation of probability density distributions during the search process. They are population-based approaches iteratively performing the following steps:

1. Estimation of the distribution $\hat{p}(\sigma)$ of the best solutions,
2. scaling the variance σ , and
3. sampling from the distribution $\hat{p}(\sigma')$

until a termination criterion is met. As distribution estimator various techniques can be employed, but in numerical optimization most frequently Gaussian distributions are assumed. As it can easily be shown that the solely scheme of estimation and sampling without variance scaling can result in premature convergence (convergence to a fitness value that is not the fitness of the optimum), variance scaling techniques become important. A comprehensive comparison of different variance scaling techniques, in particular taking into account Silverman's rule-of-thumb, σ -self-adaptation, and an estimation of the Hessian is missing in literature, and will be presented in this work for optimization in \mathbb{R}^N .

Section 2 shortly repeats the basic principles of EDAs with focus on the algorithmic framework our Gaussian EDA will be based on. In Section 3.1 we present the experimental analysis concentrating on various population sizes and selection pressures of the EDA without variance scaling. In Section 3.2 we compare to

Silverman's rule-of-thumb known from kernel density estimation. In Section 3.3 we experimentally analyze a variant with σ -self-adaptation, while Section 3.4 introduces a Hessian approach. Conclusions are presented in Section 4.

2 Estimation of Distribution Algorithms

The idea of EDAs is the successive repetition of estimation of the distribution \hat{p} of the best solutions during the search, and sampling from this distribution \hat{p} in the following generation. The distribution usually depends on one or more parameters σ , e.g., the covariance σ of the univariate Gaussian distribution $\mathcal{N}(0, \sigma)$.

Figure 1 illustrates the steps our EDA is based on. After uniform initialization of λ solutions $\mathbf{x}_1, \dots, \mathbf{x}_\lambda$ in the search space, the μ -best according to the fitness evaluation $f(\mathbf{x})$ are selected. The distribution of these μ -best solutions is estimated based on the assumption of a certain shape of the distribution (e.g., the normal distribution). After a variance scaling step $\sigma' = \delta(\sigma)$ w.r.t. an adaptive or a self-adaptive mechanism, see Section 3, the distribution $\hat{p}(\sigma')$ is the basis for sampling of λ new solutions. These steps are repeated in an iteration loop until a termination condition is fulfilled. Many EDAs employ Boltzmann- or tournament selection instead of the elitist comma-selection rule (μ best of λ) from evolution strategies [1] we apply.

The fact that the quality of a data analysis model depends on its purpose is important, in particular for EDAs. A “good” model is not necessarily the model that best represents the population, but the model that improves the search progress. Various continuous models have been proposed in the past. A comprehensive survey and taxonomy of EDAs can be found in the taxonomy by Bosman and Thierens [5].

| | |
|----|--|
| 0 | Require μ, λ |
| 1 | Start |
| 2 | Initialization of λ solutions |
| 3 | Compute $\hat{p}(\sigma)$ from μ -best solutions |
| 4 | Repeat |
| 5 | Scale variance $\sigma' = \delta(\sigma)$ |
| 6 | Sample λ solutions from distribution estimate $\hat{p}(\sigma')$ |
| 7 | Select μ best solutions |
| 8 | Compute $\hat{p}(\sigma)$ from μ solutions |
| 9 | Until termination condition |
| 10 | End |

Fig. 1. Basic EDA framework of iteratively sampling from distribution $\hat{p}(\sigma)$, scaling of σ , and estimation of $\hat{p}(\sigma)$ after selection of the μ best solutions

Normal Probability Density Function. A frequent case is the assumption of a multimodal Gaussian probability density function:

$$\mathcal{N}(\mathbf{m}, \Sigma) = \frac{1}{(2\pi)^{q/2}\det(\Sigma)} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{m})^T \Sigma^{-1} (\mathbf{x} - \mathbf{m})\right), \quad (1)$$

with covariance matrix Σ . A maximum likelihood estimation of the normal probability density function can be computed from the samples $\mathbf{x}_1, \dots, \mathbf{x}_\mu$ with:

$$\mathbf{m} = \frac{1}{\mu} \sum_{j=1}^{\mu} \mathbf{x}_j, \quad (2)$$

and:

$$\Sigma = \frac{1}{\mu} \sum_{j=1}^{\mu} (\mathbf{x}_j - \mathbf{m})(\mathbf{x}_j - \mathbf{m})^T. \quad (3)$$

The first EDA that employed this density estimate has been introduced by Rudlof and Köppen [18]. The approach is based on a vector $\mathbf{m} = (m_1, \dots, m_N)^T$ of mean values of Gaussian normal distributions. The global standard deviation σ in the univariate case is decreased during the optimization to narrow the search process. Sebag and Ducoulombier [22] have introduced a similar approach, but also use the “Hebbian” update rule for the variance. Variants with Bayesian factorization have been proposed by Bosman and Thierens [4], and Larranaga *et al.* [13]. Their experiments have shown that univariate normal probability density functions achieve worse results than multivariate variants. It turns out that the employed variants show better results on problems with linear dependencies between the variables, even with many local optima. Comparably bad results are reported on problems with non-linear dependencies.

3 Variance Scaling

The first EDAs traditionally did not use adaptive variance scaling as it was assumed that the estimation of distribution process alone achieves a satisfying adaptation process. But in practice, EDAs often suffer from premature convergence, i.e., the variance is shrinking before the optimum is reached [8][16][25]. For example, EDAs often suffer from premature convergence when the initial solution is too far away from the optimum. Artificially expanding the variance is beneficial in coping with this problem. Experiments have shown that scaling of the variance leads to improvements of the optimization process, see Ocenasek *et al.* [16]. The discussion of variance scaling shares similarities with the discussion of optimal mutation rates in the early days of genetic algorithms [9][10][14][20]. For example, Yuan and Gallagher [25] have shown success of a scaling factor of 1.5 for drawing samples from a normal probability density function. But adaptive approaches can be more flexible – as we will see in the following.

3.1 No Variance Scaling

The following experimental analysis will give insights into the behavior a Gaussian EDA employing various variance scaling techniques. As a baseline Table I shows an experimental study of a continuous Gaussian EDA without variance

scaling. We employ a variant estimating one Gaussian center \mathbf{m} , and a corresponding standard deviations σ , varying population sizes (μ, λ) , and varying selection pressures $s = \mu/\lambda$. The EDA terminates after 100 iterations.

We test the approaches on the sphere function, see Appendix A, with dimensionality $N = 5$, and optimum with fitness $f(\mathbf{x}^*) = 0.0$, and on the tangent problem, i.e., the sphere function with one constraint, and $N = 2$. Table II shows the median fitness values of differences of fitness values to optimum of 25 runs.¹. We can observe that the EDA is doing well reaching the optimum of the sphere function. The best result has been achieved with the smallest selection pressure of $s = 0.1$, in particular for population sizes around, and larger than $\lambda \geq 200$. The influence of the selection pressure on the approximation speed is obvious, introducing a descending order with decreasing pressure. Surprisingly, the influence of the population size is much weaker (employing a constant selection pressure).

In case of the tangent function significant problems in approximating the optimum with fitness $f(\mathbf{x}^*) = 2.0$ can be observed. Premature convergence of the fitness due to premature stagnation of the variance before reaching the optimum can be observed. Almost all settings fail, in particular those with the weakest selection pressure. Weak selection pressures are also disadvantageous on the unconstrained sphere function.

Table 1. Comparison of various population sizes λ , and selection pressures $s = \mu/\lambda$ for the Gaussian EDA without variance scaling. The figures show the statistical results of 25 runs measuring the difference to the optimum after termination of the EDA.

| s | 0.1 | | | | 0.25 | | | | |
|---------|---------------|---------------|---------------|---------------|------|---------------|---------------|---------------|---------------|
| | λ | 100 | 200 | 500 | 1000 | λ | 100 | 200 | 500 |
| sphere | 1.6e-36 | 3.7e-64 | 4.2e-64 | 3.7e-64 | | 7.9e-44 | 7.5e-44 | 8.5e-44 | 5.9e-44 |
| | $\pm 5.3e-25$ | $\pm 9.9e-64$ | $\pm 7.1e-64$ | $\pm 3.4e-64$ | | $\pm 5.6e-43$ | $\pm 2.1e-43$ | $\pm 7.9e-44$ | $\pm 4.0e-44$ |
| tangent | 0.0012 | 0.0006 | 0.0001 | 0.00009 | | 0.0063 | 0.0022 | 0.0003 | 0.0001 |
| | ± 0.2 | ± 0.002 | $\pm 2.5e-4$ | $\pm 1.6e-4$ | | ± 0.068 | ± 0.017 | ± 0.002 | $\pm 7.8e-4$ |
| s | 0.5 | | | | 0.75 | | | | |
| | λ | 100 | 200 | 500 | 1000 | λ | 100 | 200 | 500 |
| sphere | 8.7e-26 | 6.9e-26 | 5.7e-26 | 6.7e-26 | | 2.0e-12 | 8.0e-13 | 9.1e-13 | 7.5e-13 |
| | $\pm 8.0e-26$ | $\pm 1.0e-25$ | $\pm 4.1e-26$ | $\pm 3.9e-26$ | | $\pm 1.3e-12$ | $\pm 1.1e-12$ | $\pm 5.8e-13$ | $\pm 3.8e-13$ |
| tangent | 0.1702 | 0.0228 | 0.0086 | 0.0078 | | 18.2700 | 4.1168 | 1.5631 | 0.6549 |
| | ± 1.611 | ± 0.142 | ± 0.034 | ± 0.090 | | ± 21.961 | ± 9.525 | ± 7.216 | ± 1.653 |

3.2 Silverman Variance Adaptation

Various variance scaling techniques have been proposed in the past. The technique amBOA [16] is an approach making use of a Bayesian-like rule. In a simpler approach Bosman and Grahl [2] propose to multiply the covariance matrix with a factor that increases the variance in case of improvement from one generation

¹ The figures show the differences to the optimum $|f(\mathbf{x}^*) - f(\mathbf{x})|$ after termination of the EDA.

to the following, and decreases the variance in case of deterioration. Bosman and Thierens [6] have introduced adaptive variance scaling for multi-objective optimization EDAs. Another adaptive strategy is anticipated mean shift (AMS) by Bosman *et al.* [3]. About 2/3 of the population are sampled as usual, the rest is shifted to the anticipated direction of the gradient. If the estimate of the gradient is good, then probably all the shifted individuals survive, along with part of the non-shifted individuals, and the variance estimate in the direction of the gradient is larger than usually.

We employ a variance adaptation rule motivated from non-parametric density estimation in the following. A Gaussian-based EDA shares significant similarities with kernel density estimation (KDE) [15,24]. The $(\mu + \lambda)$ -evolution strategy (ES) can be viewed as EDA with Gaussian KDE, and σ -self-adaptive variance scaling. From this point of view we can apply KDE related bandwidth adaptation techniques. One promising approach is Silverman's rule-of-thumb that is statistically motivated [23]:

$$h = \hat{\sigma} c \mu^{-1/5}, \quad (4)$$

with μ solutions, mean \bar{x} of the population, the sample standard deviation $\hat{\sigma}$, and $c = 1.06$ for a Gaussian distribution. An experimental analysis can be found in Table 2, where Silverman's rule-of-thumb is tested with the same parameter combinations we used in Section 3.1. One main observation is that the Silverman-based variance scaling results in a much better approximation speed on the sphere function for population sizes $\lambda \geq 500$, and small s . Large population sizes are required for a reliable estimation of σ with Silverman's rule. No significant difference can be observed on the tangent problem in comparison to the previous approach.

Table 2. Analysis of the Gaussian EDA with Silverman's rule-of-thumb

| s | λ | 0.1 | | | | 0.25 | | | |
|---------|-------------|-------------|---------------|---------------|------|--------------|---------------|---------------|---------------|
| | | 100 | 200 | 500 | 1000 | 100 | 200 | 500 | 1000 |
| sphere | 0.0203 | 3.1e-27 | 6.8e-86 | 1.3e-86 | | 3.2e-06 | 1.5e-49 | 4.8e-66 | 1.6e-66 |
| | ± 3.659 | ± 0.014 | $\pm 2.1e-85$ | $\pm 1.4e-86$ | | ± 0.108 | $\pm 9.1e-15$ | $\pm 3.4e-66$ | $\pm 1.1e-66$ |
| tangent | 0.0025 | 0.0003 | 0.00006 | 0.00005 | | 0.0167 | 0.0053 | 0.0011 | 0.0004 |
| | ± 0.396 | ± 0.014 | $\pm 6.3e-4$ | $\pm 1.0e-4$ | | ± 0.320 | ± 0.054 | ± 0.006 | ± 0.001 |
| s | λ | 0.5 | | | | 0.75 | | | |
| | | 100 | 200 | 500 | 1000 | 100 | 200 | 500 | 1000 |
| sphere | 0.0074 | 1.1e-09 | 2.8e-15 | 1.2e-47 | | 5.0470 | 0.1634 | 0.0318 | 0.0003 |
| | ± 0.134 | ± 0.047 | $\pm 3.1e-06$ | $\pm 1.7e-19$ | | ± 11.128 | ± 2.006 | ± 0.202 | ± 0.003 |
| tangent | 0.3787 | 0.7170 | 0.0940 | 0.0428 | | 64.2591 | 34.1626 | 28.4621 | 29.4665 |
| | ± 7.609 | ± 1.669 | ± 0.404 | ± 0.151 | | ± 42.060 | ± 24.978 | ± 10.383 | ± 6.369 |

3.3 σ -Self-Adaptation

Self-adaptation is an automatic parameter control technique that has originally been introduced in the context of evolution strategies [1]. Self-adaptation is based on the stochastic control of strategy parameters. The parameters are bound to

the chromosome of a candidate solution, and are evolved with the help of evolutionary operators. Successful candidate solutions inherit their genetic material: the chromosome defining the solution *and* the strategy variables. Self-adaptation is based on the following assumption: successful strategy parameter lead to successful solutions, which likely inherit these successful strategy parameterizations to the following generation. Yunpeng *et al.* [7] point out that self-adaptation and cumulative path-length control [17] cannot be easily implemented into EDAs.

We will experimentally analyze self-adaptive variance scaling in the following. For this sake we employ an ES-oriented scaling of the variance: the log-normal mutation. Log-normal mutation changes the multivariate step sizes σ (i.e., the variance of the multivariate normal distribution $\mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$) with the log-normal rule [21]:

$$\sigma' := e^{(\tau_0 \mathcal{N}_0(0,1))} \cdot (\sigma_1 e^{(\tau_1 \mathcal{N}_1(0,1))}, \dots, \sigma_N e^{(\tau_1 \mathcal{N}_N(0,1))}), \quad (5)$$

for each individual before applying the mutation operator:

$$\mathbf{x}' := \mathbf{x} + (\sigma_1 \mathcal{N}_1(0, 1), \dots, \sigma_N \mathcal{N}_N(0, 1)) \quad (6)$$

on the objective variables. From another perspective, the algorithm is similar to a $(1, \lambda)$ -ES with σ -self-adaptation.

Table 3. Analysis of Gaussian EDA with σ -self-adaptation

| s | 0.1 | | | | 0.25 | | | |
|---------|---------------|---------------|---------------|---------------|--------------|--------------|-------------|-------------|
| | λ | 100 | 200 | 500 | 1000 | 100 | 200 | 500 |
| sphere | 3.5e-10 | 1.6e-10 | 9.1e-11 | 1.5e-10 | 0.1012 | 0.0406 | 0.0186 | 0.0111 |
| | $\pm 3.4e-08$ | $\pm 2.3e-09$ | $\pm 2.8e-10$ | $\pm 2.4e-10$ | ± 0.126 | ± 0.056 | ± 0.012 | ± 0.005 |
| tangent | 0.0012 | 0.0003 | 0.0003 | 0.0001 | 0.0086 | 0.0043 | 0.0017 | 0.0013 |
| | ± 0.001 | $\pm 3.0e-4$ | $\pm 3.0e-4$ | $\pm 1.0e-4$ | ± 0.015 | ± 0.004 | ± 0.001 | ± 0.001 |
| s | 0.5 | | | | 0.75 | | | |
| | λ | 100 | 200 | 500 | 1000 | 100 | 200 | 500 |
| sphere | 2.1599 | 0.9067 | 0.4962 | 0.2435 | 30.3439 | 22.6315 | 9.3148 | 7.5198 |
| | ± 2.195 | ± 0.648 | ± 0.288 | ± 0.128 | ± 18.418 | ± 13.078 | ± 4.386 | ± 4.441 |
| tangent | 0.3718 | 0.1115 | 0.1115 | 0.048 | 203.2667 | 167.2759 | 64.0001 | 18.1263 |
| | ± 1.003 | ± 0.279 | ± 0.119 | ± 0.047 | $\pm 1.3e3$ | $\pm 9.7e2$ | $\pm 1.2e3$ | $\pm 2.0e2$ |

Table 3 shows the experimental results of the Gaussian EDA that estimates the mean of the Gaussian distribution, but scales the variance with self-adaptation. We observe that self-adaptation is slower than the previous two variance scaling approaches. The tendency of a faster approximation for strong selection pressures can be confirmed. But we can observe that self-adaptation fails to control the step sizes in case of the tangent problem. It is likely that the same mechanisms are responsible that have been reported for evolution strategies with σ -self-adaptation [11]: the constrained area of the search space cuts off huge parts of the mutation

area, and decreases the success probability. But σ -self-adaptation reacts by decreasing the step sizes with the intention to increase the success probability again. In the following, we propose to solve this problem by rotation of the coordinate system with a Hessian estimation approach.

3.4 Estimation of the Hessian

Approaches that are able to exploit more gradient or Hessian information about the search space are expected to be more powerful in approximating the optimal solution. We expect the same if we employ the EDA with such information. We employ an approach based on estimation of the Hessian in the following, oriented to the work of Rudolph [19] for evolution strategies. He has shown that optimal distributions for Gaussian mutations can be achieved, if the mutation coordinate system is aligned to the contour lines of the fitness function. For example, elliptical problems can be transformed into simpler symmetric ones. This can be achieved by using the inverse of the Hessian as correlation matrix (see Equation 1) $\Sigma = \mathbf{H}^{-1}$. Rudolph [19] proposes a computationally expensive method (runtime $\mathcal{O}(N^6)$) that we will employ for the Gaussian EDA in the following. The Hessian matrix is unknown, and must be estimated. CMA-based evolution strategies share this idea, but *approximate* the Hessian. Bosman and Thierens [5] point out the similarities between CMA-ES and multivariate EDA approaches, but also place emphasis on the fact that the parameters are set in different kind of ways.

Rudolph [19] proposes the following steps that are only shortly repeated here. For a comprehensive introduction of all necessary steps to compute the Hessian matrix from a set of observations, we refer to his depiction. If we assume the function is locally quadratic, the second order Taylor expansion is:

$$f(\mathbf{x}) = b_0 + \mathbf{b}^T \mathbf{x} + \mathbf{x}^T \mathbf{B} \mathbf{x}, \quad (7)$$

with parameters $(b_0, \mathbf{b}, \mathbf{B})$ that are collected in vector $\mathbf{c} = (c_1, \dots, c_N)^T$, and can be estimated by a least squares estimator. For this sake we need a population of at least $n = \frac{1}{2}N^2 + \frac{3}{2}N + 2$ points leading to training samples $(\mathbf{x}_1, f(\mathbf{x}_1)), \dots, (\mathbf{x}_n, f(\mathbf{x}_n))$ that define a linear equation $\mathbf{G}\mathbf{c} = \mathbf{f}$, see [19]. From these n training samples, vector \mathbf{c} can be estimated by $\hat{\mathbf{c}} = (\mathbf{G}^T \mathbf{G})^{-1} \mathbf{G}^T \mathbf{f}$. Selecting the entries $\hat{c}_{N+2}, \dots, \hat{c}_n$ that define the Hessian part, and normalizing leads to the estimate $\hat{\mathbf{H}}$, from which the desired matrix $\Sigma = \hat{\mathbf{H}}^{-1}$ can be computed. In each generation the $\frac{1}{2}N^2 + \frac{3}{2}N + 2$ best solutions are used to estimate the Hessian. This estimate $\hat{\mathbf{H}}$ is used to rotate and scale the coordinate system that is basis of the Gaussian mutations. Table 4 shows the corresponding results. It shows that the Hessian approach is finally able to approximate the optimum of the tangent problem. This confirms our expectations as the coordinate system is rotated according to the approximated Hessian. In comparison to the variant without variance scaling similar results (but no improvements) are achieved on the sphere function.

Table 4. Analysis of Gaussian EDA with estimation of the Hessian

| s λ | 0.1 | | | | 0.25 | | | |
|----------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|
| | 100 | 200 | 500 | 1000 | 100 | 200 | 500 | 1000 |
| sphere | 2.7e-39 | 1.1e-65 | 2.0e-64 | 7.3e-63 | 1.5e-44 | 9.1e-44 | 5.2e-43 | 2.2e-45 |
| | $\pm 4.2e-39$ | $\pm 2.9e-64$ | $\pm 1.1e-64$ | $\pm 6.5e-62$ | $\pm 2.7e-44$ | $\pm 3.2e-44$ | $\pm 4.8e-43$ | $\pm 8.1e-44$ |
| tangent | 3.3e-8 | 2.7e-8 | 3.1e-9 | 6.3e-8 | 6.6e-12 | 4.9e-12 | 9.1e-11 | 5.3e-11 |
| | $\pm 2.8e-8$ | $\pm 8.1e-8$ | $\pm 2.2e-9$ | $\pm 7.5e-8$ | $\pm 9.1e-12$ | $\pm 5.9e-12$ | $\pm 1.1e-11$ | $\pm 4.8e-10$ |
| s λ | 0.5 | | | | 0.75 | | | |
| | 100 | 200 | 500 | 1000 | 100 | 200 | 500 | 1000 |
| sphere | 7.2e-27 | 1.9e-25 | 3.9e-25 | 1.7e-26 | 8.4e-13 | 9.1e-13 | 1.1e-13 | 6.2e-13 |
| | $\pm 6.5e-26$ | $\pm 1.6e-24$ | $\pm 3.7e-25$ | $\pm 1.2e-26$ | $\pm 9.9e-13$ | $\pm 5.2e-13$ | $\pm 7.1e-12$ | $\pm 1.1e-13$ |
| tangent | 2.3e-12 | 6.8e-21 | 8.1e-21 | 5.3e-22 | 8.9e-17 | 4.4e-15 | 5.4e-16 | 7.5e-17 |
| | $\pm 4.4e-12$ | $\pm 7.5e-21$ | $\pm 3.4e-20$ | $\pm 8.9e-22$ | $\pm 2.8e-17$ | $\pm 6.1e-15$ | $\pm 3.7e-15$ | $\pm 7.7e-17$ |

4 Conclusion

The success of EDAs significantly depends on adequate variance scaling. Among the different scaling techniques we compared a standard approach without variance scaling to an adaptive approach based on Silverman's rule-of-thumb, to σ -self-adaptation, and an approach that is based on estimation of the Hessian. Silverman's approach with large population sizes and high selection pressure turned out to be the fastest optimization method on the sphere function. Only the Hessian approach allows an approximation of the tangent problem without premature decrease of the step sizes. Of course, the computational effort is comparatively large to estimate the Hessian in each generation, in particular for large N , but this effort might be useful to invest in cases where high accuracy is required.

A Test Problems

In this paper we employ the sphere function, and the constrained variant denoted as tangent problem.

Sphere Function. The sphere function is defined as follows:

$$f(\mathbf{x}) = \mathbf{x}\mathbf{x}^T \quad (8)$$

with $\mathbf{x} \in \mathbb{R}^N$. The optimal solution is $\mathbf{x}^* = (0, \dots, 0)^T$ with $f(\mathbf{x}^*) = 0$.

Tangent Problem. The second problem is called tangent problem (TR). It is based on $f(\mathbf{x})$ from the sphere function subject to one linear constraint:

$$g(\mathbf{x}) = \sum_{i=1}^N x_i - N > 0 \quad (9)$$

with $\mathbf{x}^* = (1, \dots, 1)^T$ and $f(\mathbf{x}^*) = N$. The success rates on TR get worse when approximating the optimum [12].

References

1. Beyer, H.-G., Schwefel, H.-P.: Evolution strategies – a comprehensive introduction. *Natural Computing* 1(1), 3–52 (2002)
2. Bosman, P.A.N., Grahl, J.: Matching inductive search bias and problem structure in continuous estimation-of-distribution algorithms. *European Journal of Operational Research* 185(3), 1246–1264 (2008)
3. Bosman, P.A.N., Grahl, J., Thierens, D.: Enhancing the performance of maximum-likelihood gaussian edas using anticipated mean shift. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 133–143. Springer, Heidelberg (2008)
4. Bosman, P.A.N., Thierens, D.: Expanding from discrete to continuous estimation of distribution algorithms: The idea. In: Deb, K., Rudolph, G., Lutton, E., Merelo, J.J., Schoenauer, M., Schwefel, H.-P., Yao, X. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 767–776. Springer, Heidelberg (2000)
5. Bosman, P.A.N., Thierens, D.: Numerical optimization with real-valued estimation-of-distribution algorithms. In: Scalable Optimization via Probabilistic Modeling, pp. 91–120 (2006)
6. Bosman, P.A.N., Thierens, D.: Adaptive variance scaling in continuous multi-objective estimation-of-distribution algorithms. In: Genetic and Evolutionary Computation Conference (GECCO), pp. 500–507. ACM Press, New York (2007)
7. Cai, Y., Sun, X., Xu, H., Jia, P.: Cross entropy and adaptive variance scaling in continuous eda. In: Genetic and Evolutionary Computation Conference (GECCO), pp. 609–616. ACM Press, New York (2007)
8. Grahl, J., Minner, S., Rothlauf, F.: Behaviour of umdac with truncation selection on monotonous functions. In: Congress on Evolutionary Computation (CEC), pp. 2553–2559 (2005)
9. Grefenstette, J.: Optimization of control parameters for genetic algorithms. *IEEE Trans. Syst. Man Cybern.* 16(1), 122–128 (1986)
10. Jong, K.A.D.: An analysis of the behavior of a class of genetic adaptive systems. PhD thesis, University of Michigan (1975)
11. Kramer, O.: Premature convergence in constrained continuous search spaces. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 62–71. Springer, Heidelberg (2008)
12. Kramer, O.: Premature convergence in constrained continuous search spaces. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 62–71. Springer, Heidelberg (2008)
13. Larrañaga, P., Etxeberria, R., Lozano, J.A., Peña, J.M.: Optimization in continuous domains by learning and simulation of gaussian networks. In: Genetic and Evolutionary Computation Conference (GECCO), pp. 201–204. ACM Press, New York (2000)
14. Mühlenbein, H.: How genetic algorithms really work: Mutation and hillclimbing. In: Parallel Problem Solving from Nature (PPSN), pp. 15–26 (1992)
15. Nadaraya, E.: On estimating regression. *Theory of Probability and Its Application* 10, 186–190 (1964)
16. Ocenasek, J., Kern, S., Hansen, N., Koumoutsakos, P.: A mixed bayesian optimization algorithm with variance adaptation. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiño, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 352–361. Springer, Heidelberg (2004)

17. Ostermeier, A., Gawelczyk, A., Hansen, N.: A derandomized approach to self adaptation of evolution strategies. *Evolutionary Computation* 2(4), 369–380 (1995)
18. Rudlof, S., Köppen, M.: Stochastic hill climbing by vectors of normal distributions. In: Proceedings of the 1st Online Workshop on Soft Computing, Nagoya, Japan (1996)
19. Rudolph, G.: On correlated mutations in evolution strategies. In: Parallel Problem Solving from Nature (PPSN), pp. 107–116 (1992)
20. Schaffer, J.D., Caruana, R., Eshelman, L.J., Das, R.: A study of control parameters affecting online performance of genetic algorithms for function optimization. In: International Conference on Genetic Algorithms - ICGA 1989, pp. 51–60 (1989)
21. Schwefel, H.-P.: Adaptive Mechanismen in der biologischen Evolution und ihr Einfluss auf die Evolutionsgeschwindigkeit. Interner Bericht der Arbeitsgruppe Bionik und Evolutionstechnik am Institut für Mess- und Regelungstechnik, TU Berlin (July 1974)
22. Sebag, M., Ducoulombier, A.: Extending population-based incremental learning to continuous search spaces. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN 1998. LNCS, vol. 1498, pp. 418–427. Springer, Heidelberg (1998)
23. Silverman, B.W.: Density Estimation for Statistics and Data Analysis. Monographs on Statistics and Applied Probability, vol. 26. Chapman and Hall, London (1986)
24. Watson, G.: Smooth regression analysis. *Sankhya Series A* 26, 359–372 (1964)
25. Yuan, B., Gallagher, M.: On the importance of diversity maintenance in estimation of distribution algorithms. In: Genetic and Evolutionary Computation Conference (GECCO), pp. 719–726. ACM Press, New York (2005)

Hierarchically Structured Energy Markets as Novel Smart Grid Control Approach

Jörg Lässig¹, Benjamin Satzger², and Oliver Kramer³

¹ Enterprise Application Development Group, University of Applied Sciences Zittau/Görlitz, Germany

² Distributed Systems Group, Vienna University of Technology, Austria

³ Algorithms Group, ICSI Berkeley, USA

Abstract. The paper investigates the self-stabilization of hierarchically structured markets. We propose a new approach that is motivated by the physical structure of the energy grid and generalizes classical market structures in a natural way. Hierarchical markets have several advantages compared to monolithic markets, i.e., improved reliability and scalability, locality of information, and proximity of energy production and consumption. By simulating scenarios based on real world consumption and production data including households, different renewable energy sources, and other plant types, we present a proof-of-concept of stability of the hierarchical markets in various simulations.

Keywords: smart grid, hierarchical markets, simulation, agents.

1 Introduction

Control concepts for grid markets in practice are currently completely centralized [2]. Simply spoken, a sensor network all over the energy grid collects information for a central decision making on the producer side. This architecture is in complete antagonism to modern communication and information infrastructures, but an important first step as the recent increase of critical network states and blackouts in today's centralized controlled grids shows [5]. Due to the increasing number of decentralized producers, different storage technologies and renewable energies, alternatives to those concepts are eagerly desired. Promising are fundamentally decentralized control approaches, which directly reflect the basic structure of the network. A common way to achieve this is the application of principles from distributed artificial intelligence [7][21]. We think that agents competing in a market are a promising approach to solve this problem for a distributed and automatic control of grids.

There are many studies which show that the market price is a means to control energy production and consumption in a natural way [18][22]. Unlike those previous studies, we emphasize on a unified generic modeling of grids at all levels. We propose to describe grids as hierarchical overlapping markets what allows to model households as well as industrial plants, and even the international electricity market. While this market is a real instance of an energy market, our

architecture can be seen as a rigorous application of market-control to all levels of a grid, which –without adding further ingredients– abstracts from physical aspects.

This work is structured as follows. In the forthcoming section we review classical monolithic markets and discuss successful auction principles for this setup. Then, in Section 3 we introduce hierarchical markets and explain the benefits from adopting hierarchical structures to smart grid control scenarios. In Section 4 case studies explain the dynamic behavior of the approach and clarify abstract concepts by means of examples. In Section 5 we explain how hierarchical market scenarios have been implemented and which agent types and strategies have been applied. In specific, strategies to balance adjacent sub-markets are motivated. Section 6 describes the behavior of the proposed scenarios. The development of important parameters as price, energy production, and battery level over time are visualized in different scenarios. Section 7 concludes the paper and explains possible future research.

2 Related Work

Much work has been done on artificial intelligence solutions to manage energy networks, in specific applying distributed artificial intelligence or machine learning; for early work see e.g. [19][24][12]. Mascem [17] is a multi-agent system (MAS) to simulate competitive electricity markets. It is intended to provide a means to analyze the impact of market rules and agents' strategies on the system. Unlike our approach, Mascem consists of only one energy market divided into a trading pool, a floor for bilateral contracts, and a range of agents with specific roles. Pipattanasomporn *et al.* [16] propose to use a MAS for the special case to control a microgrid. A microgrid is an integrated energy system which can operate in parallel with the grid or in an intentional island mode. Also Li *et al.* [7] developed multi-agent technology for the management and control of distributed energy resources by explicitly coordinating distributed energy resources comprising customer loads and generators to aggregate sufficient distributed capacity to be of strategic value to market participants. Although their setup is different from our's, this aggregation can be seen as alternative but less general approach for adding more structure to the grid.

Pricing rules in electricity markets have been experimentally investigated, e.g., by Xiong *et al.* [21]. Of course, these rules have also major influence on the behavior of the agents; different bidding strategies are considered by Kian *et al.* [8] and optimal strategies are known as well [3]. Also classical auction schemes as Vickrey-Clarke-Groves auctions [6] or other game theoretic approaches and approaches from computational learning theory have been applied [12]. It should be clear, that there may be large differences between the considered markets if one looks at their inner structure, but they all fit in the scheme described within this paper. The structure discussed in this work leaves market details open for configuration. A major design goal is to assure that detailed planning is possible for energy suppliers and consumers while maintaining scalability.

3 Benefits of Hierarchy and Structure

The central new concept of our studies is the introduction of an additional hierarchical structure in energy markets, so that it dissolves into connected submarkets. This is motivated several studies on hierarchical markets as [4][10] and the structure of the physical grid, which consists of several hierarchical levels, e.g., a single household, an urban district, or a city. On higher level, the grid is also physically different, i.e., on lowest level there is the low-voltage-grid (400/230V, connected to the medium-voltage-grid, on-site power plants, solar panels), above that the medium-voltage-grid (1-50kV, connected to the high-voltage-grid, small urban power plants, wind parks, solar power plants), there is a high-voltage-grid (110kV, connected to the supergrid, medium-sized power plants, industrial consumers), and the supergrid (220-380kV, nuclear power, coal-fired power plants, hydro-electric power plants, etc.). From a practical perspective it is not very realistic to organize all participants on these different levels of the overall grid in one market. For practical applications of smart grids, this physical structure of the grid has to be introduced to the simulation model in order to investigate a realistic scenario.

3.1 Modeling the Physical Structure of the Grid

In general the physical structure of the electricity network can be modeled as a graph with nodes $v_1, v_2, \dots, v_n \in V$ and transmission lines $l \in L$, denoted as $G = (V, L)$. Each node is identified with some physical entity which is represented by an agent a_i at node v_i , i.e., there is an agent set $A \equiv V$. The *agents* act on behalf of *entities* of the electric grid. The nodes (agents) can just represent buyers and/or sellers but can also model voltage transformation substations. A modeling of this physical structure of the grid has advantages if the transmission lines and their load are relevant. E.g., in [21] the load of each single transmission line is continuously tracked, which is possible based on methods to calculate the power flow in electricity networks. While the calculation of the full alternating current (AC) power flow model has many pitfalls due to a large number of alternative solutions, which fulfill the nonlinear power balance equations, the directed current (DC) model is accurate enough and can be calculated based on a set of linear equations [15]. The focus of our studies is not the calculation of the flow on transmission lines, because in practice these lines are sufficiently dimensioned also for peak demands. Instead, our research in this paper investigates three central questions: How can hierarchical markets with market substructures be controlled? What are suitable strategies to balance adjacent markets? How is energy production control possible in hierarchical markets? To investigate these questions we relax the structure of the grid to a tree structure which reflects the logical relationships of agents in the network and their allocation to sub-markets but suppresses details like submission lines. This is described in the following section.

3.2 Modeling the Logical Structure of the Grid

First, assume that all agents are continuously connected to the internet. Of course, specific security mechanisms have to be implemented to prevent manipulation and particularly important agents should be secured by backup agents, able to replace them in case of failures. These considerations are highly important for the application of a system as proposed, but not topic of this work. We restrict ourselves to develop the self-contained structure and explain its proposed dynamics by different case studies.

Based on these insights we model the logical structure the electric grid $G = (V, L)$ as a tree $G' = (V, T)$ with the same vertex (agent) set but an semantically motivated edge set $T \subseteq L$ to form a tree. The tree edges T model the relationships among the agents where $(c, p) \in T$ means that p is part of c . Entities are characterized by having the ability to consume, produce, and transmit power. The task of an agent is to control the provisioning of sufficient energy and to reduce overcapacity over appropriate lines on behalf of the entities. These control mechanisms are implemented by using the paradigm of markets as it is done in practice in many cases to assign limited resources.

The whole electric grid is hierarchically divided into *virtual markets* of different granularity. A virtual market or simply market $M = (A', R)$ consists of a number of participating agents $A' \subseteq A$, and a number of market rules R to define its functionality. Each agent $a \in A$ forms a market together with all its children, i.e., $M^a = (\{c \in A \mid (a, c) \in T\} \cup \{a\}, R^a)$. The agent a of a market M^a is called *head* of the market. Agents are allowed to trade with other agents within the same virtual market according to the market rules but not beyond market boundaries. As the market head usually member of more than one market, it is allowed to trade in different (usually adjacent) markets. In case of agents being at a leave of the tree a market consists only of that agent itself. As no trading can take place in a single participant market, such markets can be neglected.

4 Case Studies for Hierarchical Markets

In this section we explain how the introduced architecture can be applied to enable a smart grid. We present two case studies to highlight the functionality of our architecture on different hierarchical levels.

4.1 Energy-Mix in a City

Consider the local electricity market of a city, and let an agent c be the head of the market. The actors in this market $M^c = (A^c, R^c)$, represented by an agent each, are households of the township, industrial plants, and other energy consumers, but there may also be energy producers as wind parks, which belong to this city or also biogas plants. In general markets should be defined by geographical distance but also by the voltage level of the entities, which should be the same for one market.

The market rules are described by products and protocols of the market. In this exemplary case each participant can *buy* power, where each buy-order consists of a *time window* $[t_s^r, t_e^r]$, denoting the earliest possible time and the latest possible requested time, as well as further information. These could be the least possible time of power consumption, an *amount* in kWh, and an *order modifier*, which can be a *maximum price* P or also *market order*. A maximum load at a certain time should be specified too.

Sell-orders consist of the same items, but P is a minimum price. The prices are assumed to be better for the consumer if it leaves more freedom to be fixed by the supplier, i.e., more freedom for the supplier to shift load. If energy is consumed without buying it in advance, high penalty prices are charged. A result of this construction is that the agents will try to specify demand as specific as possible, and leave as much freedom as possible for the supplier. Note that all orders that have not been executed can be deleted at any time by the ordering party or decay at time t_s . Of course, more complex or specific products are also possible, e.g., for renewable energy.

4.2 Electricity Market

The highest market level of this model are (national and international) electricity markets. Such organized electricity markets have been established in many countries starting in the 90ies (see [1] as an example), where large amounts of energy are traded by the “big players” in the energy market. The existing electricity market maps quite well to the suggested basic architecture of the system. The target of the energy market today is to reach an economically optimal allocation of energy. In the suggested approach this is a target on all market levels, approached by implementing an agent-oriented structure as suggested.

The available products are standardized energy blocks of different *duration* and *amount*, which can be traded continuously or only at specific times, depending on the market. Very important is the temporal availability of energy, i.e., the price may react to new weather forecasts to satisfy the different demand of the market [23]. Currently, the electricity market consists of different sub-markets. In a *future market* long term transactions are conducted, while short term transactions are conducted at the *spot market*. This covers contracts from several years down to 24 hour blocks, 12 hour blocks, and also one hour blocks to model peak times in the demand. In some markets the blocks go down to the level of a few minutes. This important market is not restricted to electronic agents – also humans participate to control these important extensive trades.

5 Simulated Scenario

In the following simulated scenario we further formalize the market rules and standardize the available products. The markets are day ahead and divided into trading periods of 30 minutes length each. We consider different scenarios to investigate the questions as discussed in Section 3.1.

5.1 Smart Grid Architecture

We model an electricity grid consisting of three cities. In each city we consider ten representative buildings for our simulation scenario, where some of them are equipped with solar panels.¹ One of the cities is equipped with an additional wind generator. Each of these cities is organized in a separate market. The market heads of these markets are organized in a higher market. Additionally, there is a gas power plant, which is also arranged in the higher market.

5.2 Agent Types and Their Characteristics

There are various types of agents in the system: house agents, house agents with solar panel, wind plant agents, gas power plant agents, and market agents.

Each agent is equipped with a battery of specific dimension, which is used as energy buffer. This battery models demand which is transferable in time as, e.g., to cool down the freezer, to heat the hot water tank, to recharge the electrical car. By dimensioning the batteries differently large, the influence of shiftable energy demand in the grid can be investigated. Table I shows the battery capacities for the initial experiments.

Table 1. Battery capacities and energy consumption

| Entity | Battery capacity | Energy consumption |
|-----------------|------------------|--------------------------|
| Household | 20,000 W | between 0 … 1,000 W |
| Solar household | 50,000 W | between -3,000 … 1,000 W |
| Gas power plant | 500,000 W | unlimited production |
| Wind plant | 200,000 W | between -12,000 … 0 W |

The third column in Table I shows the energy consumption of the different entities in the simulation model. Note, that households which are equipped with solar panels, the gas power plant, and the wind plant have negative consumption, i.e., they produce energy. These figures are based on real world data from the Western Wind Resources Dataset [4], from the PVWATTS calculator [3], and the Review of Residential Electrical Energy Use Data [1]. The energy consumption the simulations are based on is visualized in Figure I.

In particular, the data for Sacramento, San Francisco, and Los Angeles has been utilized. The real production or consumption values are calculated by sampling from a uniform distribution in the given interval in the third column of Table I (instead for the gas power plant, which is regulated price-driven).

5.3 Agent Strategies

The different agents in the smart grid have different strategies. Most of the agents trade according to a battery-price-strategy, which means, they are sellers if their

¹ Experiments have shown that the qualitative behavior of the system does not change for a larger number of agents and the dynamics of large systems cannot be visualized.

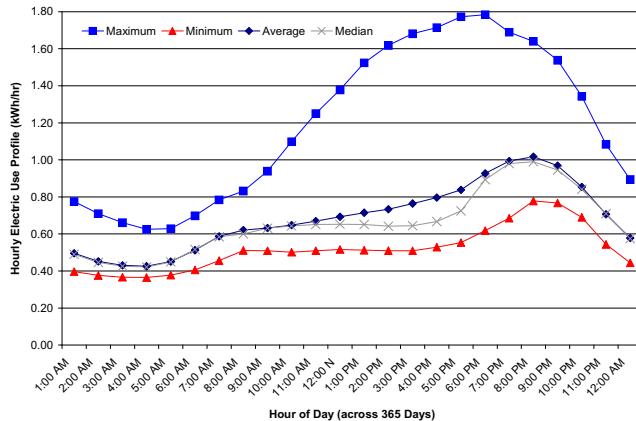


Fig. 1. Hourly average residential load profile, averaged over the year (Southern California, Edison territory) [11]

battery gets too full due to energy production or they buy if their battery gets too empty due to energy consumption. A (future) battery level of 50 percent is used as a reference point to change between these two different behaviors. The farther the level of the battery charging level is from this value, the more aggressively the agent tries to buy or sell energy by choosing an adequate price.

Depending on the battery load $\ell^{i,t} \in [0, 1]$ at time t , the agent i tries to buy ($\ell^{i,t} < 0.5$) or sell ($\ell^{i,t} > 0.5$) energy to the current market price p^t plus $\Delta^{i,t}$, where

$$\Delta^{i,t} = \begin{cases} (1 - 2\ell^{i,t}) \cdot (p_{\max} - p^t) & \text{if } \ell^{i,t} \leq 0.5 \\ (1 - 2\ell^{i,t}) \cdot p^t & \text{otherwise} \end{cases} \quad (1)$$

for $i = 1, 2, \dots, |A|$ with p_{\max} as assumed maximum price to be payed in our model. The gas power plant produces only power, if the market price is higher than a certain break even value, which marks the price level where the operation of the plant starts to realize profit. As a consequence, in the experiments of the further sections it can be observed that the gas power plant produces power in periods, but not continuously.

5.4 Load Prediction

As we have seen in Table II, the gas power plant only sells energy, households without solar panel only buy energy, and households with solar panel are able to buy and sell power. The decision to buy or to sell it based on the *future battery level*, which we now define in a formal way. In general, the future battery level is defined as $\ell^{i,t+1} = \ell^{i,t} - c^{i,t}$, $i = 1, 2, \dots, |A|$, where $\ell^{i,t}$ is the battery level of agent i at the beginning of period t and $c^{i,t}$ is the predicted energy consumption of agent i in period t (measured as portion of the battery capacity), which can be positive or negative (energy production). The ℓ -value in Equation II is equivalent

to the value $\ell^{i,t+1}$ at the end of the period t , i.e., the buy or sell orders during period t are adjusted according to the predicted battery level at the end of period t . This is only possible if the power consumption $c^{i,t}$ in period t can be well predicted. This has been realized by regression methods as described in [9], which is not further stressed here. The quantity of bought or sold energy is then $0.5 - (\ell^{i,t} - c^{i,t})$, i.e., the agents buy a quantity of power so that the predicted energy level of the battery is exactly 0.5 at the end of the period.

5.5 Trading Rounds

The continuous double auctions are organized in a way that ten trading rounds are executed within one trading period, i.e., each agent has ten opportunities to buy (or sell) [20]. The clearing of the market is organized according to this auction type. One of these trading periods here is equivalent to 30 minutes real time. In each round, multiple buyers and sellers are allowed to submit bids and asks and the market clears whenever a bid and an ask matches, in which case the round ends. In any case it is required that a new submitted bid or ask improves on the previously best bid or ask, called the outstanding bid or outstanding ask, respectively. This means bids and asks are not queued in the system and simply erased, if a better one is submitted.

5.6 Balancing Adjacent Markets

The market heads consist of two traders – a lower market trader and a higher market trader. In the model the market head is an agent who acts altruistic towards the agents in his market, but acts egoistically in the higher market. This means the two trader of the market head act according to different strategies.

Each market head owns an (hypothetic) battery which is used to save excess energy in the own market to be sold in the higher market or to save energy from the higher market to be sold in the lower market. To ensure altruistic behavior towards the agents in the lower market, an average buying price p_a is saved together with the quantity contained in the battery. It is updated after each trade according to the following rules. Assume that the battery is initially not empty but filled with load $\ell \in [0, 1]$, acquired to the average buying price p_a . The agent tries to balance the lower market by conducting transactions in the higher market. To do that, the ratio of unsatisfied asks and bids in the last round of a period is used. If there is an ask overage, the higher market trader tries to sell energy in the higher market to a price at least the average ask price. If there is a bid overage, the higher market trader tries to buy energy in the higher market to at most the average buy price in the lower market. *Lower Market Transactions:* Sell to a price p_a ; buy to the current market price. *Higher Market Transactions:* Buy to a price not higher than the current ask price in the lower market. Sell to a price not lower than p_a . All gains and losses of the market agent are shared by the market participants in the lower market according to their average business volume in the last trading period at the end of each trading period.

6 Empirical Evaluation

The proposed smart grid structure is designed according to the rule *as local communication and energy transfer as possible but as global communication and energy transfer as necessary*. This has been realized by a hierarchical structure of the market, which hides on each level existing higher and lower markets to certain extent (only adjacent market levels are visible for the agents). As a major advantage of this special structure it is conjectured that the communication and energy production overhead in the network can be reduced, because it is closer oriented to the actual energy consumption, which reduces costs. The major new ingredient which is necessary to control these hierarchical market structures is the introduction of *multi-market-trader-strategies*. To answer the three major questions as posed in Section 3.1, the following experiments are carried out: Simulation and investigation of the overall market behavior in the scenario. Investigation of the market price development in the different markets. A strategy is considered to be *good*, if the prices in adjacent markets are coupled. Investigation of the amount of produced energy w.r.t. consumed energy.

We run the simulation with the setup as previously described. Figures 2 and 3 show the development of the price in the different sub-markets, the energy production, and the battery level of the agents, respectively.



Fig. 2. Price development in the sub-markets

Figure 2 indicates that the strategy as described above couples the markets close together, so that the prices of all sub-markets remain in a narrow band, which has to be investigated in more detail by future experiments. The same is visualized well for the bid and ask price in the same market.

Figure 3 shows the energy production behavior of the different types of participants in the smart grid. For the households (red line) it can be seen that the energy consumption changes during the simulation period. A periodic change

of the energy consumption/ production can be obtained from the households with solar panels, due to the different day-times and the different solar radiation over the day. The produced wind power is fluctuating during the simulation time. Interesting is the behavior of the gas power plant: Multiple times it is switched on and off, and the energy production is fluctuating in the course of time. Comparing the Figures 2 and 3 shows that especially in times of high prices the energy production gets started or increased.

Furthermore, our experimental analysis has shown that all agents are able to refill their battery to a level of fifty percent after consuming or producing energy in a relatively short period of time (instead for the gas power plant, which battery gets usually recharged to 100 percent). The lowest obtained energy level during the simulation is about 20 percent for a house in market M4, which consists only of consuming households. Concerning the interaction between solar energy and batteries we have observed an interesting pattern, where an increased production of solar energy fills the batteries of several agents by more than 50 percent. During that time the battery of the gas power plant is completely filled, i.e., no energy from the gas power plant is consumed. This is a typical case of self-regulation behavior of the energy grid – renewable energy is consumed preferentially if it is available. In our experiments we have analyzed the response to increased solar energy production in the market price: the market price gets significantly decreased when the energy is available. This can be seen as reason for the suspension of the energy production by the gas power plant, see Figure 3. Again, it is well visualized, that abounding energy is transferred to other markets, because the prices in other markets react as well on the increase of available solar energy. In general it can be obtained, that the control of the energy production by pure price strategies is able to balance energy consumption and energy production well enough that the waste caused by overproduction is negligible.

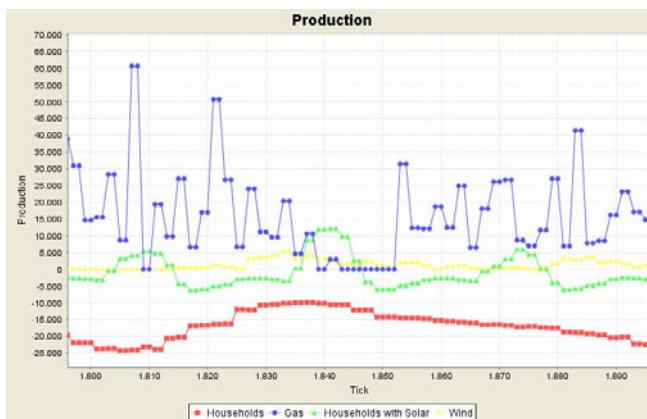


Fig. 3. Response in the curves for increased solar energy production

7 Conclusion

We have demonstrated how to apply principles from distributed computational intelligence to completely remove centralized components from the control mechanisms of a hierarchical grid. The approach is motivated by the physical characteristics of the grid and their participants. If each agent locally maximizes its benefit this leads globally to a robust and self-organizing market. The submarkets have been shown to couple well, concerning the market price and the transformation of energy between the different markets. The simulated market represents a proof-of-concept for the functionality of the proposed mechanisms. Our approach maps principles that have been known for a long time from monolithic markets to all layers of a hierarchically structured grid. Further research in this direction will be necessary.

It is important to point out that the approach can be applied specifically well if further local storage technologies get available but virtual storages (cooling of the freezer, heating of the hot water tank, charging of electric cars, etc.) can be used as well to shift consumption. The agents would always aim to refill these storage facilities as cheap as possible, like in times of power oversupply. On the other hand, they aim to consume stored power instead of buying power in times of high prices, like in times of insufficient supply. Technologies as on-site power stations would be employed more frequently, if available. There are many model details which have to be investigated more closely in future as, e.g., more sophisticated trading strategies for multi-market-trading.

References

1. Barroso, L.A., Rosenblatt, J., Guimaraes, A., Bezerra, B., Pereira, M.V.: Auctions of contracts and energy call options to ensure supply adequacy in the second stage of the Brazilian power sector reform. In: Power Engineering Society General Meeting, pp. 1–8. IEEE, Los Alamitos (2006)
2. Borenstein, S., Jaske, M., Rosenfeld, A.: Dynamic pricing, advanced metering, and demand response in electricity markets. Center for the Study of Energy Markets, UC Berkeley (2002), <http://www.escholarship.org/uc/item/11w8d6m4> (Retrieved from 2002)
3. Chen, X., Xie, J.: Optimal bidding strategies for load server entities in double-sided auction electricity markets with risk management. In: PMAPS 2006, pp. 1–6. IEEE, Los Alamitos (2006)
4. Fulp, E., Reeves, D.: Optimal provisioning and pricing of internet differentiated services in hierarchical markets. In: Lorenz, P. (ed.) ICN 2001. LNCS, vol. 2093, pp. 409–418. Springer, Heidelberg (2001)
5. Hines, P., Apt, J., Talukdar, S.: Large blackouts in north america: Historical trends and policy implications. Energy Policy 37(12), 5249–5259 (2009)
6. Hobbs, B.F., Rothkopf, M.H., Hyde, L.C., O'Neill, R.P.: Evaluation of a truthful revelation auction in the context of energy markets with nonconcave benefits. Journal of Regulatory Economics 18(1), 5–32 (2000)
7. Li, J., Poulton, G., James, G.: Agent-Based Distributed Energy Management. In: Orgun, M.A., Thornton, J. (eds.) AI 2007. LNCS (LNAI), vol. 4830, pp. 569–578. Springer, Heidelberg (2007)

8. Kian Jr., A.R., Cruz, J.B., Thomas, R.J.: Bidding strategies in oligopolistic dynamic electricity double-sided auctions. *IEEE Transactions on Power Systems* 20(1), 50–58 (2005)
9. Kramer, O., Satzger, B., Lässig, J.: Power Prediction in Smart Grids with Evolutionary Local Kernel Regression. In: Graña Romay, M., Corchado, E., García Sebastian, M.T. (eds.) HAIS 2010. LNCS, vol. 6076, pp. 262–269. Springer, Heidelberg (2010)
10. Mantegna, R.N.: Hierarchical structure in financial markets. *The European Physical Journal B-Condensed Matter and Complex Systems* 11(1), 193–197 (1999)
11. NAHB Research Center. Review of residential electrical energy use data, <http://www.toolbase.org/PDF/CaseStudies/ResElectricalEnergyUseData.pdf> (Retrieved from 2001)
12. Nanduri, V., Das, T.K.: A reinforcement learning model to assess market power under auction-based energy pricing. *IEEE Transactions on Power Systems* 22(1), 85–95 (2007)
13. National Renewable Energy Laboratory. Homepage, <http://www.nrel.gov/gis/solar.html> (Retrieved from 2010)
14. National Wind Technology Center. Homepage, <http://www.nrel.gov/wind/integrationdatasets/western/methodology.html> (Retrieved from 2010)
15. Overbye, T.J., Cheng, X., Sun, Y.: A comparison of the ac and dc power flow models for lmp calculations. In: HICSS 2004: Track 2, p. 20047.1. IEEE, Washington, DC, USA (2004)
16. Pipattanasomporn, M., Feroze, H., Rahman, S.: Multi-agent systems in a distributed smart grid: Design and implementation. In: PSCE 2009, pp. 1–8. IEEE, Los Alamitos (2009)
17. Praca, I., Ramos, C., Vale, Z., Cordeiro, M.: Mascem: A Multiagent System That Simulates Competitive Electricity Markets. *IEEE Intelligent Systems* 18, 54–60 (2003)
18. Tsoukalas, L.H., Gao, R.: From smart grids to an energy internet: Assumptions, architectures and requirements. In: DRPT 2008, pp. 94–98. IEEE, Los Alamitos (2008)
19. Varga, L.Z., Jennings, N.R., Cockburn, D.: Integrating intelligent systems into a cooperating community for electricity distribution management. *Expert Systems with Applications* 7(4), 563–579 (1994)
20. Vytelingum, P., Cliff, D., Jennings, N.R.: Strategic bidding in continuous double auctions. *Artificial Intelligence* 172(14), 1700–1729 (2008)
21. Vytelingum, P., Ramchurn, S.D., Voice, T.D., Rogers, A., Jennings, N.R.: Trading agents for the smart electricity grid. In: International Foundation for Autonomous Agents and Multiagent Systems, AAMAS 2010, pp. 897–904 (2010)
22. Vytelingum, P., Voice, T.D., Ramchurn, S.D., Rogers, A., Jennings, N.R.: Agent-based micro-storage management for the smart grid. In: International Foundation for Autonomous Agents and Multiagent Systems, AAMAS 2010, pp. 39–46 (2010)
23. Weron, R.: Energy price risk management. *Physica A: Statistical Mechanics and its Applications* 285(1-2), 127–134 (2000)
24. Xiong, G., Okuma, S., Fujita, H.: Multi-agent based experiments on uniform price and pay-as-bid electricity auction markets. In: DRPT 2004, pp. 72–76. IEEE, Los Alamitos (2004)

Compiling AI Engineering Models for Probabilistic Inference

Paul Maier, Dominik Jain, and Martin Sachenbacher

Department of Informatics, Technische Universität München, Germany
`{maierpa, jain, sachenba}@in.tum.de`

Abstract. In engineering domains, AI decision making is often confronted with problems that lie at the intersection of logic-based and probabilistic reasoning. A typical example is the plan assessment problem studied in this paper, which comprises the identification of possible faults and the computation of remaining success probabilities based on a system model. In addition, AI solutions to such problems need to be tailored towards the needs of engineers. This is being addressed by the recently developed high-level, expressive modeling formalism called probabilistic hierarchical constraint automata (PHCA).

This work introduces a translation from PHCA models to statistical relational models, which enables a wide array of probabilistic reasoning solutions to be leveraged, e.g., by grounding to problem-specific Bayesian networks. We illustrate this approach for the plan assessment problem, and compare it to an alternative logic-based approach that translates the PHCA models to lower-level logic models and computes solutions by enumerating most likely hypotheses. Experimental results on realistic problem instances demonstrate that the probabilistic reasoning approach is a promising alternative to the logic-based approach.

1 Introduction

In engineering domains, AI decision making is often confronted with problems that lie at the intersection of model-based diagnosis and probabilistic reasoning. For example, in a manufacturing scenario, the AI controller of a factory plant needs to take decisions depending on the success or failure of the production of individual products. The manufacturing of products is automatically scheduled in advance, specifying which actions to perform where and when. During production, model-based diagnosis computes most likely diagnostic hypotheses for given observations [1] and thus provides the controller with a global view of the plant. To take the aforementioned decisions, however, it additionally needs a local view indicating, for example, the probability with which the production of a particular item will succeed given the observations. For example, a product might be predicted to be successfully manufactured with probability 0.35. This is the classical probabilistic reasoning problem of computing posterior marginals. The problem of computing most likely hypotheses and, at the same time, the probabilities with which given goals are reached is a variant of the *plan assessment problem* [2].

Model-based diagnosis [3] is a technique that supports AI decision making by identifying faulty components of a technical system, typically based on logical reasoning, while probabilistic reasoning is the classical AI technique for decision support. A key

aspect in applying these methods is modeling. Both model-based diagnosis and probabilistic reasoning follow a trend towards rich and expressive formalisms and auto-generated low-level representations such as Bayesian networks (BNs), which can then be fed to off-the-shelf tools. This has three advantages: 1) It spares users the tedious effort of hand crafting low-level models and re-implementing algorithms, 2) the algorithms implemented by said tools seek to exploit problem structure in a most general fashion and 3) they are typically supported by large communities. Probabilistic hierarchical constraint automata (PHCA) [4] are a first-order model-based diagnosis formalism specifically tailored to compactly represent complex hard- and software behavior, addressing the needs of engineers. Following the approach of using off-the-shelf tools, [5] provides an automatic translation for constraint optimization, which is common in model-based diagnosis. This work was extended in [2] towards plan assessment. However, there is still a gap between high-level modeling on the model-based diagnosis side and the methods and frameworks on the probabilistic reasoning side.

This work aims to provide a way of applying the tools of probabilistic reasoning to the problem of plan assessment. We contribute a translation of first-order PHCA models to statistical relational models, which are a first-order generalisation of graphical models such as Bayesian networks. This opens up the opportunity to choose among a wide range of off-the-shelf tools for probabilistic reasoning. To our knowledge, it is the first such translation from PHCAs to a high-level probabilistic reasoning framework. We evaluate our approach by translating different PHCAs and feeding the resulting BNs to the publicly available probabilistic reasoning tool Ace 2.0 [6,7] in order to solve the plan assessment problem. We compare the results with the aforementioned model-based diagnosis approach.

Closely related is [8], which solves plan assessment for a hand-crafted statistical relational model. While there has been work on combining model-based diagnosis and probabilistic reasoning [9,10], a general bridge between the two areas, such as our translation, which allows the comparison of model-based diagnosis and probabilistic reasoning solutions, has not yet been developed. The field of probabilistic model checking [11], which is close to plan assessment, can answer queries such as “What is the probability that the system reaches state s ?”, based on a system model. However, plan assessment is ultimately geared towards supporting robust online control, which, unlike probabilistic model checking, includes generating diagnoses and focussing computation using available observations.

The remainder of this section details our example scenario. The next three sections formally define plan assessment, recap the model-based diagnosis solution and describe our novel translation along with the probabilistic reasoning solution based upon it. The final sections discuss the evaluation results and conclude the paper.

Example: metal machining and assembly. Within our aforementioned manufacturing scenario, products (toy mazes) are first being machined and then assembled. Two different faults can flaw products. The cutter of machining stations might break, damaging the alloy base plate (see Fig. 1d). This damage triggers an observable alarm at the assembly station later on. That same alarm might also be triggered if the assembly station is miscalibrated (the second fault). The question is: Given the alarm, how are the manufacturing goals affected, and what are potential faults? We modeled several slight

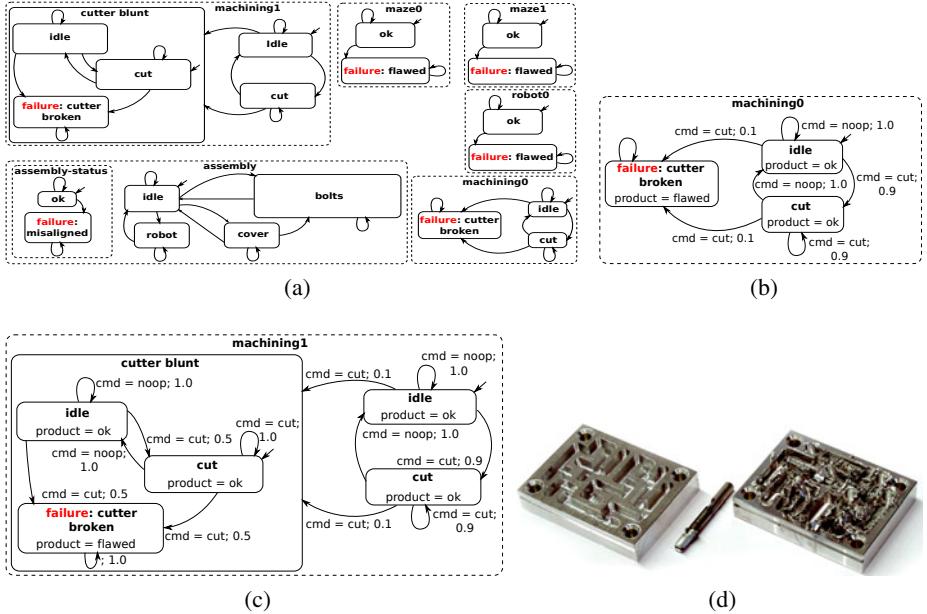


Fig. 1. (a) A PHCA modeling a factory scenario with three products

variations of this scenario. Fig. 1a shows a PHCA that models an assembly and two machining stations, as well as three products which are to be produced. Fig. 1b and 1c show detailed versions of the two machining stations.

2 PHCA-Based Plan Assessment

Plan assessment extends the maximum probability diagnosis problem [12] towards additionally computing success probabilities for given goals. The problem arises when three aspects come together: 1) A rigidly designed system with some remaining uncertainties, which would be hard to eliminate, 2) executes pre-planned operations to 3) achieve defined goals. These aspects are reflected in three formal elements: 1) A system model, in our case a PHCA M_{PHCA} , 2) an operation sequence \mathcal{S} of operation steps, and 3) a set of goals $\{\mathcal{G}_i\}$.

System Model. PHCA models define automata states, called *locations*, and transitions between them. Locations may be composed of sub-locations and transitions may be guarded and probabilistic. PHCAs are formally defined as follows [4]:

Definition 1. A PHCA is a tuple $(\Sigma, P_\Xi, \Pi, O, Cmd, \mathcal{C}, P_T)$:

- $\Sigma = \Sigma_p \uplus \Sigma_c$ is a set of locations, partitioned into primitive locations Σ_p and composite locations Σ_c , where a composite location represents a PHCA whose elements are subsets of the elements of the containing PHCA. A location may be marked or unmarked. A marked location represents an active execution branch. A marking m^t (at time t during execution) is given as a subset of Σ .

- $P_{\Xi}(\Xi_i)$ denotes the probability that $\Xi_i \subseteq \Sigma$ is the set of start locations (initial state).
- $\Pi = O \uplus Cmd \uplus Dep$ is a set of variables with finite domains. O is the set of observation variables, Cmd is the set of action or command variables, and Dep is a set of dependent hidden variables which are connected to other variables via constraints.
- C is the set of finite domain constraints defined over Π , which comprises behavior constraints of locations and guard constraints of transitions. A location's behavior constraint serves to define the observations that are consistent with the location; a transition's guard defines the conditions under which the transition can be taken (usually depending on commands).
- $P_T[l_i]$ (defined for each $l_i \in \Sigma_p$) is a probability distribution over the subset of the set of transitions T , which contains transitions leading away from l_i whose guards (elements of C) are satisfied given the current state.

Given a PHCA model M_{PHCA} of a technical system, the behavior of the system over time is estimated by generating sequences of *location markings* $\theta = (m^{t_0}, \dots, m^{t_N})$. $St(M_{\text{PHCA}})$ denotes the set of all such fixed-length sequences (called *trajectories*).

Operation Sequences. Pre-planned operations are typically given as a sequence \mathcal{S} of operation steps. Steps can be anything from simple (sets of) commands to more complicated operations, such as scheduled allocations of machines, products and actions to perform. We consider scenarios where \mathcal{S} is synthesized automatically (using, e.g., a planner or a scheduler), and later executed on the system. This is captured by an *execution adaptation function* $\mathcal{E}_{\mathcal{S}}$, which adapts a PHCA model M_{PHCA}^N unfolded for N time steps (reproducing the PHCA's components for each time step t , yielding Σ^t , Π^t , T^t and C^t). It sets all command variables as given by \mathcal{S} , removing the transitions whose guards become unsatisfiable as a result, and modifies the constraints C^t . The resulting model $M_{\text{PHCA}}^{\mathcal{S}} = \mathcal{E}_{\mathcal{S}}(M_{\text{PHCA}}^N)$ therefore no longer contains the command variables Cmd and will typically feature a reduced number of possible trajectories. Markov properties of M_{PHCA} are unaffected.

Each sequence of observations $\mathbf{o}^{0:t}$ ($t \leq N$) and each model $M_{\text{PHCA}}^{\mathcal{S}}$ defines a joint distribution $P(\theta, \mathbf{O} = \mathbf{o}^{0:t})$:

$$P(\theta, \mathbf{O}^{0:t} = \mathbf{o}^{0:t}) = P_{\Xi}(m^0) \prod_{u \in \{0..t\}} P(\mathbf{O}^u | m^u) \prod_{\tau \in T[\theta]} P(\tau) \quad (1)$$

where $T[\theta]$ is the multiset of all transitions between primitive locations as implied by θ , in which a transition τ from location l_i to l_j may occur multiple times; the transition probability is computed as $P(\tau) = P_T[l_i](\tau)$. The above distribution corresponds to the PHCA hidden Markov model semantics [54].

Goals. A goal is a tuple (l, t) of a location l that should be marked at time t . $\mathcal{G}_i := \{(m^{t_0}, \dots, m^{t_N}) \mid l \in m^t\}$ denotes the set of goal-achieving trajectories, i.e. which lead to the marking required by the goal.

Definition 2. Let M_{PHCA} be a PHCA model, \mathcal{S} a sequence of N operation steps with execution adaptation function $\mathcal{E}_{\mathcal{S}}$ and $\{\mathcal{G}_i\}$ a set of goals. Let $M_{\text{PHCA}}^{\mathcal{S}} = \mathcal{E}_{\mathcal{S}}(M_{\text{PHCA}}^N)$.

The **plan assessment problem** is, given observations $\mathbf{o}^{0:t}$, to compute the most probable diagnosis as trajectory in $St(M_{\text{PHCA}}^S)$ as well as $P(\mathcal{G}_i \mid \mathbf{o}^{0:t})$ for each i .

For example, a goal might be that the maze is ok by the time it's expected to be finished: (maze1.ok, 6). An operation sequence then specifies, among other things, steps to cut the alloy of the maze, i.e. $(\dots, (\text{maze0}, \text{machining0}, \text{cut}, 1), \dots)$. A resulting diagnosis, in form of the most probable trajectory, might indicate a broken cutter and as a result a flawed maze: $(\dots, m^t = \{\dots, \text{machining0.failure}, \text{maze1.failure}, \dots\}, \dots)$.

2.1 Solving Plan Assessment

For PHCA, no solver for plan assessment exists, therefore we chose to translate them such that existing solvers can be applied. Model representation strongly influences problem solving efficiency. Often, Markov modelling is used to exploit the Markov property during inference. However, compared to more expressive non-Markov modeling this can lead to bigger representations. This size vs. expressiveness trade-off matters even more when automatically generating such representations from high-level models. Originally, PHCA semantics were defined in terms of hidden Markov models (HMM) [4], which allows complex Markov modeling at the cost of potentially larger models. This is problematic for existing off-the-shelf HMM solvers: On the one hand, a naive automated flattening of hierarchical models would generate prohibitively large HMMs. Automatically decomposing PHCAs into explicit, tractable HMM representations, on the other hand, is only possible in a small number of special cases where components are not connected. In our example this is not the case: The assembly and manufacturing stations are connected via product models and their scheduled actions. In addition, automatic compilation typically incurs an overhead compared to custom tailored translation, which further increases the size of explicit HMMs. Consequently, it is not usually advisable to encode ground models as explicit HMMs: We would easily obtain hidden variables with domain sizes of $\approx 2^{100}$, thus requiring a $2^{100} \times 2^{100}$ transition matrix!

For these reasons, we apply a different approach. Using a translation Υ we translate, in an offline step, M_{PHCA} to low-level representations M that are still expressive enough to represent structure and are sufficiently general to allow the use of off-the-shelf solvers that do not depend on the Markov property but exploit model structure in a general fashion. We define execution adaptation functions \mathcal{E}_S on these representations, which allows to reuse the translation for different operation sequences. We now describe two solutions for plan assessment that we evaluate and compare in this work.

3 Hypothesis Enumeration

Recent work developed a model-based diagnosis-inspired solution that first computes the most likely trajectories as the best solutions to a constraint optimization problem (COP) and then computes $P(\mathcal{G}_i \mid \mathbf{o}^{0:t})$ by summing over goal-achieving trajectories among these, normalizing over all of them [2]. The work exploits an automatic translation of a PHCA to a COP $\mathcal{R} = (X, D, C)$ [5]. We term this translation Υ_{COP} , which maps a model M_{PHCA} to variables X , their finite domains D and local objective functions $c \in C$, called soft constraints. Soft constraints map partial variable assignments to $[0, 1]$. Execution functions modify \mathcal{R} by adding soft constraints. We refer to [5] for the details.

Plan assessment is solved for a translated and adapted model $\mathcal{E}_S(\Upsilon_{\text{COP}}(M_{\text{PHCA}}))$ as follows. Each COP solution of the translation corresponds to a trajectory θ of M_{PHCA} . A COP solution is an assignment to all variables in X . $P(\mathcal{G}_i \mid \mathbf{o}^{0:t})$ can be computed as $P(\mathcal{G}_i \mid \mathbf{o}^{0:t}) = \sum_{\theta \in \mathcal{G}_i} P(\theta \mid \mathbf{o}^{0:t}) = \frac{\sum_{\theta \in \mathcal{G}_i} P(\theta, \mathbf{o}^{0:t})}{\sum_{\theta \in \Theta} P(\theta, \mathbf{o}^{0:t})}$, i.e. $P(\mathcal{G}_i \mid \mathbf{o}^{0:t})$ can be computed exactly based on probabilities $P(\theta, \mathbf{o}^{0:t})$, which are retrieved from objective values of the COP solutions.

The key motivation for this approach is that powerful off-the-shelf constraint optimization solvers can be exploited to generate a list of all non-zero probability assignments, the hypotheses, sorted by probability. The most probable one is then chosen as diagnosis, while the complete list is used as set Θ . While [2] used the Toolbar solver, in this work, we use the more recent, award-winning Toulbar2 solver¹, which implements many state-of-the-art techniques (including, for example, soft-constraint consistency).

Note that Toulbar2 requires another transcoding that involves conversion between probability values and integer costs, resulting in a precision loss. For our examples, however, this loss was negligible.

4 Probabilistic Inference

From a probabilistic reasoning point of view, plan assessment can be seen as a combination of simultaneously computing most likely a posteriori hypotheses (MAP) and marginals. To solve these standard problems, probabilistic models are commonly represented as Bayesian networks (BNs). We now describe a new translation of PHCA models to abstract, generalized Bayesian networks, which can in turn be automatically instantiated to BNs. This opens up the possibility of comparing the COP-based approach from the previous section with solutions from the probabilistic reasoning community. Once translated, $P(\mathcal{G}_i \mid \mathbf{o}^{0:t})$ can be computed as the posterior marginal of a BN variable.

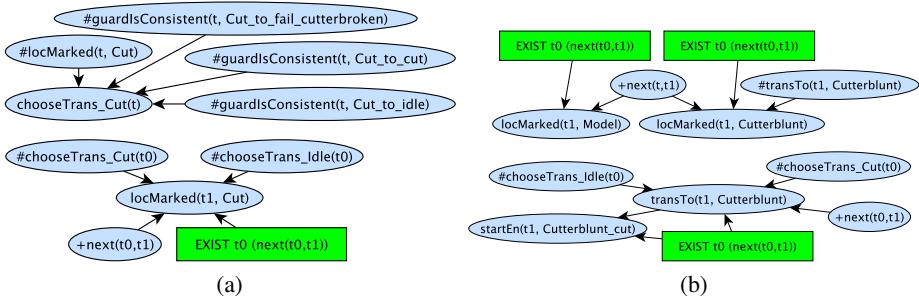


Fig. 2. Figures 1b and 1c translated to BLN fragments, showing excerpts of location marking and probabilistic transition choice (2a) as well as composite and full target marking (2b)

We translate PHCAs into Bayesian logic networks (BLNs) [13]. A BLN is a statistical relational model which can be viewed as a template for the construction of Bayesian

¹ <https://mulcyber.toulouse.inra.fr/projects/toulbar2> (02/2011).

networks. Given a set of entities, a BLN may be instantiated to either a ground BN or a ground mixed network (MN) that explicitly represents logical constraints on the distribution [14]. Probabilistic inference is often performed in such ground models. However, recent research in statistical relational reasoning tries to (partially) lift the inference problem to the first-order level in order to exploit repeated sub-structures in ground models [15]. Besides the added expressivity and flexibility offered by a first-order formalism, the possibility of exploiting first-order encodings is one of our main reasons for seeking a relational translation rather than a direct encoding to BNs.

Key elements of a BLN M_B are abstract random variables (ARVs), typed entities, probability distribution fragments and first-order logical (FOL) formulas. ARVs are parametrized random variables that correspond to either predicates or non-boolean functions. They encode, for example, partial states such as a station being faulty, relations such as stations working on specific products or a probabilistically chosen transition. The arguments of an ARV refer to abstract, typed entities, which allows, in particular, quantification over time. A fragment associates an ARV with a conditional probability table (CPT); the set of fragments (ellipses in Fig. 2) collectively defines, for a set of ground instances of ARVs, a probability distribution. The applicability of a fragment may be restricted by (mutually exclusive) first-order logic preconditions (boxes in Fig. 2). Finally, hard logical formulas may be specified in the model, restricting the set of possible worlds. When instantiating a BLN to a BN, a grounding (X, D, G, P) is created that consists of random variables X , their corresponding set of domains D , a graph G connecting variables according to parent-child condition relations defined via the fragments, and the set of CPTs P .

For our novel translation, we adapted the COP encoding γ_{COP} of PHCAs. This encoding is defined in terms of formal higher-order rules for structure, probabilistic behavior and consistency with observations and commands [5]. The translation to BLNs largely follows these rules, however often exploits first-order modeling features of BLNs. The translation function γ_{BLN} takes as input a model M_{PHCA} and creates a BLN $M_B = (\mathcal{D}, \mathcal{F}, \mathcal{L})$ and a knowledge base DB. M_B consists of the fundamental declarations \mathcal{D} , the set of fragments \mathcal{F} and the set of FOL formulas \mathcal{L} . The knowledge base defines existing objects or entities for the FOL formulas and fragments as well as known facts about relations among these entities. When the BLN is grounded, DB is extended with further evidence. Execution adaptation functions \mathcal{E}_S for BLNs add formulas to \mathcal{L} and facts to DB. \mathcal{D} contains, among other things, the entity types and predicate signatures for ARVs. Most of \mathcal{D} is model-independent, i.e. stays the same for arbitrary M_{PHCA} . Formulas in \mathcal{L} are, in particular, used to define behavior and transition guard consistency predicates in terms of formulas over assignments of PHCA variables O and Cmd . An example of a concrete behavior constraint is `behavConsistent(t, Cutter_broken) <=> var_PRODUCT(t, Faulty)`. It specifies that the behavior of location “broken” (which is part of composite location “cutter”) is consistent if and only if the product being processed will be broken in the next time step. In addition, \mathcal{L} contains the general **behavior consistency** rule: `locMarked(t, l) => behavConsistent(t, l)`. It says that for all points in time, a location’s behavior must be consistent if it is marked.

Next, we look at target marking, i.e. the marking of locations that are enabled start locations or that are transitioned to. In general, the predicate $\text{locMarked}(t, l)$ (abbreviated as $\text{lm}(t, l)$) encodes a location l being marked at time t . We first treat the marking of composite locations. The **composite target marking** rule marks a composite location if it is transitioned to or if it is an enabled start location:

$$\forall t \in \{1..N\}. \forall l_c \in \Sigma_c. \text{transTo}(t, l_c) \vee \text{startEnabled}(t, l_c) \Rightarrow \text{lm}(t, l_c)$$

The transTo predicate is defined as follows,

$$\begin{aligned} \forall t_0 \in \{0..N-1\}, \forall t_1 \in \{1..N\}. \forall l_c \in \Sigma_c. \text{next}(t_0, t_1) \Rightarrow \\ (\text{transTo}(t_1, l_c) \Leftrightarrow \exists l \in \text{parents}(l_c). \text{target}(\text{chooseTrans}(t_0, l)) = l_c) \end{aligned}$$

where the function target maps transitions to their target locations and parents maps a location to the set of locations connected to it via transitions.

Using the location *Cutterblunt* as an example, we show how the composite marking rule is translated into fragments. Generally, one fragment is created for each predicate occurring in a rule, except if the translator can determine, e.g. from the model structure, that a predicate is always True or False. Predicates are partially instantiated, removing all quantification except over time. In case of *Cutterblunt*, fragments for partially instantiated predicates $\text{transTo}(t, \text{Cutterblunt})$ and $\text{lm}(t, \text{Cutterblunt})$ are created. No fragment is created for $\text{startEnabled}(t, \text{Cutterblunt})$ because *Cutterblunt* is not a start location and the predicate is, therefore, always False. The following table shows the CPT template for $\text{lm}(t, \text{Cutterblunt})$:

| | | |
|---|-----|-----|
| $\text{transTo}(t, \text{Cutterblunt})$ | T | F |
| $\text{lm}(t, \text{Cutterblunt}) = T$ | 1 | 0.5 |
| $\text{lm}(t, \text{Cutterblunt}) = F$ | 0 | 0.5 |

The CPT encodes that *Cutterblunt* is marked if it is being transitioned to. If not, the CPT doesn't influence the marking. The fragment encoding $\text{next}()$ is evaluated already during translation: if True, the CPT is instantiated, if False (i.e. no prior time point exists), it is omitted. This allows to create a more compact CPT without this fragment as parent. See Fig. 2b for the partial fragment network.

The second rule that influences the marking of composite locations is the **hierarchical marking/unmarking** rule, which ensures that a composite location is marked iff at least one of its sub-locations (which are given by function sub) is marked:

$$\forall t \in \{0..N\}. \forall l_c \in \Sigma_c. \text{lm}(t, l_c) \Leftrightarrow \exists l_p \in \text{sub}(l_c). \text{lm}(t, l_p)$$

This rule can be directly translated using logical formulas that we can add to \mathcal{L} . For example, the following rule is added for the composite location *Cutterblunt*:

```
locMarked(t, Cutterblunt) <=> locMarked(t, Cutterblunt_idle) v
locMarked(t, Cutterblunt_cut) v locMarked(t, Cutterblunt_broken)
```

The **primitive target marking** rule marks primitive locations if they are either transitioned to or if they are enabled starting locations:

$$\begin{aligned} \forall t_0 \in \{0..N-1\}. \forall t_1 \in \{1..N\}. \forall l_p \in \Sigma_p. \exists l \in \text{parents}(l_p). \text{next}(t_0, t_1) \Rightarrow \\ (\text{target}(\text{chooseTrans}(t_0, l)) = l_p \vee \text{startEnabled}(t_1, l_p) \Leftrightarrow \text{lm}(t_1, l_p)) \end{aligned}$$

The **full target marking** rule ensures that all start sub-locations of a composite location are enabled iff this composite location is the target of a chosen transition or is itself enabled:

$$\forall t \in \{1..N\}. \forall l_c \in \Sigma_c. transTo(t, l_c) \vee startEnabled(t, l_c) \Leftrightarrow \forall l \in sub(l_c). startEnabled(t, l)$$

These two rules are handled analogously to the composite marking rule.

The central rule for probabilistic behavior is **probabilistic transition choice**. Given a primitive location, exactly one of its outgoing transitions may be chosen (according to transition probabilities defined in the model) iff the location is marked and the chosen transition's guard is consistent:

$$\forall t \in \{0..N\}. \forall l_p \in \Sigma_p. \exists \tau \in outgoing(l_p) \cup \{NoTrans\}. lm(t, l_p) \wedge guardIsConsistent(t, \tau) \Leftrightarrow chooseTrans(t, l_p) = \tau \wedge \tau \neq NoTrans$$

In this formula, the function $chooseTrans(t, l_p)$ maps time and location to an admissible outgoing transition. The translation creates BLN functions of time only, eliminating quantification over l_p (see, e.g., Fig. 2a and 2b). Their CPTs define the following probability function:

$$Pr(T_{l_p}^t = \tau | L_p^t, \mathbf{G}^t) = \begin{cases} p_\tau & \text{if (a)} \\ 1 & \text{if (b)} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where $T_{l_p}^t$ is a random variable for choosing among l_p 's outgoing transitions, L_p^t encodes $lm(t, l_p)$ and \mathbf{G}^t is a vector of random variables encoding $guardIsConsistent(t, \tau)$ for each outgoing transition. Condition (a) is $lm(t, l_p) \wedge guardIsConsistent(t, \tau)$ and (b) is $(\neg lm(t, l_p) \vee (\neg \exists \tau' \in outgoing(l_p). guardIsConsistent(t, \tau')) \wedge \tau = NoTrans)$. Note that this fragment also encodes **guard consistency**, which is more compact than having a separate logical rule as for behavior consistency.

We now address the issue of incorporating the constraints added by \mathcal{E}_S . In our implementation, S is a schedule, i.e. a sequence of tuples $\langle(p, c, t, a)\rangle_j$, each defining an action a to perform on a product p at time t on a station or component c . We call these tuples *product-component links*. We can exploit the flexibility of the BLN framework to encode these: If, at time t , a product p is worked by station c , logical formulas are added to \mathcal{L} that enforce equality of variables $\{X_p^t\}$ and $\{X_c^t\}$ for products and stations respectively. The common domain of two variables X_p^t and X_c^t encodes influences, e.g. *Faulty* for c inflicting damage on p or a faulty p causing unusual observations in c . *Ok* encodes no (harmful) influence. Commands given in S are simply added as facts to DB.

So far, we did not treat the rules for the initial time point t_0 which determine the locations that are initially marked/unmarked. Special t_0 rules encode the marking and unmarking as given by the start distribution $P_{\Xi}(\Xi_i)$ and handle special conditions for hierarchical marking, e.g. the requirement that at time t_0 , locations cannot be transitioned to. For more details we refer to [5].

4.1 Translation Correctness

We say that the translation is correct iff the BN given by translating M_{PHCA} to a BLN and grounding it, encodes the same distribution over variables L_i^t (that encode location markings at time points t) as PHCA distribution $P(\theta, \mathbf{O} = \mathbf{o}^{0:t})$. Variable L_i^t being True corresponds to location l_i being marked, i.e. $l_i \in m^t$.

Theorem 1. Let $M_{\mathcal{B}} = (\mathcal{D}, \mathcal{F}, \mathcal{L})$ be a BLN generated with the above described translation process from a PHCA model: $M_{\mathcal{B}} = \Upsilon_{\text{BLN}}(M_{\text{PHCA}})$. Let \mathcal{E}_{BLN} , $\mathcal{E}_{\text{PHCA}}$ be the execution adaptation functions for an arbitrary operation sequence, BN (X, D, G, P) the grounding of $\mathcal{E}_{\text{BLN}}(M_{\mathcal{B}})$, and $\theta = (m^{t_0}, m^{t_1}, \dots, m^{t_N})$ an arbitrary trajectory of the adapted PHCA $\mathcal{E}_{\text{PHCA}}(M_{\text{PHCA}})$. Then

$$\begin{aligned} P_{\Xi}(m^0) \prod_{u \in \{0..t\}} P(\mathbf{O}^u | m^u) \prod_{\tau \in T[\theta]} P(\tau) = \\ P(\mathbf{L}^{t_0} = \mathbf{m}^{t_0}, \dots, \mathbf{L}^{t_N} = \mathbf{m}^{t_N}, \mathbf{O}_{\text{BN}} = \mathbf{o}^{0:t} | X_{\text{aux}} = \text{True}) \end{aligned} \quad (3)$$

\mathbf{L}^{t_j} are vectors of location marking variables $L_i^{t_j}$ in the BN for each time point t_j , and \mathbf{O}_{BN} is a vector of observation variables $O_l^{t_j}$ for each time point t_j (l ranges over indices of observation variables for a given time point). X_{aux} is a set of auxiliarly variables used to represent additional logical constraints. \mathbf{m}^t is boolean vector that encodes a marking in terms of L_i^t assignments for time t .

4.2 Solving Plan Assessment with Probabilistic Reasoning Tools and Methods

Translating and executing a given PHCA, $\mathcal{E}_{\mathcal{S}}(\Upsilon_{\text{BLN}}(M_{\text{PHCA}}))$, yields a BLN as a starting point for possible probabilistic reasoning solutions for plan assessment. In our experiments, we ground the BLNs to auxiliary BNs and use the state-of-the-art inference tool Ace 2.0², which compiles a given BN into an arithmetic circuit (AC) [6][7]. Ace exploits local structure given by, e.g., determinism in the model, to achieve very compact ACs. Once an AC is given, marginals, and thus $P(\mathcal{G}_i | \mathbf{o}^{0:t})$, can be computed online in time linear in the size of the AC. Note that we focus on computing $P(\mathcal{G}_i | \mathbf{o}^{0:t})$ in this work, while, in general, solving plan assessment also requires computing the most probable trajectory (as a diagnosis). This can be done by computing the most probable explanation.

The Ace compilation is considered a (potentially expensive) offline step, the evaluation the (quick) online step. The addition of evidence, i.e. clamping variables to given values, is part of the latter. This step is independent of translation, execution adaptation, grounding, and Ace compilation, which means we can perform all of these steps offline without introducing evidence too early. Ace compilation can only be done offline if the execution adaptation is done offline, because the latter modifies the model. Real-world applications, however, might require execution adaptation to be performed online. Because the size of \mathcal{S} might be too large to allow reasoning over the complete time horizon, receding horizon schemes such as [16] may be applied: the model would be translated for a fixed number N of time steps, then iteratively moved along the operation sequence with potentially much larger length $N_S \gg N$. Consequently, $\mathcal{E}_{\mathcal{S}}$ must

² <http://reasoning.cs.ucla.edu/ace/> (03/2011).

Table 1. (Left) The size of PHCAs, COP and BN translations. (Right) Measurements for Toulbar2 and Ace solving the instances. * Results from Ace 3.0.

| instance | N | phca size | # var | # con | # nodes | instance | Toulbar2 | Ace |
|--------------------------|-----|-----------|-------|-------|---------|--------------------------|-------------------|-------------------------------|
| fm1 [2] | 6 | 11/6/27 | 643 | 670 | 1106 | fm1 [2] | 0.01 / 8 / 186 | 0.37 + 0.09 / 10 |
| fm2 | 9 | 15/8/33 | 1202 | 1251 | 2122 | fm2 | 0.05 / 10 / 1878 | 0.93 + 0.18 / 90 |
| fm3 | 9 | 17/8/33 | 1305 | 1311 | 2292 | fm3 | 0.32 / 10 / 5464 | 0.36 + 0.07 / 5 |
| fm2(long \mathcal{S}) | 19 | 15/8/33 | 2482 | 2601 | 4444 | fm2(long \mathcal{S}) | 0.65 / 19 / 11320 | 1128.04 + ERR / ≈ 900 |
| fm3(long \mathcal{S}) | 33 | 18/8/35 | 4748 | 4892 | 8878 | fm3(long \mathcal{S}) | 2.47 / 39 / 11468 | (1.34 + 0.16 / -)* |
| sm [5] | 8 | 8/4/22 | 640 | 661 | 1080 | sm [5] | 0.01 / 9 / 176 | 0.87 + 0.03 / 187 |

be applied for every iteration. It is nontrivial to redefine $\mathcal{E}_{\mathcal{S}}$ such that it could be applied online to an Ace compiled model.

A different class of approximate probabilistic reasoning algorithms for tasks such as the computation of marginals is sampling. Sampling is attractive, since it is an anytime approach with a stochastic accuracy guarantee for marginals in terms of the confidence interval [8]. Models with strong determinism, such as ours, pose practical problems owing to the rejection problem. However, schemes such as SampleSearch [17] have recently been proposed to address this problem by exploiting constraint-solving methods to quickly exclude inconsistent samples.

5 Evaluation

Experiments. We ran experiments on six different problem instances, five plan assessment instances based on factory models and one diagnosis instance based on a satellite camera model from [5]. The factory models 2 and 3 are variations of the model shown in [1], factory model 1 is taken from [2]. All models contain one assembly and one or two machining stations. Factory models 1, 2 and 3 model one, two and three products respectively. There are some other, minor differences regarding, e.g., the sensors. Factory models 2 and 3 are used for two scenarios each, one with a short and one with a longer operation sequence \mathcal{S} . Finally, the diagnosis instance simulates diagnosing hardware or software faults in a satellite camera module [5]. Tab. I lists the size statistics for the instances: the number of time steps N , the PHCA size (primitive loc./composite loc./no. transitions), the number of variables and constraints in the COP translation, and the number of nodes in the BN model obtained through the BLN translation. In the following we denote the six instances by fm1, fm2, fm3, fm2(long \mathcal{S}), fm3(long \mathcal{S}) and sm, where we abbreviate “factory model i ” with “fmi” and “satellite model” with “sm”. We used a virtual machine with 2GB of memory, one core of an intel Core 2 Duo and Ubuntu Linux. For all scenarios, we computed the exact results for $P(\mathcal{G}_i \mid \mathbf{o}^{0:t})$ using both Toulbar2 and Ace 2.0. (For the diagnosis instance we defined the goal as a certain location of interest being marked). We used default options for both tools except for Ace compilation of fm2(long \mathcal{S}), where enforcing logical model counting yielded much better results. The results table shows, for Toulbar2, search time (seconds), memory usage (MB) and expanded search tree nodes, and, for Ace, compilation + evaluation time (both in seconds) and memory usage during compilation (in MB).

Results and Discussion. When taking into account that Ace compilation must potentially be done online (as explained in the previous section), Toulbar2 outperforms Ace for most of our instances in runtime (e.g., fm2(long \mathcal{S})) or memory usage (e.g., sm). Still, Ace performs well for four instances out of six, yielding exact results within 2 seconds, even including compilation. The two bigger instances with long \mathcal{S} failed due to precision loss (indicated by “ERR”). With the not yet publicly available Ace 3.0, results for fm3(long \mathcal{S}) could be obtained, but not for fm2(long \mathcal{S}) within the 2GB memory limit. Interestingly, fm2 and fm2(long \mathcal{S}) are much more costly for Ace (compilation) than the *bigger* instances fm3 and fm3(long \mathcal{S}). Toulbar2 and Ace 2.0 differed slightly ($\triangle < 0.0001$) in their exact results, most likely due to precision loss or Toulbar2 using a lower bound to cut off solutions.

6 Conclusion

In this work, we formalized and described the first automatic translation from the high-level model-based diagnosis framework probabilistic hierarchical constraint automata (PHCA) to a high-level probabilistic reasoning framework, Bayesian logic networks (BLNs). It provides engineers with a way of applying a wide range of probabilistic reasoning methods and tools to problems such as the *plan assessment* problem, which occurs in AI decision making in engineering domains. It lies at the intersection of model-based diagnosis and probabilistic reasoning and involves the computation of the probabilities with which pre-planned operations for a technical system achieve their goals. We evaluate our approach by solving six realistic problem instances with a) an existing solution based on generating most probable hypotheses as the best solutions of a constraint optimization problem, generated by an off-the-shelf solver, and b) a novel solution based on our translation, which employs the Ace 2.0 probabilistic reasoning solver. The results demonstrate that probabilistic reasoning tools such as Ace provide a strong alternative for solving plan assessment. Future work will fully exploit our novel translation by evaluating more probabilistic reasoning solutions, such as the more recent Ace 3.0, lifted inference and sampling methods. In particular, we found combinations of sampling with state-of-the-art constraint solvers to be a promising future direction.

References

1. Kurien, J., Nayak, P.P.: Back to the Future for Consistency-Based Trajectory Tracking. In: Proc. AAAI, pp. 370–377. AAAI Press, Menlo Park (2000)
2. Maier, P., Sachenbacher, M., Rühr, T., Kuhn, L.: Automated Plan Assessment in Cognitive Manufacturing. Adv. Eng. Informat. (May 2010)
3. Hamscher, W., Console, L., de Kleer, J. (eds.): Readings in Model-Based Diagnosis. Morgan Kaufmann Publishers Inc., San Francisco (1992)
4. Williams, B.C., Chung, S., Gupta, V.: Mode Estimation of Model-based Programs: Monitoring Systems with Complex Behavior. In: Proc. IJCAI, pp. 579–590 (2001)
5. Mikaelian, T., Williams, C.B., Sachenbacher, M.: Model-based Monitoring and Diagnosis of Systems with Software-Extended Behavior. In: Proc. AAAI. AAAI Press, Pittsburgh (2005)
6. Chavira, M., Darwiche, A.: Compiling Bayesian Networks with Local Structure. In: Proc. IJCAI, pp. 1306–1312 (2005)

7. Darwiche, A.: A Differential Approach to Inference in Bayesian Networks. *Journal of the ACM* 50(3), 280–305 (2003)
8. Maier, P., Jain, D., Waldherr, S., Sachenbacher, M.: Plan Assessment for Autonomous Manufacturing as Bayesian Inference. In: Dillmann, R., Beyerer, J., Hanebeck, U.D., Schultz, T. (eds.) KI 2010. LNCS (LNAI), vol. 6359, pp. 263–271. Springer, Heidelberg (2010)
9. Abreu, R., Zoeteweij, P., Van Gemund, A.J.C.: A New Bayesian Approach to Multiple Intermittent Fault Diagnosis. In: Proc. IJCAI, pp. 653–658. Morgan Kaufmann Publishers Inc., San Francisco (2009)
10. Knox, B., Mengshoel, O.: Diagnosis and Reconfiguration using Bayesian Networks: An Electrical Power System Case Study. In: Workshop Proc. SAS (2009)
11. Rutten, J., Kwiatkowska, M., Norman, G., Parker, D.: Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems. CRM Monograph Series, vol. 23. American Mathematical Society, Providence (2004)
12. Sachenbacher, M., Williams, B.: Diagnosis as Semiring-based Constraint Optimization. In: Proc. ECAI 2004. Valencia, Spain (2004)
13. Jain, D., Waldherr, S., Beetz, M.: Bayesian Logic Networks. Technical report, Technische Universität München (2009)
14. Mateescu, R., Dechter, R.: Mixed Deterministic and Probabilistic Networks. *Annals of Mathematics and Artificial Intelligence* 54(1), 3–51 (2008)
15. Poole, D.: First-Order Probabilistic Inference. In: IJCAI, pp. 985–991 (2003)
16. Maier, P., Sachenbacher, M.: Receding Time Horizon Self-Tracking and Assessment for Autonomous Manufacturing. In: Workshop Proc. Self-X (September 2010)
17. Gogate, V., Dechter, R.: SampleSearch: A Scheme that Searches for Consistent Samples. In: Proc. AISTATS (2007)

Smooth Conditional Transition Paths in Dynamical Gaussian Networks

Michał Matuszak¹, Jacek Miękisz², and Tomasz Schreiber^{1,*}

¹ Faculty of Mathematics and Computer Science, Nicolaus Copernicus University,
Chopina 12/18, 87–100 Toruń, Poland
[{gruby, tomeks}@mat.umk.pl](mailto:{gruby,tomeks}@mat.umk.pl)

² Institute of Applied Mathematics and Mechanics, University of Warsaw,
Banacha 2, 02–097 Warsaw, Poland
miekisz@mimuw.edu.pl

Abstract. We propose an algorithm for determining optimal transition paths between given configurations of systems consisting of many objects. It is based on the Principle of Least Action and variational equations for Freidlin–Wentzell action functionals in Gaussian networks set-up. We use our method to construct a system controlling motion and redeployment between unit’s formations. Another application of the algorithm allows a realistic transformation between two sequences of character animations in a virtual environment. The efficiency of the algorithm has been evaluated in a simple sandbox environment implemented with the use of the NVIDIA CUDA technology.

Keywords: Formation Redeployment, Animation Blending, Transition Path, Reconfiguration, CUDA.

1 Introduction

Simulations of moving groups of agents that preserve their motoric characterizations play an important role across a broad spectrum of applications. Observation of biological systems initiated works on coordination among multiple agents. In a pioneering work [15], a computer model was constructed for synchronized animal motion observed for example in bird flocks or fish schools. It is important to emphasize that motion of individual units was calculated only on the basis of their local environment. In military applications [1], formations allow for a more effective use of limited resources, such as sensors, by division of the environment into portions so each formation’s member can focus attention on an assigned segment while the rest is covered by the partners. This mechanism is used for example by groups of fighter pilots to optimize the usage of their radars and visual perception. Such an approach can also be applied to spacecrafts in a deep space or in the Earth orbit, see survey papers [20, 21] for a comprehensive description. We focus our attention on the problem of a reconfiguration of formations.

* Deceased author (1975 – 2010).

Our method can be used to reduce casualties from a hostile fire, to present less vulnerable targets or to evacuate from an exposed area [22]. Other applications involve parking systems that smoothly drive cars in and out from a parking lot. Finally, the entertainment industry may use our algorithm in real-time computer strategies.

The character animation is undoubtedly an element of the computer graphics, which is of a great importance for enjoyable computer games, credible CG movies or medical visualizations. One of the most popular techniques for generating realistic motion in virtual environments is a skeletal animation technique [4]. Skeletal systems are hierarchical in nature and provide the artist with a control of the character in an efficient manner. An animator can focus his attention on the motion of a simplified structure (the skeleton) rather than manually alter the geometry (character's meshes) itself. For smooth animations it is crucial to generate smooth transitions between system's configurations. Such transitions can be generated by mixing existing ones. One of the available methods is an interpolation of motion data, which has been shown to be a powerful technique if changes between interpolated classes meet predefined/given constraints [18]. The algorithm presented below attempts to address that drawback and provides methodology to produce optimal transition paths between arbitrary configurations.

One of the approaches to describe motion of systems of interacting particles is based on a variational principle. It is assumed in classical mechanics that the trajectory of a system between two points in the space minimizes the action functional. Then by a variational calculus one obtains Euler-Lagrange equations of motion. Reformulation of such an approach to the case of the space of curves in a set-up well suited for our needs was presented in [8].

Here we use Gaussian networks. The term Gaussian network was introduced in [19] and describes an extension of a Bayesian network [12] to continuous variables. Gaussian networks are widely used for decision making and inference. In molecular biology they are used to describe protein dynamics [7]. Gaussian networks better characterize classes of behavior and provide better understanding than the standard representations [19]. To describe time evolution of Gaussian networks we use stochastic diffusion processes whose behavior is effectively characterized by the large deviation theorem due to Freidlin and Wentzell [6]. The theory is based on the property that very unlikely events, when they occur, do so with a high probability by following the pathway that is the most probable. Thus the rare events become in a sense predictable. The crucial role in the theory plays an action functional whose minimization produces an approximation of the probability of rare events and enables the computation of the maximum likelihood trajectory by which such an event occurs [8].

In Section 2, we outline the Principle of Least Action adapted to our needs and present our algorithm to simulate smooth optimal transition paths. Applications, implementation, and a discussion are contained in following sections.

2 Gaussian Network

Gaussian networks [17] are systems which consist of a finite number of nodes whose states are described by continuous variables (these might be positions of certain objects as in our examples). A state of each node is subject to a stochastic dynamics which can be decomposed into a deterministic drift and a stochastic part of the diffusion type. Both the drift and the stochastic part depend on the states of other variables (perhaps just neighboring ones in the spatial networks). We may think about such a dynamics as a continuous limit of a collection of random walks biased by states of other walkers.

In Fig. 1(a), a simple Gaussian network is presented. Arrows describe influence of neighboring nodes on a stochastic dynamics of a given node. More formally, interactions between nodes are contained in the function μ and the matrix Σ in Eq. (1) below. In Fig. 1(b) we can observe transitions between stable space configurations of a Gaussian network.

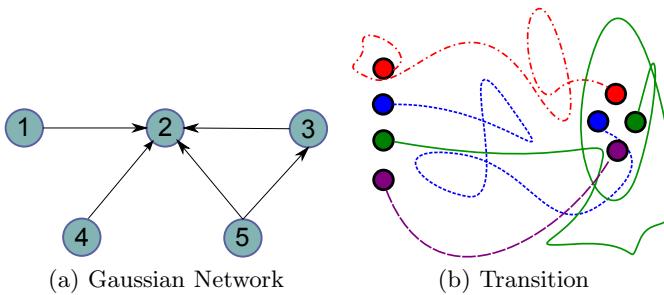


Fig. 1. (a) A schematic representation of nodes in a Gaussian network. (b) Without noise the system would stay in a stable configuration. With noise the system may leave the domain of attraction and can experience rare transitions between stable configurations.

Let $\psi(t)$ is the configuration of the system at time t . For a Gaussian network with n nodes it is a column vector in \mathbb{R}^n which evolves according to

$$\psi(t + \Delta t) = \psi(t) + \mu(\psi(t))\Delta t + \sqrt{\epsilon}\Sigma\Delta W(t), \quad (1)$$

where:

- μ is called an instantaneous drift. We will assume that $\mu(\psi) = B\psi$, for a given $n \times n$ matrix B .
- $\Delta W(t)$ are independent normal random variables with the zero mean and the variance equal to Δt . and Σ is a $n \times n$ matrix which can introduce correlations between stochastic parts of time evolution of different nodes in Eq. (1). We can think about $\Sigma\Delta W(t)$ as the source of a noise.
- ϵ is called the instantaneous variance.

In continuous time, Eq.(1) can be written as the Ito stochastic differential equation

$$d\psi(t) = B\psi(t)dt + \sqrt{\epsilon}\Sigma dW(t) \quad (2)$$

We introduce a local steering contribution \dot{w} as

$$\dot{w} = \dot{\psi} - B\psi$$

where by a dot we denote a derivative with respect to time t .

In the discrete case, $\Delta w = \sqrt{\epsilon}\Sigma\Delta W$. Hence $w(t)$ is called the full "error" or the fluctuation (deviation) of our system. During the minimization process, the value of $\sqrt{\epsilon}$ can be omitted (set to 1).

We propose the following Lagrange function which defines our system:

$$L(\psi, \dot{\psi}) := \frac{1}{2} (\dot{\psi} - B\psi)' A^{-1} (\dot{\psi} - B\psi),$$

where by an apostrophe we denote a transpose of a vector or a matrix and $A := \Sigma\Sigma'$.

The Principle of Least Action – of fundamental use for our applications – indicates that the system moves along the path which minimizes the action functional on the time interval $[0, T]$:

$$S(\psi) := \int_0^T L(\psi(t), \dot{\psi}(t)) dt \quad (3)$$

Our construction is based on the Freidlin-Wentzell theorem [6] on large deviations in stochastic processes which roughly states that the probability of the trajectory $\bar{\psi}$ which deviates from the optimal one is proportional to $\exp(-\frac{S(\bar{\psi})}{\epsilon})$.

To minimize the action functional we use the Euler-Lagrange differential equation (for the Lagrange function of a system of interacting particles we obtain in this way the Newton equations of motion),

$$\frac{\delta L}{\delta \psi} - \frac{d}{dt} \left(\frac{\delta L}{\delta \dot{\psi}} \right) = 0. \quad (4)$$

For a symmetric matrix B one obtains

$$\begin{aligned} \frac{\delta L}{\delta \psi} &= - (BA^{-1}\dot{w}(t)) \\ \frac{d}{dt} \left(\frac{\delta L}{\delta \dot{\psi}} \right) &= (A^{-1}\ddot{w}(t)) \end{aligned}$$

and hence we get

$$\ddot{w}(t) = -ABA^{-1}\dot{w}(t) \quad (5)$$

We solve Eq. (2) as a system of linear ordinary differential equations along a fixed stochastic trajectory and get

$$\psi(T) = \exp(TB)\psi(0) + \left[\int_0^T \exp((T-s)B)\dot{w}(s)ds \right]$$

We know that $\dot{w}(s) = \exp((-ABA^{-1})s)\dot{w}(0)$, so

$$\psi(T) = \exp(TB)\psi(0) + \left[\int_0^T \exp((T-s)B)\exp(-sABA^{-1})ds \right] \dot{w}(0)$$

For the uncorrelated noise ($\Sigma = \mathbf{1}$) we can write:

$$\begin{aligned} \psi(T) &= \exp(TB)\psi(0) + \left[\int_0^T \exp((T-2s)B))ds \right] \dot{w}(0) \\ &= \exp(TB)\psi(0) + \exp(TB) \left[\int_0^T \exp(-2sB)ds \right] \dot{w}(0) \\ &= \exp(TB)\psi(0) + \frac{1}{2}\exp(TB)B^{-1}[\mathbf{1} - \exp(-2TB)]\dot{w}(0) \end{aligned}$$

From the above we get the initial steering configuration

$$\dot{w}(0) = 2B[\mathbf{1} - \exp(-2TB)]^{-1}[\exp(-TB)\psi(T) - \psi(0)] \quad (6)$$

The following procedure lies at the heart of the algorithm. The starting configuration $\psi(0)$ and matrix B must be given.

1. Set the timer $t := 0$.
2. If we do not initialize the 'force' transition to a new configuration, then
 - (a) Use the rules given by Eq. 11 with $\epsilon = 1$ and $\Sigma = \mathbf{1}$
 - (b) Set $t := t + \Delta t$.
3. If we initialize the 'force' transition to a new configuration and provide matrix B_1 , then
 - (a) Compute initial steering configuration for a given transition time T , given by Eq. 6 and denote it as $\dot{w}(t)$.
 - (b) Set local timer $t1 := 0$.
 - (c) Compute the new configuration

$$\psi(t + \Delta t) = \psi(t) + (B\psi(t) + \dot{w}(t))\Delta t$$

- (d) Update the local steering contribution

$$\dot{w}(t + \Delta t) = \dot{w}(t) - (B\dot{w}(t))\Delta t$$

- (e) Set $t1 := t1 + \Delta t$
- (f) Update global timer $t := t + \Delta t$ and, if $t1 < T$, return to 3c else set $B := B_1$ and terminate the transition stage.
4. Return to 2

3 Formation Redeployment

Similarly to [20] we define a *formation* as a set of more than one unit, whose dynamic states are coupled through a common control law. That is, the members of the set must

- Track a desired state relative to a non-empty subset of other members
- The tracking control law must depend at least upon the state of this subset at the minimum.

The second point ensures that the motion of a unit is controlled not only with use of its individual state (position, velocity, etc.), but also is affected by the state of other units. Orbit correction algorithm of the GPS satellites only require position and velocity of an individual satellite thus they do not satisfy the second requirement. Several formations for a set of units are considered:

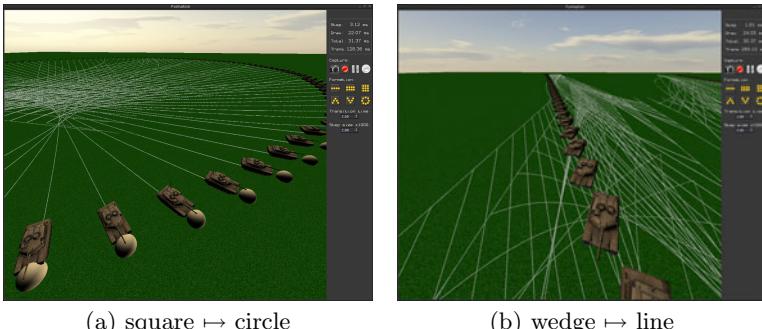


Fig. 2. Screenshots from the sample application. In (a) 225 units transit from the square formation to the circle formation. The current position of an agent is represented by a tank and the distance covered is denoted with a white trace. Similarly in (b) a transition occurs from wedge to line configuration.

- *line* - where the units move in a row
- *double line* - where the units move in a two parallel rows
- *square* - where the units are regularly distributed inside a square
- *circle* - where the units move on the edge of a circle
- *V* - where the units move in a "V" shape
- *wedge* - where the units move in a reverse "V" shape

Fig. 3 shows a schematic description of the formations i.e. every unit is represented by a dot and is connected to its spatial neighbours by edges which illustrate neighbour influence on the node state. The configuration is described as a position of each unit on the plane.

More formally the formation is represented by a Gaussian network with units as nodes and presented connections as arcs. For each pair of connected nodes x and y a pair of vectors is given:

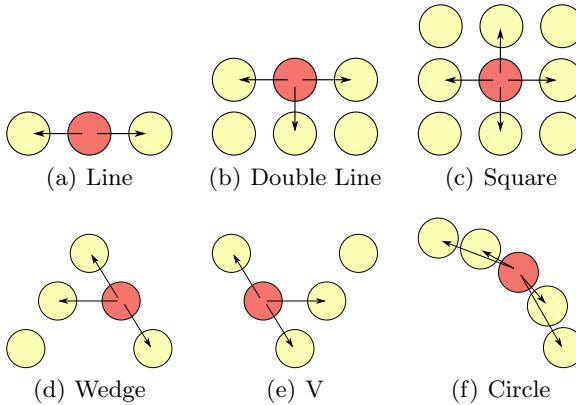


Fig. 3. Relationship between nodes in defined formations. Simple formations, such as *line* or *double line*, require only a connection between the nearest nodes. In more complex structures we have to extend the number of arcs. In *circle* formation each node is connected to the adjacent nodes and to their neighbors. Formations *V* and *wedge* require an additional connection to the nodes on the opposite branch.

- $\mathbf{v}_{x \rightarrow y}$
- $\mathbf{v}_{y \rightarrow x} = -\mathbf{v}_{x \rightarrow y}$

For movement of the group of agents let $N(x)$ denote the number of neighbors of node x , \mathbf{v} the velocity of the entire formation and $\alpha \in [0, 1]$ represents the impact of the neighbors on node position then state of x in time $t + \Delta t$ is defined as follow

$$\psi^{(x)}(t + \Delta t) = \psi^{(x)}(t) (1 - \alpha \Delta t) + \vec{v} \Delta t + \alpha \Delta t \left[\frac{1}{N(x)} \sum_{y \sim x} (\psi^{(y)}(t) + \vec{v}_{y \rightarrow x}) \right] \quad (7)$$

More precisely, the matrix B can be constructed in the following way:

1. Add an artificial node $e \equiv 1$ to the Gaussian network.
2. Set the coefficients as follow

$$\begin{aligned} - B_{xx} &= -\alpha \\ - B_{xy} &= \begin{cases} \frac{\alpha}{N(x)} & \text{if } y \sim x \\ 0 & \text{if } y \not\sim x \end{cases} \\ - B_{xe} &= \mathbf{v} + \frac{\alpha}{N(x)} \sum_{y \sim x} \mathbf{v}_{x \rightarrow y}, \end{aligned}$$

The construction of matrix B was a crucial point in this paragraph and allows for straightforward use of Eq. (1) and Eq. (6) for the simulation. In Fig. 4 the traveled paths are presented by agents during transitions between given formations. As we can see, optimal paths are not the shortest ones. It is expected behavior because the algorithm does not minimize the length of the paths, but rather looks for most probable ones (see Fig. 2). Using shortest paths is highly

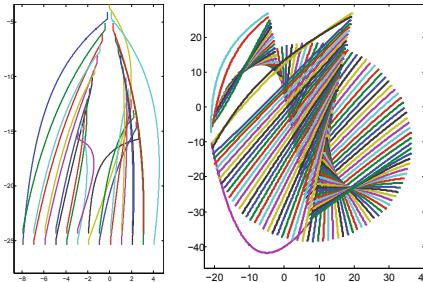


Fig. 4. On the left we see a transition of 25 units from *V* formation to *double line*. On the right 121 tanks redeploy from *circle* to *wedge* configuration.

dissuaded from use due to observed phenomenon in military, where all movements incidents during changes of formation were mainly caused by obtaining the shortest practical route [22].

4 Transitions between Animations

The main purpose of this section is to apply the technique developed in Paragraph 2 to solving the problem of finding optimal transition path between animations. First we have to translate the motion capture data into terms of a Gaussian network. The resulted network should reproduce smoother version of the given motion. Coefficients of the matrix B will be learnt with the use of a regression method. It turns out that the task is non-trivial and first we have to define hierarchical dependencies in the motion data.

The motion capture data¹ grants us with a structure of bones S and set of motions M . S is the skeleton of an animated character that is typically defined as a hierarchy of segments. The skeleton consists of a *root* segment, that is on the top in the hierarchy and does not have any ancestors. Each other segment poses a parent segment and may have one or more children. Segments represent bones and include their properties such as their length, direction, degrees of freedom, local coordinate system, etc. Character motion M_i , with respect to the skeleton S , is defined as a doublet $M_i := (R_i, A_i)$, where R_i is the position of the skeleton's *root* segment for each frame, A_i is a sequence of orientation of each segment that is $A_i = \{a_{i,k}^t; t = 0, \dots, T; k = 0, \dots, \|S\|\}$ so particularly $a_{i,k}^t$ represents an orientation of k th bone during frame t for an i th animation. Current character configuration is defined as the orientation of each bone in the local coordinate system. Using the technique of skeletal animation [4], we can simulate defined motion.

Coefficients of the Gaussian network can be learnt with the use of a linear regression method [5]. For each segment k the regression model can be described

¹ The data used in this project was obtained from <http://www.mocap.cs.cmu.edu>. The database was created with funding from NSF EIA-0196217.

by the dependent vector $a_{i,k}^{t+1}$ and a set of regressors $\{a_{i,1}^t, \dots, a_{i,\|S\|}^t\}$. Results of that straightforward method produced very unstable animations i.e. motion of a character becomes unreliable. Another approach of using *multiple linear regression* [5] with regressors defined by hierarchical structure of the skeleton suffered similar drawbacks. We were also unable to manually define correct relations in given skeletons. It appears that intuitive dependencies between human bones are misleading. For example, the position of *head* was the best described by configuration of: *right femur*, *right tibia*, *right foot* and *left wrist*. Resulting motion did not reproduce correctly given motion and was highly unstable.

To find proper relations in a given motion we decided to use one of the optimization techniques. The number of possible DAGs (Direct Acyclic Graphs) as a function of the number of nodes, $G(n)$, is given by the following recursive function [16]

$$G(1) = 1, G(n) = \sum_{k=1}^n (-1)^{k+1} \binom{n}{k} 2^{k(n-k)} G(n-k) \quad (8)$$

Let us enumerate some values of function G : $G(2) = 3$, $G(4) = 543$, $G(5) = 29281$, $G(10) = 4.2 \times 10^{18}$, $G(100) = 1.1 \times 10^{1631}$. Testing all possible DAG patterns is computationally unfeasible because the number of DAGs grows super-exponentially [12]. In our implementation we have used simulated annealing method [11] to search in possible DAG space. For a given skeleton S and animation that consists of F frames the energy function has the following form:

$$\mathcal{E}(M_i) = \frac{1}{F\|S\|} \sum_{f=1}^F \sum_{b=1}^{\|S\|} \|c_{i,b}^f - s_{i,b}^f\|^2, \quad (9)$$

where $c_{i,b}^f$ is a configuration of bone b on the i th animation at frame t obtained from motion capture data, $s_{i,b}^f$ has similar meaning, but is computed by simulation of the Gaussian network. In other words we compute *mean square error* between positions of bones given by motion capture data in each frame and their configuration computed with presented method.

In addition, we impose the following requirements:

- Coefficients in matrix B are bound by constant b_r .
- The mean square error computed for the entire skeleton should not exceed some (large) constant s_b .

The purpose of these conditions is to ensure the numerical stability of the algorithm. During simulations, we set $b_r = 10^3$ and $s_b = 10^{12}$. The applied simulated annealing algorithm would eventually converge to the target bones hierarchy (see Image 5(a)).

After applying described scheme to the motion we obtain matrix B . The animation obtained from the simulation of the Gaussian network looks very similar to the original motion. The main difference is that the new animation is more smooth. To maintain the motion in large time horizon we add small noise to

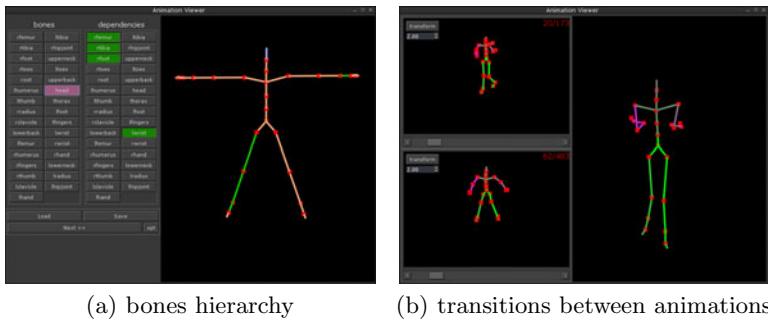


Fig. 5. Screenshots from the sandbox. In (a) we can see that orientation of the head is best described by orientation of: right femur, right tibia, right foot and left wrist, (b) presents transition from walking to jumping animation.

the simulation [2], without that, the motion gradually fades away (it converges to the steady state). At any moment, the animation of a motion can be interrupted and by use of the method from Par. [2], even for a nonsymmetric matrix B , smoothly transformed to a selected frame of another motion (Fig. [5(b)]).

5 Implementation

The programme has been implemented in language D [3]. Some matrix operations incorporate LAPACK, BLAS and CUBLAS [4] subroutines. All test runs were executed on a machine with Intel Core 2 Q9300 2.50 GHz CPU, 4GB RAM and NVIDIA GTX 480. The application is single threaded, so it is applicable to the core of only one processor. All computations were performed with double precision arithmetic.

The mean square error (MSE) between the target configuration and a simulated one can be controlled by adjusting the step size. For the *transition time* set on 2.0 and *step size* on 0.002 the observed value of MSE was lower than 0.0001.

An efficient computation of a matrix exponential plays a crucial role in the presented method. Numerous methods for computing e^A were developed. The straightforward one, which uses Euler series, is inefficient even in the scalar case. The survey [13] presented a wide variety of methods and pointed the most powerful ones. Today, due to evolution in computer hardware and highly optimized BLAS subroutines the most cogent technique is the scaling and squaring method combined with Padé approximants [9]. The sequential version of that algorithm has been implemented. To parallelize the computation of the matrix exponential we use NVIDIA CUBLAS library [10,14] and substitute serial matrix operations with parallel ones.

In Table II we can see dependencies between the number of simulated objects (tanks) and the time required for a single step and a transition. As we can see,

Table 1. Step and transition times

| quantity | step (ms) | CPU (ms) | GPU (ms) | speedup |
|----------|-----------|----------|----------|--------------|
| 25 | 0.04 | 1.7 | 40 | 0.07 |
| 100 | 0.6 | 24 | 109 | 0.22 |
| 169 | 1.6 | 75 | 193 | 0.38 |
| 225 | 3.1 | 172 | 275 | 0.63 |
| 400 | 9 | 680 | 637 | 1.07 |
| 625 | 22 | 2200 | 1470 | 1.50 |
| 900 | 49 | 12000 | 3480 | 3.45 |
| 1225 | 89 | 54000 | 6890 | 7.84 |
| 1600 | 155 | 253000 | 12400 | 17.57 |
| 2025 | 240 | 785000 | 26200 | 29.96 |

a single step can be computed very fast. The most consuming part during w_0 calculation is matrix exponential. GPU accelerated version is up to 30 times faster than the sequential one. An analysis of parallel profiler output shows that time is mainly (up to 70%) spent on matrix multiplications (*dgemm*). Transfers of the data from host to device (CPU → GPU) and vice versa take up to 30% of the time. The rest of the time is consumed on memory allocations and other matrix operations. We should emphasize that computations are performed for x, y and z coordinates independently.

6 Conclusions

The algorithm for determining optimal transition paths was presented. For the evaluation purpose two applications have been successfully implemented. The problem of the formation redeployment was solved by our algorithm and produced enjoyable results. The resulting group dynamics is highly complex and a great care must be taken when such an environment is simulated. The other tested application was devoted to transitions between animations. The problem was more complicated due to the need of representing the motion in terms of Gaussian networks. The obtained animations were satisfying for the viewer. Simulations on the GPU were also performed and provided a significant gain in the runtime, but only in the large enough networks. Future enhancements may include hierarchical grouping of Gaussian nodes before transitions which would significantly improve a computation time. Another option is to optimize the transition time rather than to use the given one.

Acknowledgements. M. Matuszak and T. Schreiber gratefully acknowledge the support from the Polish Ministry of Scientific Research and Higher Education grant N N201 385234 (2008–2010). The work of M. Matuszak has also been supported by the European Social Fund, Government of Poland and Kuyavian-Pomeranian Voivodeship as a part of Integrated Operational Program for Regional Development, Activity 8.2.2.

References

1. Balch, T., Arkin, R.C.: Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation* 14(6), 926–939 (1998)
2. Balch, T., Hybinette, M.: Behavior-Based Coordination of Large-Scale Robot Formations. In: International Conference on Multiagent Systems – ICMAS, pp. 363–364 (2000)
3. Bell, K., Iglesias, L.I., Kelly, S., Parker, M.: Learn to Tango with D. Apress (2008)
4. Burtnyk, N., Wein, M.: Interactive skeleton techniques for enhancing motion dynamics in key frame animation. *Commun. ACM* 19, 564–569 (1976)
5. Draper, N.R., Smith, H.: Applied regression analysis, 3rd edn. Wiley, New York (1998)
6. Freidlin, M.I., Wentzell, A.D.: Random perturbations of dynamical systems, 2nd edn. Grundlehren der Mathematischen Wissenschaften, vol. 260. Springer, New York (1998)
7. Haliloglu, T., Bahar, I., Erman, B.: Gaussian dynamics of folded proteins. *Physical Review Letters* 79, 3090–3093 (1997)
8. Heymann, M., Vanden-Eijnden, E.: The Geometric Minimum Action Method: A Least Action Principle on the Space of Curves. *Comm. Pure Appl. Math.* 61(8), 1052–1117 (2008)
9. Higham, N.J.: The Scaling and Squaring Method for the Matrix Exponential Revisited. *SIAM J. Matrix Anal. Appl.* 26(4), 1179–1193 (2005)
10. Kirk, D.B., Hwu, W.-M.W.: Programming Massively Parallel Processors: A Hands-on Approach, 1st edn. Morgan Kaufmann, San Francisco (2010)
11. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by Simulated Annealing. *Science* 220, 671–680 (1983)
12. Koski, T., Noble, J.: Bayesian Networks: An Introduction. John Wiley & Sons, Ltd., Chichester (2009)
13. Moler, C., Van Loan, C.: Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later. *SIAM Rev.* 45(3) (2003)
14. NVIDIA CUBLAS Library, NVIDIA Corporation (2009)
15. Reynolds, C.W.: Flocks, Herds, and Schools: A Distributed Behavioral Model. *Computer Graphics*. In: SIGGRAPH 1987 vol. 21(4), pp. 25–34 (1987)
16. Robinson, R.W.: Counting Unlabelled Acyclic Digraphs. In: Combinatorial Mathematics V. Springer Lecture Notes in Mathematics, pp. 28–43 (1977)
17. Rue, H., Held, L.: Gaussian Markov Random Fields: Theory and Applications. Monographs on Statistics & Applied Probability 104 (2005)
18. Safanova, A., Hodgins, J.K.: Analyzing the physical correctness of interpolated human motion. In: ACM Siggraph/Eurographics Symposium on Computer Animation (SCA 2005), pp. 171–180 (2005)
19. Shachter, R.D., Kenley, C.R.: Gaussian influence diagrams. *Management Science* 35(5), 527–550 (1989)
20. Scharf, D.P., Hadaegh, F.Y., Ploen, S.R.: A Survey of Spacecraft Formation Flying Guidance and Control (Part I): Guidance. In: American Control Conference, vol. 2, pp. 1733–1739 (June 2003)
21. Scharf, D.P., Hadaegh, F.Y., Ploen, S.R.: A Survey of Spacecraft Formation Flying Guidance and Control (Part II): Control. In: American Control Conference, vol. 4, pp. 2976–2985 (June 30 - July 2, 2004)
22. U.S. Marine Corps Marine Rifle Squad. Marine Corps Warfighting Publication (MCWP) 3–11.2 (1997)

HTN-Style Planning in Relational POMDPs Using First-Order FSCs

Felix Müller and Susanne Biundo

Institute of Artificial Intelligence, Ulm University, D-89069 Ulm, Germany

Abstract. In this paper, a novel approach to hierarchical planning under partial observability in relational domains is presented. It combines hierarchical task network planning with the finite state controller (FSC) policy representation for partially observable Markov decision processes. Based on a new first-order generalization of FSCs, action hierarchies are defined as in traditional hierarchical planning, so that planning corresponds to finding the best plan in a given decomposition hierarchy of predefined, partially abstract FSCs. Finally, we propose an algorithm for solving planning problems in this setting. Our approach offers a way of practically dealing with real-world partial observability planning problems: it avoids the complexity originating from the dynamic programming backup operation required in many present-day policy generation algorithms.

1 Introduction

Assisting human users in performing exceptional, complex tasks or even in managing their day-to-day life requires advanced planning technology. The reason is that user-centered planning is challenging in two respects. First, the real-world planning domains underlying these applications are large; they show hierarchical structures and typically there are numerous options to solve a given problem in a user-conform manner. Second, information about the user state is essential as the automatically generated plans should not just solve the problem, but should do so by taking also the user's emotional state into account. Information about the user state, just like information about the world state, depends on sensory input and is thus only partially observable and inherently uncertain.

In order to address partial observability, relational partially observable Markov decision processes (POMDPs) can be employed. They allow to handle complex domain dynamics and are thus appropriate to represent many real-world decision problems that arise in the context of user-centered planning. A POMDP-based system to provide assistance for persons with dementia is described by Boger *et al.* [2], for example. Solving POMDPs has been shown to be expensive, however. It ranges from PSPACE-complete for finite horizons [11] to undecidable for infinite horizons [8] even without factored representations. Therefore, applying POMDPs works well for small to medium sized domains, while scaling beyond this size is problematic.

The challenge to get along with complex domains is addressed by approaches that use (variants of) hierarchical task network (HTN) planning [4]. Among others, HTNs provide the means to encode standard solutions to planning problems and with that enable the exploitation of expert knowledge in solution discovery. Applications in many

real-world domains rely on HTN-based systems [9][10]. While making use of sophisticated representation schemes that allow to compactly describe large domains, HTN-based systems are aimed at planning for fully observable deterministic domains, however.

Planning in POMDPs with first-order abstraction has only recently started to be investigated by Sanner and Kersting [16] as well as Wang and Khordon [17]. Here, state, action, and observation spaces are factored into a first-order representation, as in traditional AI planning. There are approaches to (non-first-order) hierarchical planning in POMDPs, such as the one introduced by Hansen and Zhou [7]. Their approach uses POMDPs with abstract actions, but instead of a set of predefined finite state controllers per abstract action, a sub-POMDP is solved to compute a policy for each abstract action.

HTN planning in partially observable domains is introduced by Bouguerra and Karlsson [3], where HTN planning is extended by probabilistic action effects and observations, and uncertainty about the environment is represented using belief states. The approach does not allow for the specification of cyclic methods and does not cover full POMDP semantics with rewards, however.

In this paper, we present an approach to transfer the benefits of HTN planning to the POMDP setting, thereby allowing for the exploitation of domain expert knowledge to generate good policies in large, compactly represented POMDP domains. We begin by briefly reviewing (PO)MDPs and HTN planning in Section 2. We then enhance the FSC policy representation such that controllers for relational domains can be compactly specified (Section 3): transitions between controller nodes are governed by formulas, which represent conditions that need to be fulfilled by observations instead of annotating transitions with all possible observations directly. After that, Section 4 defines action hierarchies by introducing abstract actions, each of which is associated with one or more implementations that are given as predefined enhanced controllers, analogously to traditional HTN planning. In Section 5 we describe how solutions can be generated in our formalism before we conclude in Section 6.

2 Background

Our work builds on the POMDP model and HTN planning. We will give short introductions to both in the following.

2.1 MDPs and POMDPs

MDPs are a framework for planning in fully observable domains with probabilistic actions [14]. An MDP is a tuple (S, A, T, R) , where S is the set of states, A is the set of actions, $T(s, a, s') = P(s'|s, a)$ is the transition function, giving the probability for ending up in state s' upon executing action a in state s , and $R(s, a)$ determines the immediate rewards incurred by executing a in s . Additionally, one often specifies a horizon h for the number of time steps after which the process terminates, or a discount factor, $0 \leq \gamma \leq 1$, for discounting rewards earned at later decision stages, or both. POMDPs generalize MDPs by making the world state only indirectly observable. A POMDP introduces a set of observations O and an observation function $Z(s, a, o) = P(o|a, s)$,

denoting the probability of receiving observation o when a is executed and the *resulting* state is s . Since the state cannot be directly observed, a belief state, i. e., a probability distribution over states, is usually maintained by the planning agent.

A solution for a POMDP is a policy π that maximizes the accumulated reward, or equivalently minimizes accumulated cost, for an initial belief state b_0 . One possibility to represent a policy is a finite state controller, i. e., a directed graph where nodes are labeled with actions and edges are labeled with observations [5]. Policies can be generated using dynamic programming approaches such as policy iteration, where a dynamic programming backup operation is used to iteratively improve an explicitly or implicitly represented policy. Other algorithms include approximative approaches, e. g., point-based value iteration as described by Pineau *et al.* [13].

In this paper, we will use a POMDP variant introduced by Hansen, called indefinite-horizon POMDPs with action-based termination [6]. Such POMDPs include one or more terminal actions, which cause immediate transition to a terminal state from any state. Also, no discounting is used (so that $\gamma = 1$) and action rewards are restricted to be negative. As a result, there is no bound on the number of steps before termination, but the optimal policy (and all policies with finite value) eventually terminate with probability 1. For the sake of readability, we will sometimes speak in terms of positive costs. We will restrict the description of our approach below to a single terminal action, though it can be generalized to more than one terminal action.

To overcome the explicit state enumeration used in the definition above, Sanner and Kersting [16] use a first-order representation based on the situation calculus. We will employ a similarly powerful representation, namely the probabilistic planning domain definition language (PPDDL) [18], which is based on a probabilistic extension of the action description language ADL [12] with MDP semantics. In PPDDL, a domain is given in terms of predicates and typed objects, and states are sets of ground predicates. A ground predicate is true in a state if it is contained in the state and false otherwise (closed world assumption).

A PPDDL action as shown in Figure I transforms a state into another state by adding or removing predicates when it makes them true or false, respectively. It also defines the conditions and probabilities that govern their addition and removal, alongside with the condition for receiving rewards. Actions are parametrized, which is why we distinguish between *action schemas*, which are the members of A , and *action instances* $I(A)$, where (some) action parameters are bound to domain objects, i. e., ground. An action schema is instantiated into an action instance by substituting parameters by objects of appropriate type. To allow for partial observability, we augment PPDDL actions with the possibility to include observations by introducing the field :observation. Observations are also sets of ground predicates, similar to states. The conditions and probabilities of receiving an observation are specified in the same manner as for an action's effects. As usual for POMDP observations, the conditions for receiving observations are evaluated after the action effects are applied.

As an example from the context of companion systems, we will use the task of renovating an apartment as part of the larger domain of moving: the task is to generate a plan that guides a user in painting each room of an apartment in their preferred color. At the same time, the plan should account for the user's preferences and emotional

```
(:action paint_room
:parameters (?r - Room ?c - Color)
:effect
(and
(has_color ?r ?c)
(forall (?c_old - Color) (when (not (?c_old = ?c)) (not (has_color ?r ?c_old))))
(when (happy) (decrease (reward) 1))
(when (not (happy)) (decrease (reward) 10))
(when (happy) (probabilistic 0.3 (not (happy)))))
:observation
(and
(when (happy) (probabilistic 0.9 (happy_O) 0.1 (not (happy_O))))
(when (not (happy)) (probabilistic 0.8 (not (happy_O)) 0.2 (happy_O))))
```

Fig. 1. The `paint_room` action of our example domain, formulated in augmented PPDDL, which changes the color of a room and has low cost if the user is happy and high cost if they are not. Painting a room also has a chance to make the user unhappy and gives a noisy observation of the user’s mood.

state and choose actions accordingly. More precisely, states are defined by the predicates (`user_wants_color ?r - Room ?c - Color`), (`has_color ?r - Room ?c - Color`), and (`happy`), which represent the desired and current colors of each room and the user’s emotional state, respectively. Note that for the sake of presentational simplicity, the user’s emotional state can only be happy or unhappy. A more complex emotional model, such as the three-dimensional pleasure arousal dominance (PAD) model [15], could be modeled easily by introducing predicates (`pleasure ?l - Level`), (`arousal ?l - Level`), and (`dominance ?l - Level`). Both the user’s emotional state and the color the user wants to paint a room can only be observed indirectly via the separate observation predicates (`user_wants_color_O ?r - Room ?c - Color`) and (`happy_O`).

There are five available actions. The first is (`ask_user ?r - Room`) which asks the user to tell the system the preferred color for a room. It generates a perfect observation of the user’s desired color for a room at a low cost. The second action is the “easy” way to paint a room: call a painter, i.e., (`call_painter ?r - Room ?c - Color`). This action changes the color of the room accordingly at a medium cost. The other possibility to paint a room is to let the user paint the room themselves using (`paint_room ?r - Room ?c - Color`) (the action represented by Figure 1), which has a low cost if the user is happy and high cost if they are not. Painting a room by themselves also has a chance to make the user unhappy. To improve the user’s mood, it is possible to cheer up the user with the action (`cheer_up`), which also has a low cost and a good chance to make them happy. Both the (`paint_room`) and (`cheer_up`) actions generate a noisy observation of the user’s mood. Finally, there is the terminal action (`finish`), which ends the process and has a high cost if there exists a room for which the actual color differs from the color the user wants. As a result, there is some choice as to how the goal of having the rooms of the apartment painted according to the user’s preferences can be fulfilled using the actions in the domain: when the user is unhappy, there is a trade-off between cheering them up and letting them paint by themselves versus calling a painter.

2.2 HTN Planning

HTN planning [4] is an approach to planning in fully observable deterministic domains. World states are described in a similar manner as shown above, yet actions are deterministic and the (single) initial world state is completely known. Thus, no observations are required, because the evolution of the world can be predicted with certainty. Apart from the domain dynamics, the HTN planning problem definition includes a hierarchy of actions: in addition to normal, directly executable actions, there are also abstract actions. Abstract actions cannot be executed directly. Instead, one or more implementations, called decomposition methods, are provided for each abstract action in form of partial plans: a method is a partially ordered set of partially instantiated primitive or abstract actions, called a task network. Methods are specified by domain experts and represent known possible solutions to subproblems in the form of abstract actions, therefore encoding a domain expert’s knowledge about typical problem-solving “recipes”. Some HTN planners such as the original SHOP [10] use a more simple kind of method, where the tasks are totally ordered into a sequence of actions.

In contrast to reward-based goal models or goal state specifications, the goal in HTN planning is to *perform* one or more abstract actions specified in the initial task network. At planning time, abstract actions are iteratively eliminated by replacing abstract actions with implementing methods until a plan is found that contains only primitive actions. Because the goal is to perform actions instead of generating a particular state, actions are also often called (primitive or abstract, respectively) tasks in HTN planning. In the course of our paper, we will use both interchangeably.

3 First-Order FSCs

For relational POMDPs such as our example from the previous section, policies can be represented using a first-order version of finite state controllers. The definition of first-order FSCs (FOFSCs) is the first contribution of this paper and is required for the introduction of action hierarchies in Section 4. The rationale behind introducing FOFSCs is that because the number of possible observations is very large in relational POMDPs, including an edge label for every observation as in ordinary FSCs is inconvenient. FOFSCs allow for a more concise representation by using formulas to govern transitions. Note that our definition already includes terminal controller states and nodes associated with partially instantiated actions to allow for the addition of hierarchical elements described below.

Definition 1 (First-Order Finite State Controller). A first-order finite state controller (FOFSC) is a tuple $(Q, q_0, q_t, A, \alpha, O, \delta)$. A and O are sets of actions and observations, respectively. Q is the set of controller nodes. The node $q_0 \in Q$ is the start node, and the node $q_t \in Q$, $q_t \neq q_0$, is the terminal node. Each node $q \in Q \setminus \{q_t\}$ is associated with a (partially) instantiated non-terminal action $a_i \in I(A)$ via the action association function $\alpha : Q \setminus \{q_t\} \rightarrow I(A)$. Transitions are defined via the transition function $\delta : Q \setminus \{q_t\} \times Q \rightarrow \mathcal{F}(O)$, where $\mathcal{F}(O)$ is the set of well-formed formulas over the observation space. Transitions need to be mutually exclusive and exhaustive, i. e., for all obtainable observations o of $\alpha(p)$, (1) whenever $o \models \delta(p, q_i)$ it holds that $o \not\models \delta(p, q_j)$ for all $q_j \neq q_i$ and (2) $o \models \bigvee_{q_i \in Q} \delta(p, q_i)$.

For nodes p and q and observation o received from executing $\alpha(p)$, $\delta(p, q) = \varphi$ means that the controller state changes from p to q if $o \models \varphi$. Making the formulas mutually exclusive guarantees that there is no ambiguity when choosing the successor state. Exhaustiveness guarantees that every possible observation of an action is covered. However, this requirement can be dropped by assuming transition to an “error node” (associated with an action that incurs strongly negative reward) when no condition is fulfilled.

Executing an FOFSC works by iteratively executing the action associated with a node (starting with q_0), receiving an observation, and following the link whose formula is fulfilled by the observation. This is done until the terminal node is reached, at which point the terminal action is executed. Figure 2 shows an example FOFSC. For the sake of clarity, we omit edges between two nodes p and q when $\delta(p, q) \equiv \perp$. Action parameters do not need to be ground according to the FOFSC definition, allowing for multiple usage of the same variable in several nodes of the FOFSC. However, non-ground parameters introduce a source of nondeterminism into the FOFSC that might prohibit the unique assignment of an accumulated reward value to the policy the FOFSC represents. Therefore, we define that an FOFSC is *executable* if and only if it is fully ground. For an executable FOFSC, the accumulated reward earned by executing it in a given initial belief state can for example be calculated by solving the induced system of linear equations as described by Hansen [5], where the accumulated reward of the terminal state q_t is defined by the cost of the terminal action. When the planning horizon is finite, cycles in the FOFSC can be eliminated for evaluation purposes: since execution is assumed to stop when either the terminal node or the horizon is reached, at the latest, they can be unrolled. This allows for a fast forward-evaluation of FOFSCs.

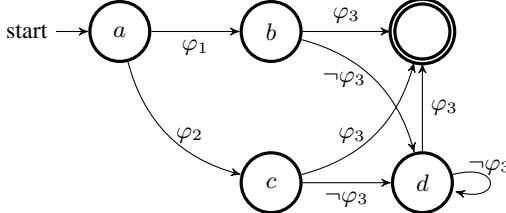


Fig. 2. An example FOFSC, where $a = (\text{ask_user } ?r)$, $b = (\text{paint_room } ?r \text{ WHITE})$, $c = (\text{paint_room } ?r \text{ LIGHTBLUE})$, $d = (\text{cheer_up})$, $\varphi_1 = (\text{user_wants_color_O } ?r \text{ WHITE})$, $\varphi_2 = (\text{user_wants_color_O } ?r \text{ LIGHTBLUE})$, and $\varphi_3 = (\text{happy_O})$. In words, the described course of action proposes to ask the user for the desired room color, paint the room accordingly, and cheer up the user until their mood seems better.

Note that FOFSCs are a natural generalization of the sequences of ground actions generated by many planners for fully observable and deterministic planning domains, such as SHOP.

4 Hierarchical Relational POMDPs

Now, we can begin adding hierarchical information to create action hierarchies by introducing abstract actions and the means to carry them out. Abstract actions are specified

in terms of a name and a parameter list just like ordinary actions of the underlying POMDP, but as they are not part of the underlying POMDP, no transition probabilities, observation probabilities or rewards are associated with them. However, as only name and parameter list of an action are needed for the construction of FOFSCs, it is possible to construct FOFSCs containing abstract actions. Obviously, such abstract actions cannot be executed directly and neither can FOFSCs that contain abstract actions.

Intuitively, hierarchical planning as we will define it in this section starts with an initial FOFSC containing abstract actions that have to be iteratively refined into more concrete implementations. This is done by applying *decomposition methods*, each mapping an abstract action to a possible implementation, until all actions are primitive as is done in HTN planning. However, in contrast to normal HTN planning, where the goal is to find an executable primitive plan, the goal here is to find the optimal-reward primitive FOFSC in the induced hierarchy according to evaluation with an initial belief state and a planning horizon.

For a formal definition of hierarchical relational POMDPs using FOFSCs with abstract actions, let $\text{FSCs}(X)$ denote the set of FSCs over a set of actions X .

Definition 2 (Hierarchical Relational POMDP). A hierarchical relational POMDP (*HPOMDP*) is a tuple $(P, A_a, M, C_{\text{init}}, b_0, h)$, where

- $P = (S, A, T, O, Z)$ is the underlying relational indefinite-horizon POMDP with action-based termination,
- A_a is a finite set of abstract actions with $A_a \cap A = \emptyset$,
- $M \subseteq A_a \times \text{FSCs}(A \cup A_a)$ is a finite set of decomposition methods mapping abstract actions to their possible implementations,
- $C_{\text{init}} \in \text{FSCs}(A \cup A_a)$ is the initial FOFSC containing abstract actions,
- b_0 is the initial belief state, and
- $h \in \mathbb{N} \cup \{\infty\}$ is the planning horizon.

A possible abstract action for the example POMDP introduced in the previous section is `(handle_room ?r)`, which completely handles one room. Expert knowledge is then used to define useful courses of action as implementation for abstract actions. Consequently, one possible method is to ask the user for the desired room color, painting the room accordingly and cheer them up afterwards, if needed. Hence, the FOFSC given in Figure 2 is a method for `(handle_room ?r)`. Other implementations include asking for the desired color and calling a painter, and so on.

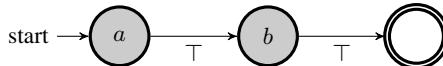
Applying a method transforms a partially abstract FOFSC into a more concrete version by replacing the abstract action with the implementation specified by the method. For a formal definition of method application, let $\text{pd}_C(q)$ denote the set of predecessors of a node q in an FSC C with node set Q and transition function δ , i.e., $\text{pd}_C(q) = \{p \in Q | \delta(p, q) \neq \perp\}$ and $\text{sc}_C(q)$ the set of successors of q , i.e., $\text{sc}_C(q) = \{p \in Q | \delta(q, p) \neq \perp\}$. We use superscripts to disambiguate the members of the different FSCs involved, e.g., we write Q^1 for the node set of the FSC C^1 .

Definition 3 (Method Application). Let C^1 be an FOFSC containing a node q_a^1 with $a_i = \alpha^1(q_a^1)$ being an abstract action instance, $m = (a, C)$ a method for the corresponding action schema a of a_i , and C^2 an isomorphic copy of C with $Q^1 \cap Q^2 = \emptyset$.

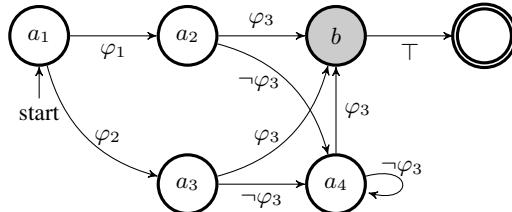
Let σ be a parameter substitution such that $\sigma(a) = a_i$. Applying m to q_a^1 results in a new FOFSC $C^3 = (Q^3, q_0^1, q_t^1, A^1, \alpha^3, O^1, \delta^3)$ with:

$$\begin{aligned}
 - Q^3 &:= (Q^1 \cup Q^2) \setminus \{q_a^1, q_t^2\} \\
 - \alpha^3(q) &:= \begin{cases} \alpha^1(q), & q \in Q^1 \\ \sigma(\alpha^2(q)), & q \in Q^2 \end{cases} \\
 - \delta^3(p, q) &:= \begin{cases} \delta^1(p, q), & p, q \in Q^1 \\ \sigma(\delta^2(p, q)), & p, q \in Q^2 \wedge \neg(p \in \text{pd}_{C^2}(q_t^2) \wedge q = q_0^2) \\ \sigma(\delta^2(p, q)) \vee \delta^1(q_a^1, q_a^1), & p \in \text{pd}_{C^2}(q_t^2) \wedge q = q_0^2 \\ \delta^1(p, q_a^1), & p \in \text{pd}_{C^1}(q_a^1) \wedge q = q_0^2 \\ \sigma(\delta^2(p, q_t^2)) \wedge \delta^1(q_a^1, q), & p \in \text{pd}_{C^2}(q_t^2) \wedge q \in \text{sc}_{C^1}(q_a^1) \\ \perp, & \text{else} \end{cases}
 \end{aligned}$$

Applying a method m to a node q in an FOFSC C is denoted by $C[q \leftarrow m]$. Intuitively, when an abstract action is replaced by an implementation, all incoming edges of the abstract action are “redrawn” to point to the initial node of the implementing FOFSC, and all outgoing edges of the abstract action are combined with the edges pointing to the terminal node of the implementing FOFSC. Additionally, the substitution is required to pass the already bound action parameters of the replaced abstract action to its implementation. Figure 3 shows how the abstract action (handle_room KITCHEN) is replaced by one of the introduced methods in our example.



(a) A partially abstract FOFSC in our example, where the kitchen and the living room are to be painted, so $a = (\text{handle_room KITCHEN})$ and $b = (\text{handle_room LIVING_ROOM})$.



(b) The resulting FOFSC after applying a method ($a_1 = (\text{ask_user KITCHEN})$, $a_2 = (\text{paint_room KITCHEN WHITE})$, $a_3 = (\text{paint_room KITCHEN LIGHTBLUE})$, $a_4 = (\text{cheer_up})$, $\varphi_1 = (\text{user_wants_color_O KITCHEN WHITE})$, $\varphi_2 = (\text{user_wants_color_O KITCHEN LIGHTBLUE})$, and $\varphi_3 = (\text{happy_O})$).

Fig. 3. The initial FOFSC C_{init} for our example domain (Figure (a)) and the resulting FOFSC after the method shown in Figure 2 has been applied to the node associated with action a (shown in Figure (b)). Note how the parameter $?x$ is substituted by KITCHEN.

Precisely, the new transition function δ^3 is constructed to capture the intended meaning of method application while fulfilling the “mutually exclusive and exhaustive”

requirement of Definition 1: the first two cases preserve the internal transitions of C^1 and C^2 . The third case accounts for the special case when there is a sling at q_a^1 : a sling means the abstract action can be repeated immediately after its completion. This translates to a transition from a last action to the first action in its implementation. As the implementation may already contain such a transition, a disjunction of both transition conditions is introduced. The fourth case handles transitions to q_a^1 , which result in transitions to the initial node of C^2 . The fifth case connects the predecessor nodes of q_t^2 to the successor nodes of q_a^1 . As both the transitions to q_t^2 and from q_a^1 are labeled with transition conditions in C^1 and C^2 , respectively, each possible combination in $\text{pd}(q_t^2) \times \text{sc}(q_a^1)$ is labeled with the conjunction of the respective transition conditions. Finally, the last case states that no other transitions are introduced.

By iteratively replacing all abstract actions and binding action parameters to constants until all actions are primitive and ground, we can generate executable FSCs. For a primitive FSC C , we can evaluate its accumulated cost $V(C, b_0, h)$ with respect to the initial belief state b_0 and horizon h . This leads us to the definition of a solution for a hierarchical relational POMDP:

Definition 4 (Solution). A solution of a hierarchical relational POMDP is a fully primitive and executable decomposition C^* (i. e., fully ground and without abstract actions) of C_{init} , such that there exists no other fully primitive and executable decomposition C with smaller accumulated cost, i. e., $\forall C, V(C, b_0, h) \geq V(C^*, b_0, h)$, with respect to evaluation using b_0 and h .

Note that a solution is required to be a decomposition of C_{init} . This means that it is in general not an optimal policy for the underlying POMDP. The relative solution quality of the HPOMDP solution compared to the optimal solution of the underlying POMDP directly depends on the modeled methods: methods can both be chosen so that the HPOMDP solution approximates the optimal solution arbitrarily closely and so that the HPOMDP solution is arbitrarily bad. Since the methods are modeled by a domain expert, relative HPOMDP solution quality is determined by the quality of the expert knowledge.

5 Algorithm

The algorithm we propose for hierarchical relational POMDP planning is outlined in Algorithm 1 and employs an A* search in the space of FOFSCs induced by the action hierarchy: it iteratively decides whether an FOFSC is a solution (line 4), generates successor FOFSCs if it is no solution (line 6–9), ranks them according to their cost-plus-heuristic value (line 10), and proceeds by considering the lowest-cost candidate next. A closed-set D that contains already visited FOFSCs is used to prevent the algorithm from considering the same FOFSC twice (lines 5 and 10). Note that the method application in line 8 generates only ground instances, so that the cost and heuristic functions defined below can be evaluated unambiguously.

5.1 Costs and Heuristics

For the cost and heuristic definitions, we distinguish between the part of an FSC that can be assessed exactly cost-wise, and the part for which cost must be estimated. Cost can be calculated exactly when there is no abstract action involved, i. e., for the part

between q_0 and the first nodes associated with an abstract action. For an abstract action, neither immediate cost nor transition probabilities are known, therefore it is impossible to calculate costs correctly beyond the first abstract action encountered. Let $Q_f(C)$ denote the set of those “first abstract nodes”, i.e., nodes q for which $\alpha(q)$ is abstract and there is a fully primitive path with probability greater than zero leading from q_0 to q . The cost function $g(C)$ of an FSC C is then defined as the accumulated cost V of an FSC C' where every edge pointing to a node in $Q_f(C)$ is redrawn to the terminal node. Note that for a fully primitive FSC C , $C = C'$ and hence, $g(C) = V(C)$, i.e., $g(C)$ is the true cost of C .

```

Input : A HPOMDP  $(P, A_a, M, b_0, C_{\text{init}}, h)$ .
Output : A solution or fail.
1  $F := \langle C_{\text{init}} \rangle, D := \emptyset$ 
2 while  $F \neq \varepsilon$  do
3    $C_{\text{cur}} := \text{head}(F), F := \text{tail}(F)$ 
4   if  $C_{\text{cur}}$  is executable then return  $C_{\text{cur}}$ 
5    $D := D \cup C_{\text{cur}}$ 
6    $Q :=$  the nodes  $q$  in  $C_{\text{cur}}$  with  $\alpha(q)$  abstract
7   foreach  $q \in Q$  do
8      $\langle m_1, \dots, m_n \rangle :=$  the methods for  $\alpha(q)$ 
9      $F_q := \langle C_{\text{cur}}[q \leftarrow m_1], \dots, C_{\text{cur}}[q \leftarrow m_n] \rangle$ 
10     $F := \text{merge}(F, F_q \setminus D)$ 
11 return fail
```

Algorithm 1. The HPOMDP Planning Algorithm, which employs A* search in the space of FOFSCs. The call to merge() sorts all plans C in F and F_q in ascending order according to their cost-plus-heuristic value $f(C) = g(C) + h(C)$.

The heuristic estimate $h(C)$ tries to assess the cost induced by the remaining part of C . We will present admissible estimates h for both the finite and the indefinite horizon cases. For the indefinite horizon case, we will calculate a lower bound for the cost incurred by reaching q_t from each node $q \in Q_f(C)$. The value of $h(C)$ is then calculated as the weighted sum of the cost bounds for each q according to the probabilities of reaching q through a fully primitive path, which are in turn lower bounds for the true probabilities of reaching each q . To calculate the estimate for each $q \in Q_f(C)$, we assume that we can choose transitions at each node in C instead of determining the successor node according to received observations. Additionally, we assign to each edge the minimal cost of the action of its predecessor node if it is primitive and 0 if it is abstract. The cost bound for a node $q \in Q_f(C)$ is then given by the cost of the shortest path from q to q_t in the resulting graph.

In the finite horizon case, it is possible that an expensive action “close” to the end of the planning horizon is “pushed beyond” the horizon when an abstract action is replaced by an implementation and is replaced with a cheaper action for evaluation purposes. Therefore, for each time step remaining after reaching a node $q \in Q_f(C)$, we simply allocate the minimum cost of *any* action and 0 when the terminal node is reached. This way, the heuristic is guaranteed to underestimate the remaining costs. Note that FOFSCs are unrolled for evaluation and comparison for finite horizons.

5.2 Properties

The algorithm is correct for both the finite and indefinite horizon cases, because the requirements for the usage of A* search are met. Firstly, since all action costs are non-negative, applying a method never decreases the cost g of an FOFSC. Therefore, there are no negative path costs as required for A* search. Secondly, the heuristics are constructed to underestimate the real costs. Thirdly, because both the cost and the heuristic estimate only depend on the FOFSC at hand, not on the path that generated it, the usage of a closed-set is also allowed.

In the finite horizon case, it is also complete and guaranteed to terminate. This is because there are only finitely many unrolled FOFSCs whose length is limited by the horizon h . Since every possible FOFSC in the hierarchy is generated and no FOFSC is considered twice through the use of the closed-set D , the execution terminates.

Because of the infinite plan space in the indefinite horizon case, completeness cannot be guaranteed in this setting. For some special cases though, the algorithm is also complete and guaranteed to terminate, for example when the action hierarchy is non-recursive, i.e., an abstract action is never reintroduced after it has been decomposed.

6 Conclusion and Future Work

We have presented an approach that allows for exploitation of domain knowledge in relational POMDP planning. While the FOFSCs generated by our approach do not represent optimal policies for the underlying POMDP, good policies can be generated through the use of suitable expert knowledge in the form of methods. If an optimal policy for the underlying POMDP is required, the FOFSC generated by our approach can be used as an input for traditional, optimal POMDP solution algorithms such as value or policy iteration.

Our approach covers both finite and indefinite horizons. Especially in the context of user-centered planning, finite horizons seem appropriate to represent the limited patience of a human user. As a prerequisite for our approach, we introduced a first-order generalization of FSCs that also corresponds to a generalization of plans generated by traditional AI planning systems for fully observable deterministic domains.

As the next step, we plan to implement our formalism and evaluate its effectiveness in terms of plan generation speed and ease of modeling, using domains from the 2011 International Probabilistic Planning Competition IPPC-2011. We expect that expert knowledge required for our approach is available in domains like elevator control or traffic light control, where present-day solutions are hand-crafted by experts. Another important part in our research is the development of suitable heuristics to improve plan generation speed.

Acknowledgements. This work was done within the Transregional Collaborative Research Centre SFB/TRR 62 “Companion-Technology for Cognitive Technical Systems” funded by the German Research Foundation (DFG).

References

1. Biundo, S., Bercher, P., Geier, T., Müller, F., Schattenberg, B.: Advanced user assistance based on AI planning. In: Cognitive Systems Research, Special Issue on “Complex Cognition”, pp. 219–236 (2011)
2. Boger, J., Poupart, P., Hoey, J., Boutilier, C., Fernie, G., Mihailidis, A.: A decision-theoretic approach to task assistance for persons with dementia. In: IJCAI, pp. 1293–1299 (2005)
3. Bouguerra, A., Karlsson, L.: Hierarchical task planning under uncertainty. In: 3rd Italian Workshop on Planning and Scheduling (2004)
4. Erol, K., Hendler, J., Nau, D.S.: UMCP: A sound and complete procedure for hierarchical task-network planning. In: AIPS, pp. 249–254 (1994)
5. Hansen, E.A.: Solving POMDPs by searching in policy space. In: UAI, pp. 211–219 (1998)
6. Hansen, E.A.: Indefinite-horizon POMDPs with action-based termination. In: AAAI, pp. 1237–1242 (2007)
7. Hansen, E.A., Zhou, R.: Synthesis of hierarchical finite-state controllers for POMDPs. In: ICAPS, pp. 113–122 (2003)
8. Madani, O., Hanks, S., Condon, A.: On the undecidability of probabilistic planning and related stochastic optimization problems. Artificial Intelligence 147(1-2), 5–34 (2003)
9. Nau, D., Au, T.-C., Ilghami, O., Kuter, U., Muñoz-Avila, H., Murdock, J.W., Wu, D., Yaman, F.: Applications of SHOP and SHOP2. In: IEEE Intelligent Systems (2004)
10. Nau, D., Cao, Y., Lotem, A., Munoz-Avila, H.: SHOP: simple hierarchical ordered planner. In: IJCAI, pp. 968–973 (1999)
11. Papadimitriou, C.H., Tsitsiklis, J.N.: The complexity of Markov decision processes. Mathematics of Operations Research 12(3), 441–450 (1987)
12. Pednault, E.P.D.: ADL: exploring the middle ground between STRIPS and the situation calculus. In: KR, pp. 324–332 (1989)
13. Pineau, J., Gordon, G., Thrun, S.: Anytime point-based approximations for large POMDPs. JAIR 27, 335–380 (2006)
14. Puterman, M.L.: Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons, Chichester (1994)
15. Russell, J.A., Mehrabian, A.: Evidence for a three-factor theory of emotions. Journal of Research in Personality 11(3), 273–294 (1977)
16. Sanner, S., Kersting, K.: Symbolic dynamic programming for first-order POMDPs. In: AAAI, pp. 1140–1146 (2010)
17. Wang, C., Khordon, R.: Relational partially observable MDPs. In: AAAI, pp. 1153–1158 (2010)
18. Younes, H.L.S., Littman, M.L.: PPDDL 1.0: an extension to PDDL for expressing planning domains with probabilistic effects. Technical Report CMU-CS-04-167, Carnegie Mellon University, Pittsburgh (2004)

Gates for Handling Occlusion in Bayesian Models of Images: An Initial Study

Daniel Oberhoff¹, Dominik Endres², Martin A. Giese², and Marina Kolesnik¹

¹ Fraunhofer FIT-LIFE, Schloss Birlinghoven, St. Augustin, Germany

{daniel.oberhoff,marina.kolesnik}@fit.fraunhofer.de

² Section for Computational Sensomotorics, Dept. of Cognitive Neurology,
University Clinic, CIN, HIH and University of Tübingen,

Frondsbergstr 23, 72070 Tübingen, Germany

dominik.endres@klinikum.uni-tuebingen.de, martin.giese@uni-tuebingen.de

Abstract. Probabilistic systems for image analysis have enjoyed increasing popularity within the last few decades, yet principled approaches to incorporating occlusion *as a feature* into such systems are still few [11, 10, 7]. We present an approach which is strongly influenced by the work on *noisy-or* generative factor models (see e.g. [3]). We show how the intractability of the hidden variable posterior of *noisy-or* models can be (conditionally) lifted by introducing gates on the input combined with a sparsifying prior, allowing for the application of standard inference procedures. We demonstrate the feasibility of our approach on a computer vision toy problem.

1 Introduction

Both computer vision systems and models of (mammalian) biological vision have been researched extensively in the past few decades [4, 5, 9]. A key decision to make in the design of both types of system is which aspects of the visual environment are represented explicitly, and what details are to be disregarded by the system. The former determines the *specificity* of (the parts of) the system, while the latter are referred to as *invariances*. For example, invariance against shift, rotation, scaling and deformative transformations have been extensively modeled, which is due to these transformations being an ubiquitous part of the processes that generate natural images. We are concerned with another aspect of the image-generating process which has received less attention: *occlusion*. We argue, like [11, 10], that occlusion is an important (and frequent) enough aspect to warrant explicit modeling, rather than treating it as noise. In other words, we propose to treat occluded parts of objects in an image as unobserved data, rather than formulating a generative model which expects these parts to be visible (this would e.g. be the assumption underlying linear generative models, or standard *noisy-or* models).

2 The Gated Convolutional Model

Our model is a convolutional directed Bayesian model in which the image is generated from a set of discrete latent variables, each of which has connections

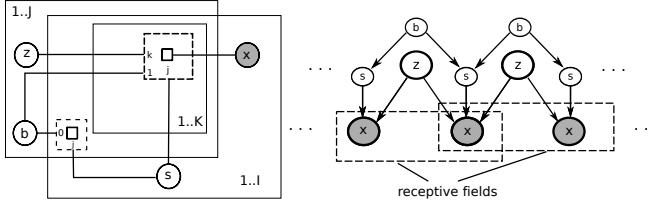


Fig. 1. Factor (*left*) and directed (*right*) graphs of a model layer with categorical factors and sparsity-promoting bits b . The factor graph uses plate notation and gates [8]. The observable data x are explained by the emission model (here: Gaussian clusters) associated with a latent variable z , if the gating variable s which connects x and z at a given point in the image is ‘on’. A gating variable s can only be ‘on’ for a given x/z combination if the corresponding b is ‘on’. We use a sparsity-promoting prior on b , i.e. most b are ‘off’ most of the time. Thus, the model will try to explain the image with few z only.

with a local receptive field in the image, where neighboring receptive fields are shifted by one pixel with respect to each other¹. This effectively makes it a mixture model over (overlapping) image patches of fixed size. The novelty of our approach lies in the fact that we do not combine the predictions of multiple mixture models converging on one input, nor do we infer about a global ordering over the generated image patches [10, 7]. Instead we introduce auxiliary variables which assign each input to a single mixture model only. This effectively implements an occlusion model because the image generated by one mixture model can ‘occlude’ parts of the image generated by a neighboring mixture model. In the following we will call these auxiliary variables ‘gates’. Interestingly, this approach corresponds to the common variational approximation to the posterior of the QMR-DT multiple-causes model [6], where similar selector variables are introduced, effectively turning the noisy-or into an exclusive-or. We employ mixture models with categorical latent variables mainly for computational simplicity; factor models [10, 7] could in principle also be used. To promote the forming of proper object hypotheses and to avoid simplistic solutions where each input pixel is modeled by a separate latent variable, we place sparsity promoting priors on the gates, encouraging them to use only few latent variables to explain the input.

The joint distribution for a layer with gated mixture models is:

$$\begin{aligned} p(\mathbf{x}, \mathbf{z}, \mathbf{s}, \mathbf{b}, \boldsymbol{\Theta}) &= \prod_{ijk} \left[p(x_i | \boldsymbol{\Theta}_{ijk})^{s_{ij} z_{jk}} p(z_j) \right]^{b_j} \\ &\times \prod_{ij} \mathbb{I}[b_j \vee (\neg s_{ij} \wedge \neg b_j)] p(\mathbf{b}) p(\mathbf{s}) p(\boldsymbol{\Theta}). \end{aligned} \quad (1)$$

¹ Our receptive fields are quadratic regions of the image and pixels contain all available channels of the image. For example, in an RGB image a pixel comprises a three-tuple with one real number per channel.

where i is an input variable index, j is a latent variable index, k is the mixture component index (components are shared among latent variables, i.e. the network is convolutional). Θ_{ijk} are the parameters of the emission model (here: Gaussian clusters with diagonal covariance) which connect the latent variables z_j to the observable data x_i . The latent variables z_j are vectors in 1-out-of- K encoding. Similarly, the gate variables s are comprised of I vectors in 1-out-of- J encoding. $p(\mathbf{z})$ and $p(s)$ are the priors over the latent and gate variables, respectively. We furthermore introduce one binomial variable b_j per latent variable, indicating the availability of latent variables for explaining the data. When the b_j associated with a latent variable is 'off', gate settings which assign an input to the latent variable become *forbidden*. By assigning a low prior probability to the 'on' state of each b_j , sparsity of the b_j is encouraged, therefore the inputs will be assigned to a small subset of latent variables. The second product with the indicator function ensures that no gate switches to a latent variable whose bit is off. $p(\mathbf{b})$ is a product of identical binomial distributions for each bit. The above graphical model is sketched in plate and gate notation (see [8] for an explanation of the gate notation) in figure [1]. Note that due to the convolutional nature of the network together with the possibility to 'pick' a subset of receptive fields for explaining the data, we can get shift invariance for free, since the gates can always choose a receptive field with the 'right' shift to explain some subset of input pixels (i.e. an 'object').

Inference in our model can be performed efficiently by blocked Gibbs sampling, since the latent variables are independent given the gates, and the gates are independent given the bits and the latent variables. During sampling gate proposals within one receptive field are generated together with proposals for the corresponding latent variable. Note that a latent variable whose bit is 'off' does not need to be updated, since it has no observable effects.

Learning the Parameters: we put conjugate exponential priors on all parameters and use the inferred latent and gate variable distributions to compute an approximate posterior. In effect, this is variational Bayesian expectation maximization (VBEM) [1]. For the mixture prior we use the truncated stick breaking approximation to the Dirichlet process, thus avoiding the need for random initialization and reducing the effect of the (arbitrary) choice of the number of available mixture components [2].

3 Results and Conclusion

We demonstrate the ability of the model to separate objects that have a constant shape but occlude each other in various configurations on a toy data set consisting of an artificially generated RGB image sequence in which three geometric shapes of red greed and blue color move randomly over a black background. The shapes occasionally occlude each other, examples are shown in figure [2] left². For this experiment receptive field sizes are chosen to match the size of the 'objects'.

² Here the input pixels thus consist of RGB-triples, and thus a separate gate variable exists for each such triple.

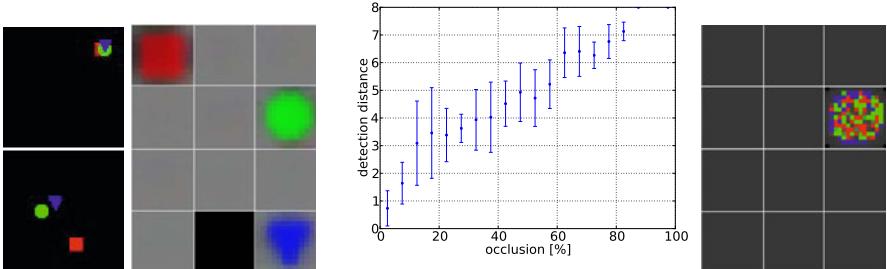


Fig. 2. *Left:* two example frames from the toy example sequence. The full sequence is one hundred frames long. Each frame has a size of 80x80 pixels. *Middle Left:* Expected receptive fields of the mixture components learned from the toy data. Due to the sparsity promoting prior the model is forced to explain the data with few latent variables, leading to a proper representation of the encountered objects and the background by separate mixture components. *Middle Right:* object localization accuracy on the toy data set. Localization degrades gracefully with increasing occlusion. *Right:* Expected receptive field of the mixture components learned without the sparsity prior (i.e. all latents are always available for explaining the input).

To promote sparse solutions, we assign a very low 'on' probability to each b_j ³. We use 5 Gibbs iterations per latent variable during inference. The model learns a separate component for each of the three objects and the background (see figure 2, middle left). For comparison we have also trained a model without the sparsity promoting prior, meaning that any latent can be used to explain the input at any time, and the resulting expected receptive fields are shown in figure 2, right. In this case a single high entropy component is learned, and the image is explained purely by the gates, i.e. no object concept is formed.

The model with sparsity promoting prior can now be used as an object detector: A detection is indicated by a latent variable, with enabled associated b_j , selecting one of the object representing components. The object location of this detector is defined as the most probable location of the original object image within the expected receptive field of the corresponding mixture component. In figure 2, middle right, we plotted the accuracy of this localization as a function of object occlusion. Localization performance degrades gracefully with increasing occlusion.

Conclusion: We have shown a feasible approach for modeling occlusion in a Bayesian image model. Its main features are a gated 'competition' between possible foreground explanations and strong sparsity promotion. We have demonstrated the viability of this approach by applying it to a toy data set, were we could also demonstrate the key role of the sparsity prior for forming stable object hypotheses.

³ We usually used a sparsity prior of 0.0001, but the exact value, as long as it is below the actual expected sparsity, did not make much of a difference.

Acknowledgments. this work was supported by EU projects FP7-ICT-215866 SEARISE, FP7-249858-TP3 TANGO and FP7-ICT-248311 AMARSi.

References

1. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Heidelberg (2006)
2. Blei, D.M., Jordan, M.I.: Variational methods for the Dirichlet process. In: Proceedings of the 21st International Conference on Machine Learning (2004)
3. Courville, A., Eck, D., Bengio, Y.: An infinite factor model hierarchy via a noisy-or mechanism. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) Advances in Neural Information Processing Systems, vol. 22, pp. 405–413 (2009)
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR, vol. (1), pp. 886–893 (2005)
5. Földfak, P.: Learning invariance from transformation sequences. Neural Computation 3, 194–200 (1991)
6. Jaakkola, T.S., Jordan, M.I.: Variational probabilistic inference and the qmr-dt network. J. Artif. Int. Res. 10, 291–322 (1999),
<http://portal.acm.org/citation.cfm?id=1622859.1622869>
7. Lücke, J., Turner, R., Sahani, M., Henniges, M.: Occlusive components analysis. In: Proceedings of NIPS, vol. 22, pp. 1069–1077 (2009)
8. Minka, T., Winn, J.: Gates: A graphical notation for mixture models. In: Proceedings of NIPS, vol. 21, pp. 1073–1080 (2008)
9. Olshausen, B.A., Field, D.J.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature 381(6583), 607–609 (1996)
10. Roux, N.L., Heess, N., Shotton, J., Winn, J.: Learning a generative model of images by factoring appearance and shape. Tech. rep., Microsoft Research (2010)
11. Tamminen, T., Lampinen, J.: A Bayesian occlusion model for sequential object matching. In: Proc. British Machine Vision Conference 2004, pp. 547–556 (2004)

TGA-Based Controllers for Flexible Plan Execution

Andrea Orlandini¹, Alberto Finzi², Amedeo Cesta³, and Simone Fratini³

¹ CNR – Consiglio Nazionale delle Ricerche, ITIA, Milan, Italy

² DSF – Università Federico II, Naples, Italy

³ CNR – Consiglio Nazionale delle Ricerche, ISTC, Rome, Italy

Abstract. Plans synthesized by Temporal Planning and Scheduling systems may be temporally flexible hence they identify an envelope of possible solutions. Such flexibility can be exploited by an executive systems for robust on-line execution. Recent works have addressed aspects of plan execution using a quite general approach grounded on formal modeling and formal methods. The present work extends such an approach by presenting the formal synthesis of a plan controller associated to a flexible temporal plan. In particular, the controller synthesis exploits Timed Game Automata (TGA) for formal modeling and UPPAAL-TIGA as a model checker. After presenting a formal extension, the paper introduces a detailed experimental analysis on a real-world case study that demonstrates the viability of the approach. In particular, it is shown how the controller synthesis overhead is compatible with the performance expected from a short-horizon planner.

1 Introduction

Robust plan execution in uncertain and dynamic environments is a critical issue for plan-based autonomous systems (see [10] and many others). Once a planner has generated a temporal plan, it is up to the executive system to decide, at run-time, how and when to execute each planned activity preserving both plan consistency and controllability. Such a capability is even more crucial when the generated plan is temporally flexible being a flexible temporal plan partially specified. Such a plan captures an envelope of potential behaviors to be instantiated during the execution taking into account temporal/causal constraints and controllable/uncontrollable activities and events. Previous works have tackled these issues within a Constraint-based Temporal Planning (CBTP) framework deploying specialized techniques based on temporal-constraint networks. Several authors [14][13][17] proposed a *dispatchable execution* approach where a flexible temporal plan is then used by a plan executive that schedules activities on-line while guaranteeing constraint satisfaction.

In this paper, we propose an alternative and novel approach to flexible plan dispatching/execution which is based on formal modeling and controller synthesis using Timed Game Automata (TGA). Such a technique is a direct consequence of the formalization proposed in [5]. Analogously to that work, the generated flexible temporal plan and the dynamic domain are encoded into TGA models. However, here, a different perspective is explored exploiting the model checker to directly synthesize a real-time plan controller for the flexible plan. Such controller guarantees plan execution along with other

domain dependent properties. In contrast to the dispatchable execution approach, our synthesized controller works here as an on-line generated dispatcher where all the decisions are already predefined, hence the run-time constraints check and propagation effort is not needed anymore, with a consequent reduction of the plan execution latency. As a potential drawback, we have to consider the additional overhead due to the synthesis of the controller itself which can be too expensive when deployed at run-time within a planning and execution cycle. For this reason, this paper, after presenting the new formal properties, discusses the practical applicability of the approach in a realistic robotic application scenario. The collected results are quite interesting. They not only show the feasibility of the approach in the real-world scenario but also suggest interesting relations among planning tasks, generated flexible plans, and generated controllers.

In literature, analogous formal methods have been applied to plan synthesis and plan verification, but they never address flexible temporal plan execution. Closely related to our work, in [1] plan synthesis with TGAs is proposed and contrasted with flexible plan generation through CBTP. As a difference, we deploy control synthesis with TGAs to generate a controller for a CBTP flexible plan. In [2] timed automata are exploited for incremental verification within the plan generation process. In [5] the authors propose UPPAAL-TIGA for flexible plan verification task, but neither plan control synthesis nor plan execution are addressed.

2 Plan-Based Robot Control

Our interest in plan-based autonomy is motivated by the GOAC project. We are developing a Goal Oriented Autonomous Controller [4] for the European Space Agency (ESA) integrating different software solutions. In particular: (a) a timeline-based deliberative layer which integrates a planner based on the APSI Platform [6] and an executive a la T-REX [16]; (b) a functional layer which integrates G^{en}_oM and BIP [2]. However, independent on our current use, the work described in this paper is valid for any generic three layered control architecture [8] that integrates a temporal planning and scheduling system.

The Robotic Domain. In the paper, we use a real-world running example taken from the GOAC project¹. Let us consider a planetary rover equipped with a Pan-Tilt Unit (PTU), two stereo cameras (mounted on top of the PTU) and a communication facility. The rover is able to autonomously navigate the environment, move the PTU, take pictures and communicate images to a Remote Orbiter. Finally, during the mission, the Orbiter may be not visible for some periods. Thus, the robotic platform can communicate only when the Orbiter is visible. The mission goal is a list of required pictures to be taken in different locations with an associated PTU configuration. A possible mission action sequence is the following: navigate to one of the requested locations, move the PTU pointing at the requested direction, take a picture, then, communicate the image to the orbiter during the next available visibility window, put back the PTU in the safe position and, finally, move to the following requested location. Once all the locations

¹ Thanks to Felix Ingrand and Lavindra De Silva from LAAS-CNRS for the time spent to explain us the details of their robotic platform.

have been visited and all the pictures have been communicated, the mission is considered successfully completed. The rover must operate following some operative rules to maintain safe and effective configurations. Namely, the following conditions must hold during the overall mission: **(C1)** While the robot is moving the PTU must be in the safe position (pan and tilt at 0); **(C2)** The robotic platform can take a picture only if the robot is still in one of the requested locations while the PTU is pointing at the related direction; **(C3)** Once a picture has been taken, the rover has to communicate the picture to the base station; **(C4)** While communicating, the rover has to be still; **(C5)** While communicating, the orbiter has to be visible.

3 Timeline-Based Planning and Execution

Timeline-based planning is an approach to temporal planning that has been applied in the solution of several real world problems – e.g., [15]. The approach pursues a general idea that planning and scheduling for controlling complex physical systems consists in the synthesis of desired temporal behaviors (or *timelines*).

State Variables and Timelines. According to this paradigm a domain is modeled as a set of features with an associated set of temporal functions on a finite set of values. The time varying features are called *multi-valued state variables* as in [15]. As in classical control theory, the evolution of the features is described by some causal laws and limited by domain constraints. These are specified in a *domain specification*. The task of a planner is to find a sequence of decisions that brings the timelines into a final desired set always satisfying the domain specification and special conditions called *goals*. We assume that the temporal features have a finite set of possible values assumed over temporal intervals. The temporal evolutions are sequences of operational states. Causal and temporal constraints specify which value transitions are allowed, the duration of each valued interval and synchronization constraints between different state variables.

More formally, a state variable is defined by a tuple $\langle \mathcal{V}, \mathcal{T}, \mathcal{D} \rangle$ where: (a) $\mathcal{V} = \{v_1, \dots, v_n\}$ is a finite set of *values*; (b) $\mathcal{T} : \mathcal{V} \rightarrow 2^{\mathcal{V}}$ is the *value transition* function; (c) $\mathcal{D} : \mathcal{V} \rightarrow \mathbb{N} \times \mathbb{N}$ is the *value duration* function, i.e. a function that specifies the allowed duration of values in \mathcal{V} (as an interval $[lb, ub]$). (b) and (c) specify the operational constraints on the values in (a). Given a state variable, its associated timeline is represented as a sequence of values in the temporal interval $\mathcal{H} = [0, H]$. Each value satisfies previous (a-b-c) specifications and is defined on a set of not overlapping time intervals contained in \mathcal{H} .

Timeline Specification for the Robotic Domain. To obtain a timeline-based specification of our robotic domain, we consider two types of state variables: *Planned State Variables* to represent timelines whose values are decided by the planning agent, and *External State Variables* to represent timelines whose values over time can only be observed. Planned state variables are those representing time varying features like the temporal occurrence of navigation, PTU, camera and communication operations. We use four of such state variables, namely the *RobotBase*, *PTU*, *Camera* and *Communication*.

In Fig. 1, we detail the values that can be assumed by these state variables, their durations and the legal value transitions in accordance with the mission requirements and the robot physics.² Additionally, one external state variable represents contingent events, i.e., the communication opportunities. The *Orbiter Visibility* state variable maintains the visibility of the orbiter. The allowed values for this state variable is *Visible* or *NotVisible*

and are set as an external input. The robot can be in a position (*At(x,y)*) or moving towards a destination (*GoingTo(x,y)*). The PTU can assume a *PointingAt(pan,tilt)* value if pointing a certain direction, while, when moving, it assumes a *MovingTo(pan,tilt)*. The camera can take a picture of a given object in a position $\langle x, y \rangle$ with the PTU in $\langle pan, tilt \rangle$ and store it as a file in the on-board memory (*TakingPicture(file-id,x,y,pan,tilt)*) or be idle (*CamIdle()*). Similarly, the communication facility can be operative and dumping a given file (*Communicating(file-id)*) or be idle (*ComIdle()*).

Representing Domain Causality. Domain operational constraints are described by means of *synchronizations*. A synchronization models the existing temporal and causal constraints among the values taken by different timelines (i.e., patterns of legal occurrences of the operational states across the timelines).

Fig. 2 exemplifies the use of synchronizations implementing the operative rules (see Section 2) in our case study domain. The synchronizations depicted are: *GoingTo(x,y)* must occur during *PointingAt(0,0)* (C1); *TakingPicture(pic,x,y,pan,tilt)* must occur during *At(x,y)* and *PointingAt(pan,tilt)* (C2); *TakingPicture(pic,x,y,pan,tilt)* must occur before *Communicating(pic)* (C3); *Communicating(file)* must occur during *At(x,y)* (C4); *Communicating(file)* must occur during *Visible* (C5). In addition to those synchronization constraints, the timelines must respect transition constraints among values and durations for each value specified in the domain (see again Fig. 1).

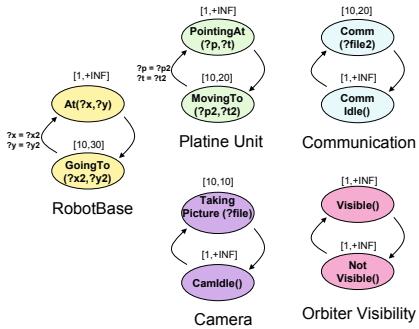


Fig. 1. State variables describing the robotic platform and the orbiter visibility (durations are stated in seconds)

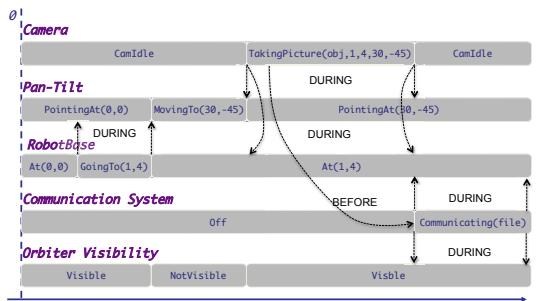


Fig. 2. An example of timeline-based plan with synchronizations

² Note that variables (e.g., $?x$) represents parameters with values in a finite set of symbols, used to compactly represent the allowed values for a given state variable. Moreover, the symbol $+INF$ is used to state the upper bound of the temporal interval $[0,H]$.

Timeline-Based Planning. *Planning goals* are expressed as desired timeline values in temporal intervals; the planning task is to build a set of timelines that describe valid sequences of values that achieve the desiderata. Hence, a *plan* is a set of *timelines*, that is, a sequence of state variable values, a set of ordered transition points between the values, and a set of distance constraints between transition points. When the transition points are bounded by the planning process (lower and upper bounds are given for them) instead of being exactly specified, we refer to the timeline as *time flexible* while a *flexible plan* is the plan resulting from a set of flexible timelines. A flexible plan defines a set of admissible temporal behaviors. Considering a partial horizon H' (with $H' < H$), the same flexible plan defines a set of *partial* temporal behaviors \mathcal{PB} . A *flexible plan* $\mathcal{P} = \{TL_1, \dots, TL_n\}$ is defined over a given horizon \mathcal{H} . A solution plan is *valid* with respect to a domain theory if every temporal occurrence of a value satisfies the expected synchronizations.

Plan Execution. During plan execution, the plan, or a partial segment of it, is under responsibility of the executive system that forces value transitions over the timelines dispatching commands to the functional layers while continuously accepting observations and, thus, monitoring the plan execution. Additionally, not all the value transitions are under responsibility of the executive, but events exist that are under control of the *environment*. In such cases, the values for the controllable state variables should be chosen so that they do not constrain uncontrollable events. This is the *controllability problem* ([18]). Controllability issues underlying a plan representation have been formalized and investigated for the Simple Temporal Problems with Uncertainty (STPU) representation in [18] where basic formal notions are given for *dynamic* controllability (see also [13]). In [5] these notions have been extended to the timeline-based framework.

In order to endow the executive system with an execution strategy, a plan controller is needed taking into account also the controllability problem. That is, the executive system is to robustly execute a flexible temporal plan. More formally, a *plan controller* \mathcal{C} is a partial function from the set of partial behaviors \mathcal{PB} and possible horizons to the set of controllable values for state variables plus a special action λ representing the *wait* action as for TGA (see the following section), $\mathcal{C} : \mathcal{PB} \times \mathbb{N} \rightarrow \mathcal{V}_1 \cup \dots \cup \mathcal{V}_n \cup \{\lambda\}$.

4 Timed Game Automata and Controllers

Timed game automata (TGA) have been introduced in [11] to model control problems on timed systems [3], in this section we briefly recall some definitions that we shall use in the rest of the paper.

Definition 1. A **Timed Game Automaton** is a tuple $\mathcal{A} = (Q, q_0, \text{Act}, X, \text{Inv}, E)$ where: Q is a finite set of locations; $q_0 \in Q$ is the initial location; Act is a finite set of actions split in two disjoint sets, Act_c the set of controllable actions and Act_u the set of uncontrollable actions; X is a finite set of non-negative, real-valued variables called clocks; $\text{Inv} : Q \rightarrow B(X)$ is a function associating to each location $q \in Q$ a constraint $\text{Inv}(q)$ (the invariant of q); $E \subseteq Q \times B(X) \times \text{Act} \times 2^X \times Q$ is a finite set of transitions. Where $B(X)$ is the set of constraints in the form $x \sim c$, where $c \in \mathbb{Z}$, $x \in X$, and $\sim \in \{<, \leq, \geq, >\}$. We write $q \xrightarrow{g,a,Y} q' \in E$ for $(q, g, a, Y, q') \in E$.

A *state* of a TGA is a pair $(q, v) \in Q \times \mathbb{R}_{\geq 0}^X$ that consists of a discrete part and a valuation of the clocks (i.e., a value assignment for each clock in X). An *admissible state* for an automaton \mathcal{A} is a state (q, v) s.t. $v \models \text{Inv}(q)$. From a state (q, v) a TGA can either let time progress or do a discrete transition and reach a new state. A *time transition* for \mathcal{A} is 4-tuple $(q, v) \xrightarrow{\delta} (q, v')$ where $(q, v) \in S$, $(q, v') \in S$, $\delta \in R_{\geq 0}$, $v' = v + \delta$, $v \models \text{Inv}(q)$ and $v' \models \text{Inv}(q)$. That is, in a time transition a TGA does not change location, but only its clock values. Note that all clock variables are incremented by the same amount δ in valuation v' . This is why variables in X are named *clocks*. Accordingly, δ models the *elapsed time* during the time transition. A *discrete transition* for \mathcal{A} is 5-tuple $(q, v) \xrightarrow{a} (q', v')$ where $(q, v) \in S$, $(q', v') \in S$, $a \in \text{Act}$ and there exists a transition $q \xrightarrow{g, a, Y} q' \in E$ s.t. $v \models g$, $v' = v[Y]$ and $v' \models \text{Inv}(q')$. In other words, there is a discrete transition (labeled with a) from state (q, v) to state (q', v') if the clock values (valuation v) satisfy the *transition guard* g and the clock values after resetting the clocks in Y (valuation v') satisfy the invariant of location q' . Note that an admissible transition always leads to an admissible state and that only clocks in Y (reset clocks) change their value (namely, to 0). A *run* of a TGA \mathcal{A} is a finite or infinite sequence of alternating time and discrete transitions of \mathcal{A} . We denote with $\text{Runs}(\mathcal{A}, (q, v))$ the set of runs of \mathcal{A} starting from state (q, v) and write $\text{Runs}(\mathcal{A})$ for $\text{Runs}(\mathcal{A}, (q_0, \mathbf{0}))$. If ρ is a finite run, we denote with $\text{last}(\rho)$ the last state of run ρ and with $\text{Duration}(\rho)$ the sum of the elapsed times of all time transitions in ρ . A *network* of TGA (nTGA) is a finite set of TGA evolving in parallel with a CCS style semantics for parallelism [12]. Namely, at any time, only one TGA in the network can change location, unless a synchronization on labels takes place. In the latter case, the two automata synchronizing on the same label move together. Note that time does not elapse during synchronizations.

Given a TGA \mathcal{A} and three symbolic configurations *Init*, *Safe*, and *Goal*, the *reachability control problem* or *reachability game* $RG(\mathcal{A}, \text{Init}, \text{Safe}, \text{Goal})$ consists in finding a *strategy* f such that \mathcal{A} starting from *Init* and supervised by f generates a winning run that stays in *Safe* and enforces *Goal*. A strategy is a partial mapping f from the set of runs of \mathcal{A} starting from *Init* to the set $\text{Act}_c \cup \{\lambda\}$ (λ is a special symbol that denotes "do nothing and just wait"). For a finite run ρ , the strategy $f(\rho)$ may say (1) no way to win if $f(\rho)$ is undefined, (2) do nothing, just wait in the last configuration ρ if $f(\rho) = \lambda$, or (3) execute the discrete, controllable transition labeled by l in the last configuration of ρ if $f(\rho) = l$. The restricted behavior of a TGA \mathcal{A} controlled with some strategy f is defined by the notion of *outcome*. The outcome $\text{Outcome}(q, f)$ is defined as the subset of $\text{Runs}(\Pi, \mathcal{A})$ that can be generated from q executing the uncontrollable actions in Act_u or the controllable actions provided by the strategy f . A *maximal run* ρ is either an infinite run or a finite run that satisfies either i) $\text{last}(\rho) \models \text{Goal}$ or ii) if $\rho \xrightarrow{a} \text{then } a \in \text{Act}_u$ (i.e. the only possible next discrete actions from $\text{last}(\rho)$, if any, are uncontrollable actions). A strategy f is a *winning strategy* from q if all maximal runs in $\text{Outcome}(q, f)$ are in $\text{WinRuns}(q, \mathcal{A})$. A state q in a TGA \mathcal{A} is *winning* if there exists a winning strategy f from q in \mathcal{A} . We denote by $W(\mathcal{A})$ the set of winning states of \mathcal{A} .

4.1 The Encoding as nTGA Model

As discussed before, TGAs allow to model real-time systems and controllability problems representing uncontrollable activities as *adversary moves* within a game between

the controller and the environment. Following the approach presented in [5], flexible timeline-based plan verification can be performed by solving a Reachability Game using UPPAAL-TIGA. To this end, we compile flexible timeline-based plans, state variables, and domain theory descriptions into a set of TGA (nTGA). This is obtained with the following steps: (1) a flexible timeline-based plan \mathcal{P} is mapped into a nTGA *Plan*. Each timeline is encoded as a sequence of locations (one for each timed interval), while transition guards and location invariants are defined according to (respectively) lower and upper bounds of flexible timed intervals; (2) the set of state variables SV is mapped into a nTGA *StateVar*. Basically, we define a one-to-one mapping from state variables descriptions to TGA. In this encoding, value transitions are partitioned into controllable and uncontrollable. (3) an *Observer* automaton is introduced to check for value constraints violations and synchronizations violations. In particular, we have two locations: an Error location, to state constraint/synchronization violations, and a Nominal (OK) location, to state that the plan behavior is correct. The *Observer* is defined as fully uncontrollable. (4) the nTGA \mathcal{PL} composed by the set of automata $StateVar \cup Plan \cup \{\mathcal{A}_{Obs}\}$ encapsulates flexible plan, state variables and domain theory descriptions.

Given such an encoding, the following theorems have been demonstrated (in [5]).

Theorem 1. *The nTGA \mathcal{PL} describes all and only the behaviors implemented by the flexible plan \mathcal{P} .*

That is, the nTGA \mathcal{PL} models a flexible plan along with state variables and domain theory descriptions.

Theorem 2. *Given $RG(\mathcal{PL}, Init, Safe, Goal)$ defined considering $Init$, $Safe$ and $Goal$ as above, the existence of a winning strategy implies plan validity for \mathcal{P} .*

Considering a Reachability Game $RG(\mathcal{PL}, Init, Safe, Goal)$ where *Init* represents the set of the initial locations of each automaton in \mathcal{PL} , *Safe* is the Observer's OK location, and *Goal* is the set of goal locations, one for each automaton in *Plan*, plan verification can be performed solving the $RG(\mathcal{PL}, Init, Safe, Goal)$ defined above. If there is no winning strategy, UPPAAL-TIGA provides a counter strategy for the opponent (i.e., the environment) to make the controller lose. That is, an execution trace showing a faulty evolution of the plan is provided.

4.2 Synthesizing Controllers

This section provides the formal definitions for synthesizing plan controllers as a direct extension of the previous theorems.

Theorem I defines a one-to-one mapping between flexible temporal behaviors over $[0, H]$ defined by \mathcal{P} and the automata behaviors defined by \mathcal{PL} . This property can be directly extended to partial plans. Indeed, since Theorem I holds from any horizon H , for each partial temporal behavior $pb \in \mathcal{PB}$ defined over $H' < H$, there exists a unique run ρ_{pb} of \mathcal{PL} such that ρ_{pb} represents the temporal behavior pb over the same horizon H' . That is, ρ_{pb} of \mathcal{PL} represents the same valued intervals sequence in \mathcal{P} limited to H' and $Duration(\rho_{pb})$ is exactly the horizon H' .

Analogously, by extending Theorem I to partial plans, the winning strategy obtained as a side effect of the verification process represents a flexible plan controller that

achieves the planning goals maintaining the dynamic controllability during the overall plan execution. More formally, a plan controller \mathcal{C}_f derived from a winning strategy f can be defined as follows.

Definition 2. Given the reachability game $RG(\mathcal{PL}, \text{Init}, \text{Safe}, \text{Goal})$ defined as above, and the winning strategy f generated by UPPAAL-TIGA, a plan controller \mathcal{C}_f is defined as follows: for each partial behavior $pb \in \mathcal{PB}$ over H' , $\mathcal{C}_f(pb, H') = f(\rho_{pb})$, where each action $a \in \text{Act}_c \cup \{\lambda\}$ represents the associated values in $\mathcal{V}_1 \cup \dots \cup \mathcal{V}_n \cup \{\lambda\}$, $\mathcal{C}_f(pb, H')$ is undefined otherwise.

As a consequence, the following theorem holds:

Theorem 3. A controller \mathcal{C}_f defined according to Definition 2 satisfies the following: (i) it correctly executes the plan \mathcal{P} reaching the given planning goals and (ii) maintains the dynamic controllability property during plan execution.

Moreover, it is also possible to define optimized controllers for flexible plans. Given a fixed temporal interval $[u, g]$ and a reachability game, UPPAAL-TIGA is able to generate a winning strategy f^* within that interval which minimize the plan execution duration [3]. Since a flexible plan is associated with a planning horizon $[0, H]$, an optimized controller can be generated with $[u, g] = [0, H]$. This allows to conclude the following:

Theorem 4. Given a reachability game $RG(\mathcal{PL}, \text{Init}, \text{Safe}, \text{Goal})$ defined as above within the temporal interval $[u, g] = [0, H]$, the winning strategy f^* provided by UPPAAL-TIGA is time optimal and, because of Theorem 3 the derived controller \mathcal{C}_{f^*} is also time optimal.

5 Empirical Results

This section investigates the practical feasibility of the approach by using our robotic case study as a benchmark. Our aim is to test the controller generation performance in a real world scenario to check whether on-line control synthesis is viable and compatible with the short latencies of a planning and execution cycle. In this context, we want also to assess the controller synthesis overhead w.r.t. to the planning and verification costs. For this purpose, we introduce different planning/execution scenarios obtained by varying the problem complexity along the following dimensions: *plan length* by playing on both the number of pictures to be taken and the plan horizon; *plan flexibility* by modifying the allowed temporal tolerance for uncontrollable actions; *plan choices* by changing the number of communication opportunities. More specifically:

(1) *Plan Length.* We considered problem instances with an increasing number of requested pictures (from 1 to 5). At the same time, we consider flexible plans with a horizon length ranging from 150 to 550 seconds.

(2) *Plan Flexibility.* For each uncontrollable activity (i.e., GoingTo, MovingTo, TakingPicture, and Communicating), we set a minimal duration, but allow temporal flexibility on the activity termination, namely, the end of each activity has a tolerance ranging

from 0 to 20 seconds. This temporal interval represents the degree of temporal flexibility/uncertainty that we introduce in the system.

(3) *Plan Choices*. We define from 1 to 4 visibility windows that can be exploited to communicate picture content.

Notice that an increasing number of communication opportunities raises the complexity of the planning problem with a combinatorial effect. More in general, among all the generated problem instances, the ones with higher number of required pictures, higher temporal flexibility, and higher number of visibility windows result as the hardest ones. In these scenarios, we analyzed the performance of our method considering model generation, controller synthesis, and plan execution. We used OMPS [7] as a CBTP Domain Independent Planner. The experiments have been ran on a MacBook Pro endowed with a Intel Core i5 (2.5GHz) processor and 4GB RAM. In what follows the reported timings are in seconds.

Model Generation. As a preliminary step of our evaluation, we considered the cost of generating the UPPAAL-TIGA model associated with the planning task. The dimension of the generated model is given in terms of number of generated states and the file size in bytes. As we can see in Fig. 3, for all the configurations, the generation process is very fast, taking less than 200ms for each instance, while the dimension of the generated model gradually grows with respect to the dimension of the flexible plan in terms of both plan length and number of visibility windows. The temporal flexibility does not affect the dimension of the generated models. Thus, we can conclude that model generation is not a critical step in our method.

| | | 1 wind | 2 wind | 3 wind | 4 wind |
|--------|---------------|--------|--------|--------|--------|
| #pic 1 | bytes | 8108 | 8108 | 8671 | 8960 |
| H 150 | nr. of states | 29 | 29 | 33 | 35 |
| #pic 2 | bytes | 10094 | 10370 | 10674 | 10936 |
| H 250 | nr. of states | 39 | 39 | 43 | 45 |
| #pic 3 | bytes | 13051 | 13326 | 13603 | 13892 |
| H 350 | nr. of states | 53 | 53 | 57 | 59 |
| #pic 4 | bytes | 14102 | 14378 | 14655 | 14943 |
| H 450 | nr. of states | 59 | 63 | 65 | 69 |
| #pic 5 | bytes | 18151 | 16402 | 16678 | 16967 |
| H 550 | nr. of states | 69 | 70 | 73 | 75 |

Fig. 3. Size of generated models

Controller Synthesis. The cost of controller synthesis has been analyzed with respect to the cost of planning and the cost of plan verification (i.e., dynamic controllability check). The planning costs are collected in Fig. 4 where we can observe how the planner performance decreases with increasing communication windows and temporal flexibility. In particular, while simple instances (i.e., 1 or 2 communication windows) are solved in few seconds,

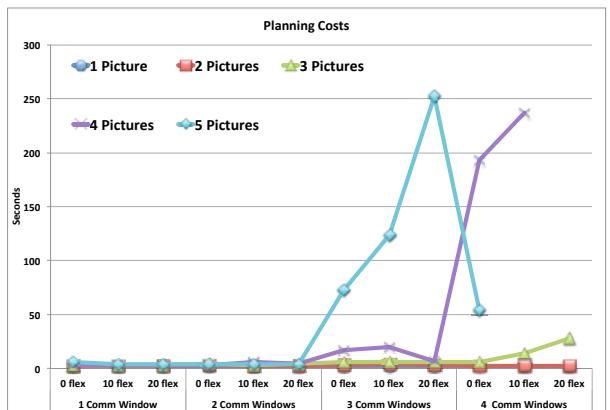


Fig. 4. Plan generation cost varying the number of pictures, visibility windows and temporal flexibility

the hardest ones require an additional planning effort (i.e., 4 communication windows and more than 3 required pictures). Actually, in this case some of the instances are not solved due to memory limit (in Fig. 4 values are missing for the last problem instances with 4 and 5 requested pictures).

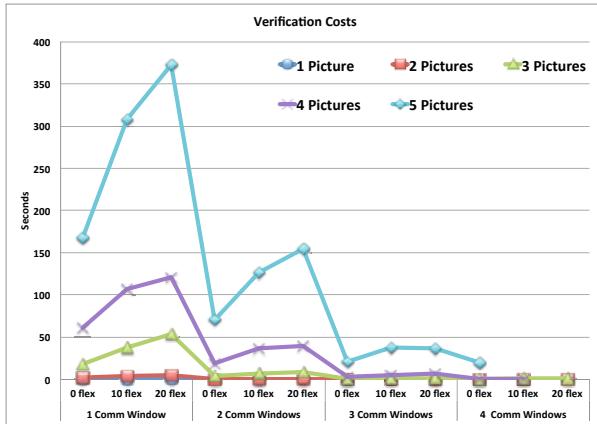


Fig. 5. Dynamic Controllability verification cost

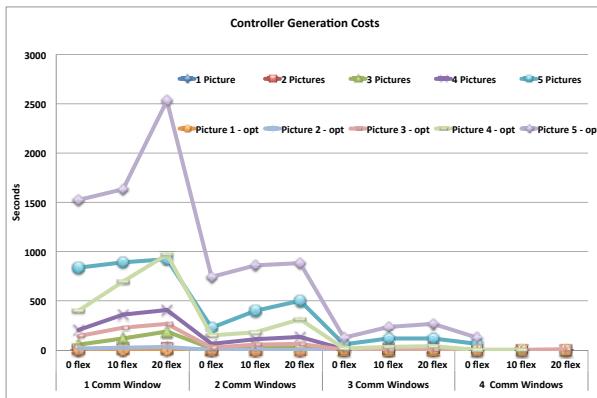


Fig. 6. Strategies generation cost for both optimized and non optimized cases

simplifies the UPPAAL-TIGA task which can check and generate strategies more quickly (see again the values for 4 communication windows in Fig. 5 and 6). Indeed, this is an expected behavior of the verification tool. In fact, the more non-determinism, the harder it is for UPPAAL-TIGA to generate strategies.

Plan Execution. As a final evaluation of the generated plan controllers, we considered the time needed for plan execution comparing optimized and non-optimized controllers. Besides providing empirical indications of the controllers effectiveness, our aim here is

The results collected for dynamic controllability checking (see Fig. 5) and strategies generation (see Fig. 6) show a quite different behavior. Interestingly, for hard problem instances flexible plan verification and strategy generation are very fast (3 and 4 communication windows in Fig. 5 and 6). While, with simpler instances (1 and 2 communication windows in Fig. 5 and 6), we do not observe the expected improvement in performance.

This is mainly due to the fact that simple planning problem instances are associated with few constraints to be considered, hence our planner can generate highly flexible temporal plans. However, this flexibility provides a wide search space to the verification tool reducing its performance. In contrast, harder planning problems lead the planner to produce flexible plans that are strongly constrained, i.e., with a lower degree of flexibility. This

| 1 Comm. Window | | | 2 Comm. Windows | | | 3 Comm. Windows | | | 4 Comm. Windows | | |
|----------------|---------|----------|-----------------|-----|---------|-----------------|----------|-----|-----------------|----------|----------|
| pic | 0s flex | 10s flex | 20s flex | pic | 0s flex | 10s flex | 20s flex | pic | 0s flex | 10s flex | 20s flex |
| 1 | 139±0 | 146±3 | 148±1 | 1 | 131±0 | 142±7 | 141±3 | 1 | 132±0 | 137±6 | 145±3 |
| 2 | 243±0 | 211±6 | 243±6 | 2 | 198±0 | 232±13 | 238±11 | 2 | 213±0 | 231±8 | 230±8 |
| 3 | 242±0 | 291±2 | 339±7 | 3 | 238±0 | 313±5 | 336±9 | 3 | 231±0 | 284±6 | 337±12 |
| 4 | 431±0 | 427±5 | 542±7 | 4 | 421±0 | 415±10 | 437±8 | 4 | 423±0 | 401±6 | 423±6 |
| 5 | 535±0 | 537±9 | 542±7 | 5 | 507±0 | 527±8 | 536±12 | 5 | 538±0 | 525±7 | 528±10 |
| Optimal | | | Optimal | | | Optimal | | | Optimal | | |
| 1 | 98±0 | 118±7 | 132±4 | 1 | 81±0 | 97±6 | 121±8 | 1 | 78±0 | 87±4 | 108±3 |
| 2 | 173±0 | 194±11 | 229±16 | 2 | 167±0 | 211±9 | 218±13 | 2 | 145±0 | 176±17 | 201±7 |
| 3 | 237±0 | 286±9 | 332±12 | 3 | 231±0 | 307±8 | 327±4 | 3 | 227±0 | 279±4 | 331±14 |
| 4 | 428±0 | 428±6 | 432±7 | 4 | 418±0 | 411±11 | 430±12 | 4 | 420±0 | 397±12 | 421±7 |
| 5 | 512±0 | 527±14 | 531±9 | 5 | 494±0 | 518±6 | 521±10 | 5 | 528±0 | 511±8 | 507±9 |

| (a) | (b) |
|-----|-----|
|-----|-----|

Fig. 7. Plan controllers execution performance (average durations and variances)

also to assess whether the gain in the execution performance in the optimized case can justify the generation cost overhead. The execution has been simulated in UPPAAL-TIGA considering time average and variance of 20 runs and randomly generating the temporal occurrences (within their duration intervals) of the uncontrollable events mentioned in the plan. In Fig. 7 the collected results show a slight enhancement in time efficiency in the optimized version, but this gain seems negligible, in particular when compared with the generation cost. This seems to suggest that sub-optimal controllers provide a better trade-off between control synthesis and plan execution.

Discussion. The experimental results show the practical feasibility of the TGA approach in increasingly complex instances of a real-world robotic case study. The collected data show also an interesting relationship among the complexities of the planning tasks, generated flexible plans, and generated controllers. We observe that additional efforts during the planning phase usually reduces the cost of plan verification and control synthesis and, vice-versa, simpler planning problems are usually associated with more complex controller synthesis tasks. Furthermore, more complex planning tasks should be associated with longer execution latencies, hence if we contrast the controller generation time w.r.t. the planning horizon length (assuming a comparable time available for planning), taking apart the hardest instances (5 pictures, and 3 or 4 pictures with less than 3 visibility windows), all the other cases are treatable. Moreover, if we consider plan complexity (visibility windows) and control synthesis cost overhead with respect to the plan generation cost, we observe that with few required pictures (e.g., 1 or 2 pictures), the control synthesis cost remains acceptable for all the visibility windows, while, with additional visibility windows (e.g., 4 visibility windows), additional pictures can be introduced. In conclusion, this empirical analysis shows how the control synthesis overhead remains very low in most of the considered instances. In particular, the controller synthesis method is compatible with the performance required by a fast short-horizon planner.

6 Conclusion

The paper has introduced a formal method to automatically synthesize controllers for flexible temporal plans. While flexible temporal plan execution is usually addressed

using temporal constraint networks methods and algorithms to reduce the plan in a dispatchable form, this paper proposes an alternative and novel technique based on the generation of a winning strategy with TGAs. According to this approach, the plan execution problem can be solved completely as a side effect of dynamic controllability checking; hence, all the plan execution decisions can be available before the plan execution with an acceptable overhead with respect to the planning activity. It is worth mentioning how this method relies foremost on off-the-shelf planning/verification tools such as the OMPS and UPPAAL-TIGA tool-chain and on an encoding tool tailored to translate the plan specification into TGAs.

Acknowledgment. Cesta, Fratini and Orlandini are partially supported by EU under the ULISSE project (Contract FP7.218815), and by MIUR under the PRIN project 20089M932N (funds 2008). Finzi is partially supported by EU under the AIRobots project (Contract FP7.248669). Orlandini acknowledges support by a grant within “Accordo di Programma Quadro CNR-Regione Lombardia: Progetto 3”.

References

1. Abdedaim, Y., Asarin, E., Gallien, M., Ingrand, F., Lesire, C., Sighireanu, M.: Planning Robust Temporal Plans: A Comparison Between CBTP and TGA Approaches. In: ICAPS 2007, pp. 2–10 (2007)
2. Bensalem, S., de Silva, L., Gallien, M., Ingrand, F., Yan, R.: “Rock Solid” Software: A Verifiable and Correct-by-Construction Controller for Rover and Spacecraft Functional Levels. In: i-SAIRAS 2010 (2010)
3. Cassez, F., David, A., Fleury, E., Larsen, K.G., Lime, D.: Efficient on-the-fly algorithms for the analysis of timed games. In: Abadi, M., de Alfaro, L. (eds.) CONCUR 2005. LNCS, vol. 3653, pp. 66–80. Springer, Heidelberg (2005)
4. Ceballos, A., Bensalem, S., Cesta, A., De Silva, L., Fratini, S., Ingrand, F., Ocon, J., Orlandini, A., Py, F., Rajan, K., Rasconi, R., Van Winnendael, M.: A goal-oriented autonomous controller for space exploration. In: ASTRA 2011 (2011)
5. Cesta, A., Finzi, A., Fratini, S., Orlandini, A., Tronci, E.: Analyzing Flexible Timeline-based Plans. In: ECAI 2010, vol. 215, pp. 471–476. IOS Press, Amsterdam (2010)
6. Cesta, A., Fratini, S.: The Timeline Representation Framework as a Planning and Scheduling Software Development Environment. In: PlanSIG 2008 (2008)
7. Fratini, S., Pecora, F., Cesta, A.: Unifying Planning and Scheduling as Timelines in a Component-Based Perspective. Archives of Control Sciences 18(2), 231–271 (2008)
8. Gat, E.: On three-layer architectures. In: Artificial Intelligence and Mobile Robots, pp. 195–210. MIT Press, Cambridge (1997)
9. Goldman, R.P., Musliner, D.J., Pelican, M.J.: Exploiting implicit representations in timed automaton verification for controller synthesis. In: Tomlin, C.J., Greenstreet, M.R. (eds.) HSCC 2002. LNCS, vol. 2289, pp. 225–238. Springer, Heidelberg (2002)
10. Lemai, S., Ingrand, F.: Interleaving Temporal Planning and Execution in Robotics Domains. In: AAAI 2004, pp. 617–622 (2004)
11. Maler, O., Pnueli, A., Sifakis, J.: On the Synthesis of Discrete Controllers for Timed Systems. In: Mayr, E.W., Puech, C. (eds.) STACS 1995. LNCS, vol. 900, pp. 229–242. Springer, Heidelberg (1995)
12. Milner, R.: Communication and concurrency. Prentice-Hall, Inc., Englewood Cliffs (1989)
13. Morris, P.H., Muscettola, N.: Temporal Dynamic Controllability Revisited. In: AAAI 2005, pp. 1193–1198 (2005)

14. Morris, P.H., Muscettola, N., Vidal, T.: Dynamic Control of Plans With Temporal Uncertainty. In: IJCAI 2001, pp. 494–502 (2001)
15. Muscettola, N.: HSTS: Integrating Planning and Scheduling. In: Zweben, M., Fox, M.S. (eds.) Intelligent Scheduling, pp. 169–212. Morgan Kauffman, San Francisco (1994)
16. Py, F., Rajan, K., McGann, C.: A Systematic Agent Framework for Situated Autonomous Systems. In: AAMAS 2010, pp. 583–590 (2010)
17. Shah, J., Williams, B.C.: Fast Dynamic Scheduling of Disjunctive Temporal Constraint Networks through Incremental Compilation. In: ICAPS 2008, pp. 322–329 (2008)
18. Vidal, T., Fargier, H.: Handling Contingency in Temporal Constraint Networks: From Consistency To Controllabilities. Journal of Experimental and Theoretical Artificial Intelligence 11(1), 23–45 (1999)

A Metric to Evaluate a Cluster by Eliminating Effect of Complement Cluster

Hamid Parvin, Behrouz Minaei, and Sajad Parvin

Islamic Azad University, Nourabad Mamasani Branch, Nourabad, Iran
`{parvin, b_minaei, s.parvin}@iust.ac.ir`

Abstract. In this paper a new criterion for clusters validation is proposed. This new cluster validation criterion is used to approximate the goodness of a cluster. A clustering ensemble framework based on the new metric is proposed. In the framework first a large number of clusters are prepared and then some of them are selected for final ensemble. The clusters which satisfy a threshold of the proposed metric are selected to participate in final clustering ensemble. For combining the chosen clusters, a co-association based consensus function is applied. Since the Evidence Accumulation Clustering (EAC) method cannot derive the co-association matrix from a subset of clusters, a new EAC based method which is called Extended EAC, EEAC, is applied for constructing the co-association matrix from the subset of clusters. Employing this new cluster validation criterion, the obtained ensemble is evaluated on some well-known and standard data sets. The empirical studies show promising results for the ensemble obtained using the proposed criterion comparing with the ensemble obtained using the standard clusters validation criterion.

Keywords: Clustering Ensemble, Stability Measure, Extended EAC, Cluster Evaluation, Selecting Scheme.

1 Introduction

Data clustering or unsupervised learning is an important and very difficult problem. The objective of clustering is to partition a set of unlabeled objects into homogeneous groups or clusters. Clustering has been considered a very challenging problem in Data Mining due to its lack of supervision. It is desired to partition data in such a way that the data points that belong to a cluster have maximum similarities while the data points that belong to different clusters have minimal similarities [6]. Clustering techniques require the definition of a similarity measure between patterns. Since there is no prior knowledge about cluster shapes, choosing a specific clustering method is not easy [17]. Because of the difficulty of the problem and the weaknesses of primary clustering, the researches' direction has turned to clustering ensemble. Cluster ensemble methods attempt to find a better and more robust clustering solution by fusing information from several primary data partitionings [11].

Fern and Lin [7] have suggested a clustering ensemble approach which selects a subset of solutions to form a smaller but better-performing cluster ensemble than

using all primary solutions. The ensemble selection method is designed based on quality and diversity, the two factors that have been shown to influence cluster ensemble performance. This method attempts to select a subset of primary partitions which simultaneously has both the highest quality and diversity. The Sum of Normalized Mutual Information, SNMI [8]-[10] and [18] is used to measure the quality of an individual partition with respect to other partitions. Also, the Normalized Mutual Information, NMI, is employed for measuring the diversity among partitions. Although the ensemble size in this method is relatively small, this method achieves significant performance improvement over full ensembles. Law et al. proposed a multi objective data clustering method based on the selection of individual clusters produced by several clustering algorithms through an optimization procedure [14]. This technique chooses the best set of objective functions for different parts of the feature space from the results of base clustering algorithms. Fred and Jain [10] have offered a new clustering ensemble method which learns the pairwise similarity between points in order to facilitate a proper partitioning of the data without the a priori knowledge of the number of clusters and of the shape of these clusters. This method which is based on cluster stability evaluates the primary clustering results instead of final clustering.

Moller and Radke [16] have introduced an approach to validate a clustering results based on partition stability. This method uses a perturbation which is produced by adding some noise to the data. An empirical study robustly indicates that the perturbation usually outperforms bootstrapping and subsampling. Whereas the empirical choice of the subsampling size is often difficult [5], the choosing of the perturbation strength is not so crucial. This method uses a Nearest Neighbor Resampling approach (NNR) that offers a solution to both problems of information loss and empirical control of the change degree made to the original data. The NNR techniques were first used for time series analysis [3]. Inokuchi et al. [12] have proposed a kernelized validity measures where a kernel means the kernel function used in support vector machines. Two measures are considered in this measure. One is the sum of the traces of the fuzzy covariances within clusters and the second is a kernelized Xie-Beni's measure [19]. This validity measure is applied to the determination of the number of clusters and also the evaluation of robustness of different partitionings. Das and Sil [4] have proposed a method to determine the number of clusters which validates the clusters using splitting and merging technique in order to obtain optimal set of clusters.

Alizadeh et al. discuss the drawbacks of the common approaches and then have proposed a new asymmetric criterion to assess the association between a cluster and a partition which is called Alizadeh-Parvin-Minaei criterion, APM. The APM criterion compensates the drawbacks of the common method. Also, a clustering ensemble method is proposed which is based on aggregating a subset of primary clusters. This method uses the Average APM as fitness measure to select a number of clusters. The clusters which satisfy a predefined threshold of the mentioned measure are selected to participate in the clustering ensemble. To combine the chosen clusters, a co-association based consensus function is employed [20].

To evaluate a cluster, the NMI method has many weaknesses that are described in [20]. Alizadeh et al. propose another version of NMI named max method. They also show that the max method also has some drawbacks, so they propose another metric named APMM, which is first of their author names [1].

This paper proposes a new measure to evaluate a cluster in that it is desired to evaluate the average similarity of the cluster with other clusters by eliminating its complement.

A large number real standard dataset from UCI repository [15] are used as benchmarks and it is shown that the proposed metric is very effective.

2 Proposed Method

In this section, first our proposed clustering ensemble method is briefly outlined, and then its phases are described in detail. The main idea of our proposed clustering ensemble framework is similar to Max and APMM (max; ie) to utilize a subset of the best performing primary clusters in the ensemble, rather than using all of clusters. Only the clusters which satisfy a stability criterion are better to participate in the consensus function. The cluster stability is defined according to NMI. Fig. 1 depicts the proposed clustering ensemble procedure.

As it is observed in the Fig. 1, the proposed framework has four steps. In the first step B partitionings are extracted out of dataset. The partitioning i is denoted by partitioning_i . The partitioning_i is obtained by a k-means algorithm with a new initialization of the seed points. Note that the partitioning_i is to extract $k(i)$ clusters out of dataset. Then each partitioning is broken in some distinct partitions (or clusters). It means partitioning_i converted to $k(i)$ clusters denoted by c_1^i, c_2^i, \dots and $c_{k(i)}^i$ respectively. After obtaining a pool of clusters, in the second step, a stability value is computed as a tag for each of them. The stability value of the cluster c_j^i is denoted by stab_j^i . The manner of computing stability for each cluster is described in the sections 2.2 in more detail. A subset of stable clusters having a good diversity is selected by a thresholding scheme in the third step. This step is explained in detail in section 2.3. In the next step, the selected clusters are used to construct the consensus partitioning. This is done in two subparts: (a) to extract a co-association matrix from them (section 2.4) along with (b) a linkage clustering. Since the original EAC method [8] cannot truly identify the pairwise similarities between dataitems when there is only a subset of clusters, we use a method explained in [1] to construct the co-association matrix from the base selected clusters. This method is called EEAC. The hierarchical single-link clustering is done along with the extraction of the co-association matrix extract the consensus clusters.

2.1 First Step: Producing of the Primary Clusters

In the first step B partitionings are extracted out of dataset by B independent runnings of the k-means algorithm. The partitioning_i is obtained by the i -th running of the k-means algorithm with a new initialization of the seed points. To produce the diverse cluster as much as possible the k-means algorithms are run, aiming at extracting different number of clusters out of dataset. It means that the partitioning_i extracts

$k(i)$ clusters out of dataset. As it is mentioned the proposed method tries to select a subset of well-performing clusters (or equivalently partitions) instead of a subset of clusterings (or equivalently partitionings). So each partitioning is broken in some distinct partitions clusters (or equivalently partitions).

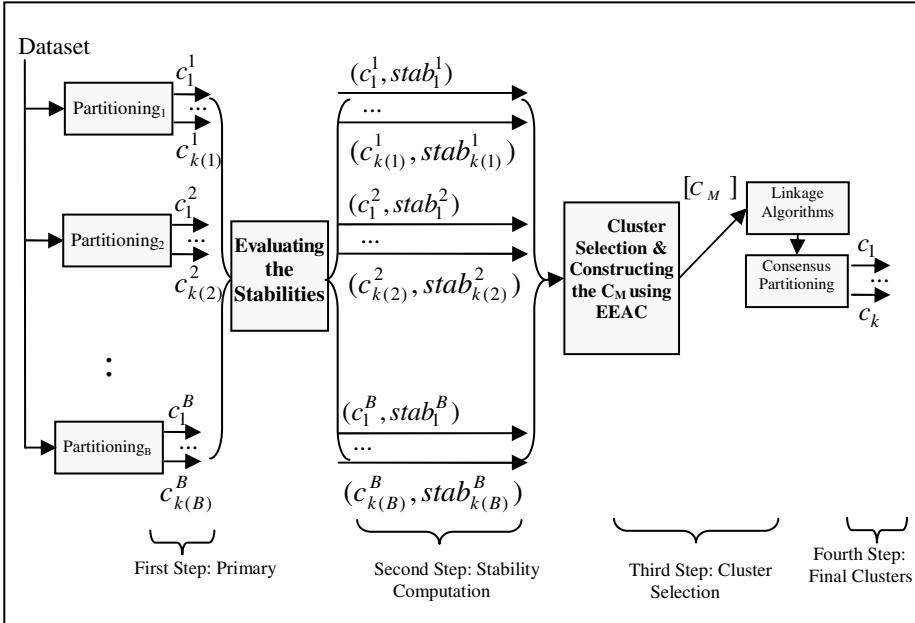


Fig. 1. The proposed Scheme

2.2 Second Step: Stability Computation

Since the goodness of a cluster C_i is determined by all of the data points, the goodness function $g_j = (C_i, D)$ depends on both the cluster C_i and the entire dataset D , instead of C_i alone. The stability as a measure of cluster goodness is used in [1], [13] and [20]. A stable cluster is the one that has a high likelihood of recurrence across multiple applications of a clustering algorithm. Stable clusters are usually preferable, since they are robust with respect to minor changes in the dataset [14].

Now assume that the stability of cluster C_i is to be computed. In this method first a set of partitionings over dataset is provided which is called the reference set. One can consider the partitionings obtained in the first step as reference set for decreasing the runtime. In this notation D is dataset and $P_w(D)$ is a partitioning over D . Now, the problem is: "How many times is the cluster C_i repeated in the reference partitions?" Assume that the NMI between the cluster C_i and a reference partition $P_w(D)$ is denoted by $NMI(C_i, P_w(D))$. While the most of previous works only compare *a partition with another partition* [18], however, the stability used in [14] evaluates the similarity between *a cluster and a partition* by transforming the cluster C_i to a partition and after that by employing the common partition-to-partition *NMI*. To

illustrate this method let $P_1 = P^a = \{C_i, D/C_i\}$ be a partition with two clusters, where D/C_i denotes the set of data points in D that are not in C_i . Then we may assume a second partition $P_2 = P^b = \{C_w^*, D/C_w^*\}$, where C_w^* denotes the union of all “positive” clusters in $P_w(D)$ and others are in D/C_w^* . A cluster C_r in $P_w(D)$ is positive cluster for C_i if more than half of its data points also belongs to C_i .

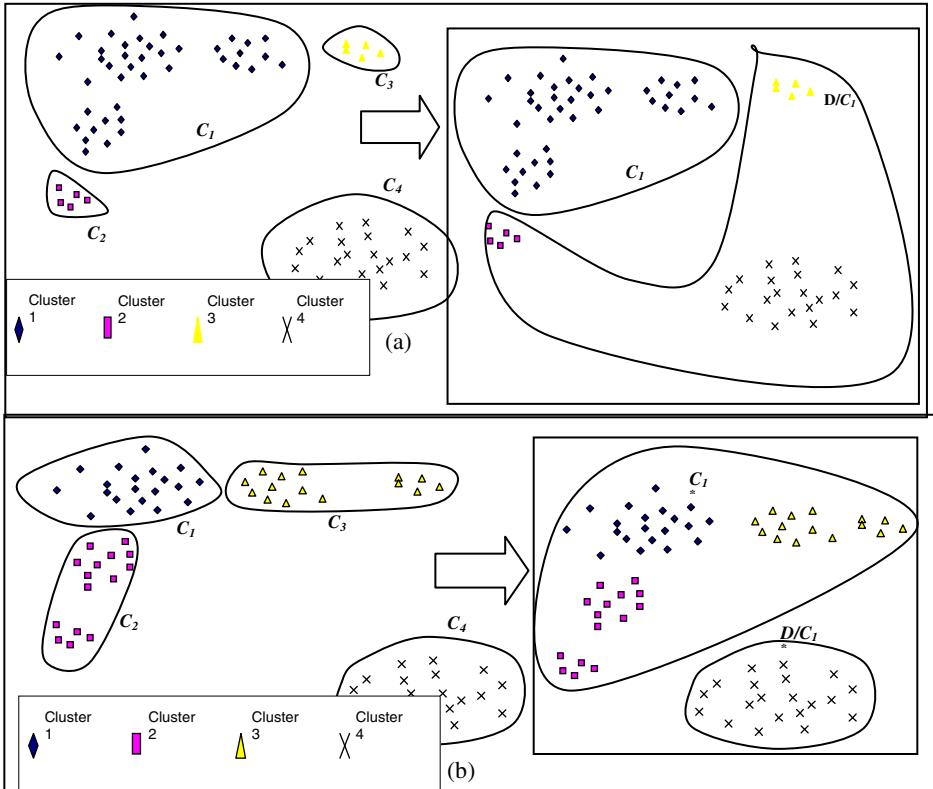


Fig. 2. Computing the stability of Cluster 1 of the partition in Fig. 2 (a) considering the partition in the Fig. 2 (b) of the reference set using NMI method

Now, define $NMI(C_i, P_w(D))$ by $NMI(P^a, P^b)$ which is calculated as [9]:

$$NMI(P^a, P^b) = \frac{-2 \sum_{i=1}^{k_a} \sum_{j=1}^{k_b} n_{ij}^{ab} \log \left(\frac{n_{ij}^{ab} \cdot n}{n_i^a n_j^b} \right)}{\sum_{i=1}^{k_a} n_i^a \log \left(\frac{n_i^a}{n} \right) + \sum_{i=1}^{k_b} n_i^b \log \left(\frac{n_i^b}{n} \right)} \quad (1)$$

where n is the total number of samples and n_{ij}^{ab} denotes the number of shared patterns between clusters $C_i^a \in P^a$ and $C_j^b \in P^b$; n_i^a is the number of patterns in the cluster i of partition a ; also n_j^b are the number of patterns in the cluster j of partition b .

This computation is done between the cluster C_i and all partitions available in the reference set. This method is named NMI method. Fig. 2 illustrates the NMI method.

After producing P_1 , if we assume a second partition $P_2 = P^b = \{C_w^*\} \cup Cs_w^*$, where C_w^* denotes the same clusters in $P_w(D)$ defined by APM [1] and for each of other data we consider a cluster. The set of these clusters is denoted by Cs_w^* . Fig. 3 shows the method explained above which is named Edited APM, EAPM.

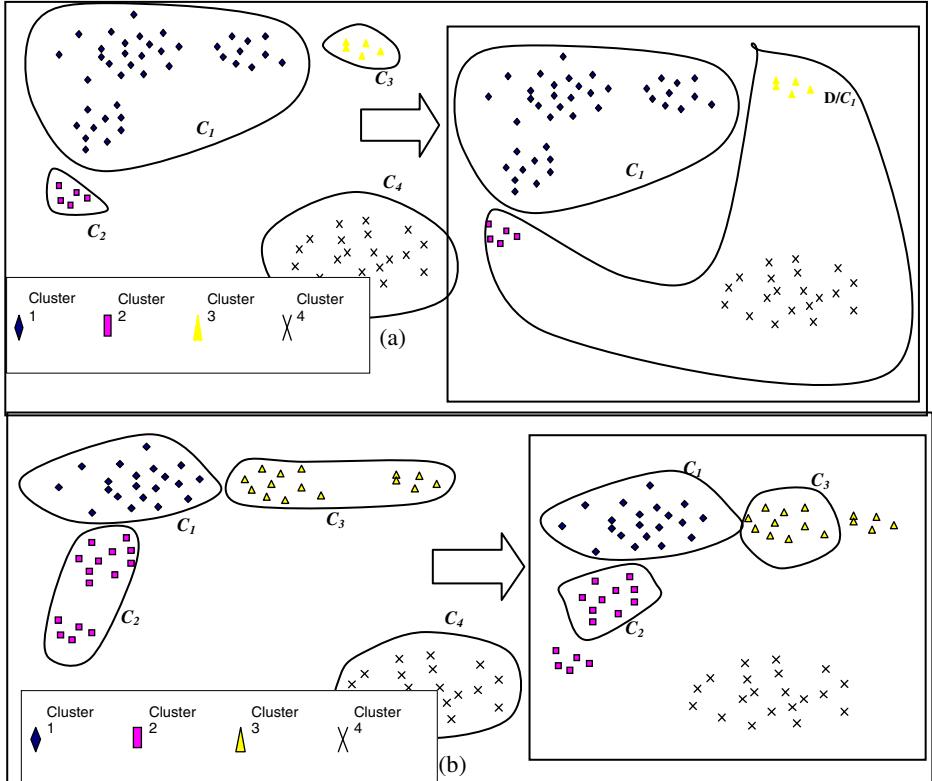


Fig. 3. Computing the stability of Cluster 1 of the partition in Fig. 3 (a) considering the partition in the Fig. 3 (b) of the reference set using EAPM method

NMI_h in Fig. 4 shows the stability of cluster C_i with respect to the h th partition in reference set. The total stability of cluster C_i is defined as:

$$Stab(C_i) = \frac{\sum_{j=1}^B NMI_j}{B} \quad (2)$$

This procedure is applied for each cluster available in the pool clusters obtained in the first step. It means this procedure must be iterated q times, where q is computed as equation 3.

$$q = \sum_{i=1}^B k(i) \quad (3)$$

2.3 Third Step: Stability-Based Selection

This step is simply done by a thresholding. It means that the clusters with higher stability values are selected for next step and others are omitted.

2.4 Forth Step: Consensus Function and Obtaining Final Partition

In this step, the selected clusters are used to produce final clusters in a co-association based model. In the step it is to construct the co-association matrix and then to apply a hierarchical clustering. To construct the co-association matrix from the selected clusters EEAC is employed. In the EAC method the m primary partitions from dataset are accumulated in a $n \times n$ co-association matrix. Each entry in this matrix is computed from equation 4.

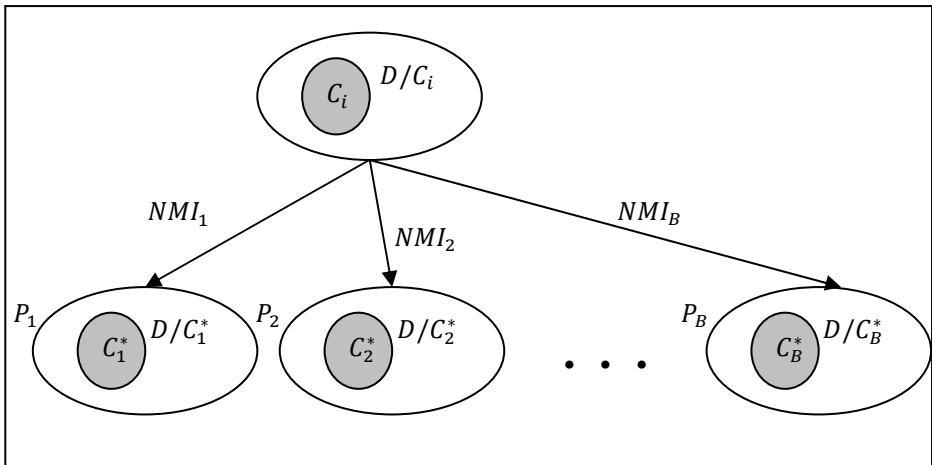


Fig. 4. Computing the Stability of Cluster C_i considering a reference set

$$C_{ij} = \frac{n_{ij}}{m_{ij}} \quad (4)$$

where m_{ij} counts the number of clusters shared by objects with indices i and j in the pool of all clusters obtained in the first step. It is worthy to note that the maximum possible value of m_{ij} computed as equation 3. Also n_{ij} is the number of partitions where this pair of objects is simultaneously present in the selected clusters. Note that the value of n_{ij} is at most as many as the number of selected clusters which is less than the value of m_{ij} .

3 Experimental Results

This section reports and discusses the empirical studies. The proposed method is examined over 5 different standard datasets. It is tried for datasets to be diverse in their number of true classes, features and samples. A large variety in used datasets can more validate the obtained results. Brief information about the used datasets is available in [15].

All experiments are done over the normalized features. It means each feature is normalized with mean of 0 and variance of 1, $N(0, 1)$. All of them are reported over means of 10 independent runs of algorithm. The final performance of the clustering algorithms is evaluated by re-labeling between obtained clusters and the ground truth labels and then counting the percentage of the true classified samples. Table 1 shows the performance of the proposed method comparing with most common base and ensemble methods.

Table 1. Experimental results

| Metric Evaluation | Dataset | | | | | | | | | | |
|-------------------|------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | N. Breast Cancer | Iris | N. Bupa | N. SAHeart | Ionosphere | N. Glass | Halfring | N. Galaxy | N. Yeast | Wine | N. Wine |
| NMI | 95.73 | 76.13 | 54.33 | 63.36 | 70.60 | 47.76 | 74.48 | 31.27 | 42.93 | 69.38 | 85.17 |
| MAX | 96.49 | 84.87 | 57.42 | 63.87 | 57.75 | 44.35 | 74.55 | 29.85 | 51.27 | 70.00 | 94.44 |
| APM | 95.46 | 90.00 | 55.07 | 63.85 | 70.66 | 45.79 | 54.00 | 30.65 | 53.10 | 70.23 | 96.63 |
| EAPM | 96.93 | 88.67 | 54.78 | 63.20 | 71.23 | 43.93 | 88.00 | 30.65 | 50.47 | 70.23 | 97.19 |

The results show that although each of the metrics can obtain a good result over a specific dataset, it does not perform well over other datasets. For example, according to Table 1 the ensemble based on NMI obtains a good clustering result over Glass dataset. But, it has lower performance in comparison to results of ensemble based on other metrics in the case of Bupa dataset. The results of the ensemble methods are the results of an ensemble of 100 K-means which are fused by EAC method. The 90% sampling from dataset is used for creating diversity in primary results. The sub-sampling (without replacement) is used as the sampling method. Also the random initialization of the seed points of K-means algorithm helps them to be more diverse. The single linkage algorithm is applied as consensus function for deriving the final clusters from co-association matrix. The top 33% stable clusters are employed in constructing co-association matrix.

4 Conclusion and Future Works

In this paper a new clustering ensemble method is proposed which is based on a subset of total primary spurious clusters. Since the quality of the primary clusters are not equal and presence of some of them can even yield to lower performance, here a method to select a subset of more effective clusters is proposed. A common cluster validity criterion which is needed to derive this subset is based on normalized mutual information. In this paper some drawbacks of this criterion is discussed and an alternative criterion is suggested which is named EAPM. The experiments show that the EAPM criterion does slightly better than NMI criterion generally; however it significantly outperforms the NMI criterion in the case of synthetic data sets. Because of the symmetry which is concealed in NMI criterion and also in NMI based stability, it yields to lower performance whenever symmetry is also appeared in the data set. Another innovation of this paper is a method for constructing the co-association matrix where some of clusters and respectively some of samples do not exist in partitions. This new method is called Extended Evidence Accumulation Clustering, EEAC. The empirical studies over several data sets robustly show that the quality of the proposed method is usually better than other ones.

References

1. Alizadeh, H., Minaei-Bidgoli, B., Parvin, H.: An Asymmetric Criterion for Cluster Validation. In: 16th Iberoamerican Congress on Pattern Recognition (CIARP 2011). LNCS. Springer, Heidelberg (in press, 2011) ISSN: 0302-9743
2. Ayad, H., Kamel, M.S.: Cumulative Voting Consensus Method for Partitions with a Variable Number of Clusters. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 30(1), 160–173 (2008)
3. Brandsma, T., Buishand, T.A.: Simulation of extreme precipitation in the Rhine basin by nearest-neighbour resampling. *Hydrology and Earth System Sciences* 2, 195–209 (1998)
4. Das, A.K., Sil, J.: Cluster Validation using Splitting and Merging Technique. In: Int. Conf. on Computational Intelligence and Multimedia Applications, ICCIMA (2007)
5. Davison, A.C., Hinkley, D.V., Young, G.A.: Recent developments in bootstrap methodology. *Statistical Science* 18, 141–157 (2003)
6. Faceli, K., Marcilio, C.P., Souto, D.: Multi-objective Clustering Ensemble. In: Proceedings of the Sixth International Conference on Hybrid Intelligent Systems, HIS 2006 (2006)
7. Fern, X.Z., Lin, W.: Cluster Ensemble Selection. In: SIAM International Conference on Data Mining, SDM 2008 (2008)
8. Fred, A., Jain, A.K.: Data Clustering Using Evidence Accumulation. In: Proc. of the 16th Intl. Conf. on Pattern Recognition, ICPR 2002, Quebec City, pp. 276–280 (2002)
9. Fred, A., Jain, A.K.: Combining Multiple Clusterings Using Evidence Accumulation. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 27(6), 835–850 (2005)
10. Fred, A., Jain, A.K.: Learning Pairwise Similarity for Data Clustering. In: Proc. of the 18th Int. Conf. on Pattern Recognition, ICPR 2006 (2006)
11. Fred, A., Lourenco, A.: Cluster Ensemble Methods: from Single Clusterings to Combined Solutions. *SCI*, vol. 126, pp. 3–30 (2008)
12. Inokuchi, R., Nakamura, T., Miyamoto, S.: Kernelized Cluster Validity Measures and Application to Evaluation of Different Clustering Algorithms. In: IEEE Int. Conf. on Fuzzy Systems, Canada (July 16–21, 2006)
13. Lange, T., Braun, M.L., Roth, V., Buhmann, J.M.: Stability-based model selection. In: Advances in Neural Information Processing Systems, vol. 15. MIT Press, Cambridge (2003)
14. Law, M.H.C., Topchy, A.P., Jain, A.K.: Multiobjective data clustering. In: Proc. of IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 424–430 (2004)
15. Newman, C.B.D.J., Hettich, S., Merz, C.: UCI repository of machine learning databases (1998), <http://www.ics.uci.edu/~mlearn/MLSummary.html>
16. Möller, U., Radke, D.: Performance of data resampling methods based on clustering. *Intelligent Data Analysis* 10(2) (2006)
17. Roth, V., Lange, T., Braun, M., Buhmann, J.: A Resampling Approach to Cluster Validation. In: Intl. Conf. on Computational Statistics, COMPSTAT (2002)
18. Strehl, A., Ghosh, J.: Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research* 3, 583–617 (2002)
19. Xie, X.L., Beni, G.: A Validity measure for Fuzzy Clustering. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 13(4), 841–846 (1991)
20. Alizadeh, H., Minaei-Bidgoli, B., Parvin, H.: A New Criterion for Clusters Validation. In: Artificial Intelligence Applications and Innovations (AIAI 2011). LNCS. Springer, Heidelberg (in press 2011) ISSN: 0302-9743

Predicting Numbers: An AI Approach to Solving Number Series

Marco Ragni and Andreas Klein

Center for Cognitive Science,
Friedrichstr. 50, 79098 Freiburg, Germany
`{ragni,klein}@cognition.uni-freiburg.de`

Abstract. Solving number series poses a challenging problem for humans and Artificial Intelligence Systems. The task is to correctly predict the next number in a given series, in accordance with a pattern inherent to that series. We propose a novel method based on Artificial Neural Networks with a dynamic learning approach to solve number series problems. Our method is evaluated on an own experiment and over 50.000 number series from the Online Encyclopedia of Integer Sequences (OEIS) database.

1 Introduction

Identifying the pattern in a number series can pose serious problems for human and artificial reasoning systems. Take, for example:

- (i) 5, 7, 9, 11, 13, ...
- (ii) 2, 5, 9, 19, 37, 75, 149, 299, ...

The first problem is easy to solve and requires only the addition of 2, while the second problem requires the application of the function $a_n := 2a_{n-1} + (-1)^n$ on $a_1 := 2$ and is, as we will see, difficult for humans and for ANNs. Number series problems are interesting for Artificial Intelligence as principally any computable function can be hidden and the set of operators is not necessarily restricted to $+, -, \times$. Number series problems are used in intelligence tests for determining mathematical pattern-recognition capabilities. Especially the latter aspect identifying patterns is an important aspect of any intelligent system, and can provide a fruitful benchmark for general approaches. There is an Online Encyclopedia of Integer Sequences [19] (OEIS) and a journal dedicated to the study of integer sequences [2].

In the following, we propose a method using artificial neural networks (ANNs) and a novel dynamic approach to solve number series problems. We compare our approach with human performance in solving number series problems and use the OEIS database for benchmarking. Finally, we discuss the results and performance and provide a short outlook of our further work.

Table 1. Pattern generation example for an ANN using the dynamic approach with three input nodes and a number series of seven given numbers (n_1, \dots, n_7) with the n_8 -th to predict. Four patterns (p_1, \dots, p_4) are used for training, each with three training values (v_i, v_{i+1}, v_{i+2}) and one target value $t = n_{i+3}$. One pattern (p_5) is used for testing the ANN without a target value.

| Pattern | n_1 | n_2 | n_3 | n_4 | n_5 | n_6 | n_7 | n_8 |
|---------|-------|-------|-------|-------|-------|-------|-------|-------|
| p_1 | | v_1 | v_2 | v_3 | t | | | |
| p_2 | | | v_2 | v_3 | v_4 | t | | |
| p_3 | | | | v_3 | v_4 | v_5 | t | |
| p_4 | | | | | v_4 | v_5 | v_6 | t |
| p_5 | | | | | v_5 | v_6 | v_7 | ? |

2 Using a Dynamic Learning Approach

For our attempt to solve number series with artificial neural networks [3], we use three-layered networks with error back-propagation and hyperbolic tangent as activation. We used $f_i = \frac{n}{10^{len(n)}}$ as input function for the network and the inverse, $f_o = f_i^{-1}$, as output function of the network. The weights were initially randomly assigned with values between zero and one. A momentum factor of 0.1 was used.

For data analysis, we systematically varied the learning rate, the number of input nodes, the number of hidden nodes and the number of training iterations. These variations should allow for a comparison of the different ANNs with empirical results. For all configurations only one output node was used.

To generate patterns for training and testing a network, we built patterns as tuples of training values and one target value. The number of training values of a pattern is equivalent to the number of input nodes m of the network used. Starting with the first number, a subsequence of training values was shifted through the number series. As corresponding target value, the next number of the subsequence was used (cf. Table II).

Since the last given value of the number series with length n remains as target value and we need at least one training and one test pattern, the maximum length of a subsequence of training values for a pattern is $n - 2$. Hence, for a network configuration with m input nodes exactly $n - m$ patterns were generated. Consequently, the first $(n - m) - 1$ patterns were used for training, while the last one remained for testing and thus predicting the last given number of the sequence. Therefore, the last given number of the number series - the main target value - was never used for training.

The training of the network was iterated on the patterns for a various number of times. After training the network we tested the learning result with the last pattern without a given target value and compared the prediction with the actual value.

Table 2. Results of our empirical analysis with 17 participants for 20 number series and the performance of 840 network configurations with four variants of training iterations. C indicates the number of correct answers, I the number of incorrect ones and U the number of participants who were unable to solve the number series.

| Number Series | C. | I. | U. | Number of solving configurations with iterations: 0.5k; 1k; 5k; 10k | Number Series | C. | I. | U. | Number of solving configurations with iterations: 0.5k; 1k; 5k; 10k |
|-----------------|----|----|--------------------|---|----------------|----|----|-----------------|---|
| 12,15,8,11,4 | 15 | 2 | 306; 385; 475; 530 | 4,11,15,26,41 | 8 | 1 | 8 | 6; 14; 32; 22 | |
| 148,84,52,36,28 | 12 | 2 | 555; 637; 670; 689 | 5,6,7,8,10 | 10 | 1 | 6 | 83; 91; 65; 114 | |
| 2,12,21,29,36 | 14 | 1 | 2 | 405; 440; 502; 539 | 54,48,42,36,30 | 16 | 1 | | 274; 299; 338; 376 |
| 2,3,5,9,17 | 13 | 1 | 3 | 3; 7; 61; 192 | 6,8,5,7,4 | 16 | 1 | | 134; 169; 198; 219 |
| 2,5,8,11,14 | 9 | 3 | 5 | 581; 618; 659; 667 | 6,9,18,21,42 | 14 | 1 | 2 | 48; 24; 94; 101 |
| 2,5,9,19,37 | 6 | 4 | 7 | 0; 0; 0; 0 | 7,10,9,12,11 | 14 | 3 | | 111; 202; 380; 404 |
| 25,22,19,16,13 | 16 | 1 | | 562; 615; 648; 667 | 8,10,14,18,26 | 13 | 1 | 3 | 57; 46; 30; 29 |
| 28,33,31,36,34 | 17 | | | 121; 183; 315; 332 | 8,12,10,16,12 | 17 | | | 37; 75; 41; 51 |
| 3,6,12,24,48 | 13 | 1 | 3 | 0; 0; 0; 0 | 8,12,16,20,24 | 15 | 2 | | 507; 546; 594; 613 |
| 3,7,15,31,63 | 12 | 3 | 2 | 0; 0; 0; 0 | 9,20,6,17,3 | 16 | 1 | | 255; 305; 397; 406 |

2.1 Empirical Results

Are humans better at solving number series problems than artificial networks systems? To have reliable data we conducted an experiment with 20 number series and 17 participants to evaluate reasoning difficulty and to benchmark the results of our ANNs.

For the analysis of the empirical data, we varied the learning rate within 0.125 and 0.875, with a step-width of 0.125, the number of input nodes (iterated from one to six), the number of hidden nodes (iterated from one to twenty), and raised the number of training iterations in three steps from 500 over 1000 and 5000 up to 10000 iterations.

There are three number series which were not solved by any ANN configuration. All others could be solved (cf. Table 2). Comparisons between the different configurations show different results for the number of iterations and between the number series - reflecting the difficulties of the configurations to learn the concept of the number series. Furthermore, worse results for ANNs with six or seven input nodes, and an increasing learn rate (cf. Fig. 1) .

2.2 Testing the Online Encyclopedia of Integer Sequences

How good is our method in general? Unfortunately, no benchmark is available. To investigate this question we used the OEIS database. Of the 187.440 number series in the OEIS, we selected those series which consist of at least 20 numbers with values ± 1.000 . In total 57.524 number series were used to benchmark our ANN approach with dynamic learning.

For our analysis we varied the learning rate within 0.125 and 0.875, with a step-width of 0.250, and the number of hidden nodes (iterated from one to

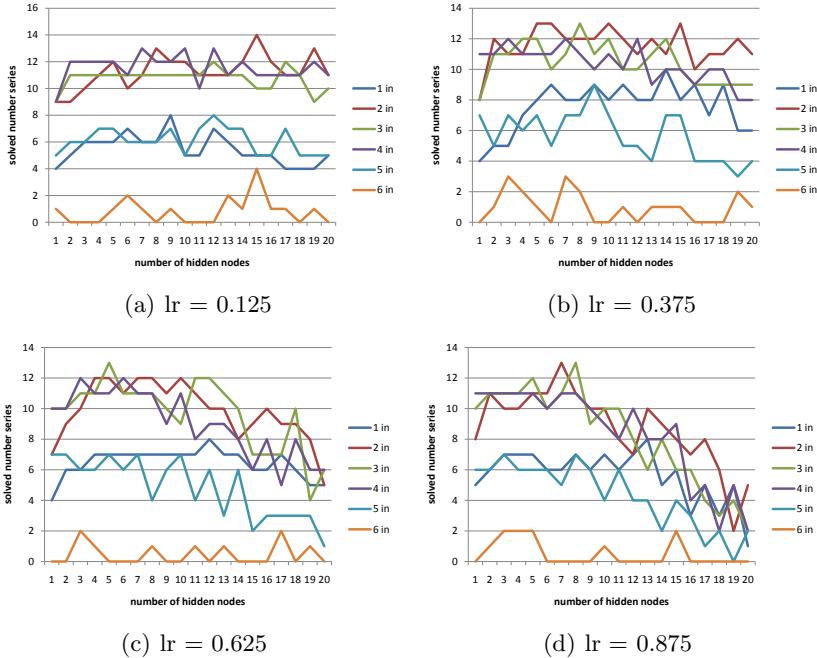


Fig. 1. Results for ANN configurations (lr = learning rate, in = input nodes) with 10.000 training iterations applied to the 20 problems depicted in Table 2

seven). We used four input nodes and 1000 training iterations. The best configuration (input nodes = 4, hidden nodes = 2, learning factor = .75) was able to solve 12.764 number series problems and all 49 tested configurations together were able to solve 26.951 of the number series. Again, solving means, they were able to correctly predict the last number of the series, even though they had never trained it. The number series in the OEIS can have many construction principles (e.g., A001003 Schroeder's second problem, i.e., the number of ways to insert parentheses in a string of n symbols), which do not depend on "simple" generation patterns.

3 Discussion

We systematically varied input nodes, hidden nodes, and the learning rate to compare the different artificial neural networks structures. Using a dynamic learning approach, we were able to predict correctly 17 of 20 times in one experiment and in another 26.951 of 57.524 number series. One first conclusion we can draw is that the structure of the Artificial Neural Networks can determine the success of solving a number sequence – there is a systematic pattern between learning rate, input, and output nodes – showing that 2-4 input nodes and about 5-6 hidden nodes provide the best framework to solve number series.

Table 3. Some examples of the automatically generated problems reported in [8]

| OEIS Number | Solved by Example Configuration |
|-------------|---|
| A009087 | 10/49 0.25, 0.75, 0.875: 4-4-1 |
| A023194 | 0 (not in our benchmark set) |
| A033950 | 0 |
| A036433 | 22/49 0.25, 0.375, 0.625, 0.75: 4-4-1 |
| A036438 | 22/49 0.25, 0.5, 0.875: 4-1-1 |
| A046952 | 0 (not in our benchmark set) |

Some of the sequences in the OEIS database [1] were automatically generated [8] by an extended version of the HR tool [7]. Although this seems to be in some sense inverse to our approach (generating instead of solving number series) the underlying problem is identical, as there must be a valid principle of how these problems are generated. In Table 3 we briefly report of how our approach deals with some of the automatically invented sequences reported in [8]. Half of the automatically generated problems could be solved by our method. Two of them increase too fast and therefore are not in our benchmark set. This is a property which is more difficult to approximate. Related to our approach is the SeekWhence tool [6,4] to model a cognitive plausible human performance on finding definitions of sequences. This model is principally symbolic but can – depending on the *temperature* – show a more randomized behavior. The Guess (or Rate tool) [5] provides a closed definition of a given sequence. This approach uses mathematical interpolation to solve number series. Future work shall systematically investigate the architectural properties of these networks, i.e., to provide a theory which number series characteristics is best solved by which ANN.

References

1. The On-Line Encyclopedia of Integer Sequences, <http://oeis.org/>
2. Journal of Integer Sequences, <http://www.cs.uwaterloo.ca/journals/JIS/>
3. McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics 5, 115–133 (1943)
4. Meredith, M.J.E.: Seek-Whence: A Model of Pattern Perception. Ph.D. Dissertation, Department of computer Science, Indiana University Indianapolis, IN, USA (1986)
5. Krattenthaler, C.: Advanced determinant calculus. Technical report, Institute of Mathematics, University of Vienna (1991)
6. Hofstadter, D.: Fluid Concepts and Creative Analogies. Basic Books, New York (1995)
7. Colton, S., Bundy, A., Walsh, T.: Automatic concept formation in pure mathematics. In: Dean, T. (ed.) Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, pp. 786–791 (1999)
8. Colton, S., Bundy, A., Walsh, T.: Automatic Invention of Integer Sequences. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence AAAI, 558–563 (2000)
9. Sloane, N.J.A.: The On-Line Encyclopedia of Integer Sequences. Notices of the American Mathematical Society 50(8), 912–915 (2003)

Prediction of Classifier Training Time Including Parameter Optimization

Matthias Reif, Faisal Shafait, and Andreas Dengel

German Research Center for Artificial Intelligence, Trippstadter Str. 122,

67663 Kaiserslautern, Germany

`{matthias.reif,faisal.shafait,andreas.dengel}@dfki.de`

Abstract. Besides the classification performance, the training time is a second important factor that affects the suitability of a classification algorithm regarding an unknown dataset. An algorithm with a slightly lower accuracy is maybe preferred if its training time is significantly lower. Additionally, an estimation of the required training time of a pattern recognition task is very useful if the result has to be available in a certain amount of time.

Meta-learning is often used to predict the suitability or performance of classifiers using different learning schemes and features. Especially landmarking features have been used very successfully in the past. The accuracy of simple learners are used to predict the performance of a more sophisticated algorithm.

In this work, we investigate the quantitative prediction of the training time for several target classifiers. Different sets of meta-features are evaluated according to their suitability of predicting actual run-times of a parameter optimization by a grid search. Additionally, we adapted the concept of landmarking to time prediction. Instead of their accuracy, the run-time of simple learners are used as feature values.

We evaluated the approach on real world datasets from the UCI machine learning repository and StatLib. The run-time of five different classification algorithms are predicted and evaluated using two different performance measures. The promising results show that the approach is able to reasonably predict the training time including a parameter optimization. Furthermore, different sets of meta-features seem to be necessary for different target algorithms in order to achieve the highest prediction performances.

1 Introduction

Selecting the best classifier for a given problem instance is an important part of developing a pattern recognition system. The choice can be made by different points of views. Typically, the achieved classification performance is the most important aspect. However, the No-Free-Lunch theorem [19] tells us that there is no uniformly best algorithm.

Furthermore, the computation time is often a significant factor when choosing an algorithm, too. If the expected performance values of several algorithms are

the same, the algorithm with a lower run-time is usually preferred. Also slight decreases in performance may be acceptable if the reduction in run-time is significant. Sometimes, only a limited amount of time is available for computing the results. Furthermore, if the user has to pay for the computation time, he might not want to start a possibly time-consuming process without any idea about its duration. All these considerations make the estimation about the expected time needed a valuable information, especially for the increasing amount of large datasets.

Usually, an algorithm is described by a general statement about its complexity. For example, Multilayer Perceptrons (MLP) are known to have a rather high training time compared to a k -Nearest Neighbor approach. Nevertheless, the actual run-time often depends on the dataset and the exact parameter values of the algorithm. Furthermore, categorical time estimations like “high” or “low” do not give the user the same amount of information like actual time values using real units. For example, a “high” run-time could mean “several hours” to “several weeks” or even longer. Such nominal values are only limitedly useful for comparisons of multiple classifiers. In contrast to this, the prediction of real numbers can be much more precise. Additionally, values of actual time units make the estimation much more useful for the user. The theoretical computational complexity is also known for many algorithms. Since constant terms are neglected in computational complexity theory, the practical usefulness of such indications is limited, too.

However, a challenge in predicting run-times is the dependency of the hardware the algorithm runs on. A faster computer will always decrease the run-time for the same dataset and the same algorithm. This makes the estimation of real time values harder if the prediction model was trained on time data that was gathered on a different machine. The goal is to predict the time by taking the users machine into account as well.

In this paper, a method for predicting actual run-times of classification algorithms is presented. Since most algorithms contain parameters that influence their performance, they are typically optimized. Therefore, we do not predict the time of one training or one application of a classifier but the time needed for a grid search of the most important parameters. This includes multiple training and application phases.

The presented approach uses concepts of meta-learning including features of datasets. A regression model is learned that is able to predict the run-time of a particular algorithm regarding an unknown dataset. We investigated traditional features of datasets as well as features that are more specialized for predicting time values. Therefore, we adapted the successfully used concept of landmarking to the time domain. This new type of features enables the presented approach to take the performance of the users machine into account. The learned model becomes more independent of the actual computing power and is able to predict the run-time more precisely.

The rest of the paper is structured as follows. In the next section, we describe the approach of meta-learning including related work. The challenges of

predicting the run-time of a grid search are presented in Section 3. Section 4 contains the detailed description of the presented approach. The evaluation and its results are given in Section 5. The last section comprises of the conclusion.

2 Meta-Learning

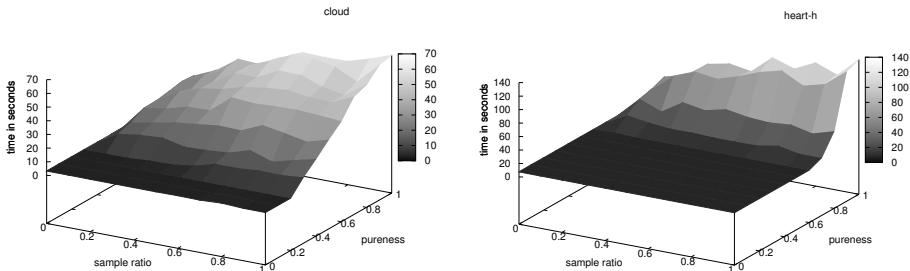
Meta-learning uses knowledge about already solved problem instances to gain information about unknown problems regarding one or multiple learning schemes. Typically, meta-learning is based on features of datasets. These features are often called meta-features. They describe properties of a dataset by using different approaches. Simple meta-features use directly accessible properties like the number of samples, the number of attributes or the number of classes. More sophisticated features are statistical measures [8,17], which are based on statistical analysis of the data. Typical measures of this group are the kurtosis and the skewness. The group of information theoretic features basically use the entropy values of the attributes and the class label [16].

Two more recently proposed groups are landmarking and model-based meta-features. Both approaches utilize other classification algorithms. Landmarking [15,4,2,9] applies simple and fast computable algorithms on the dataset and uses the achieved classification performance as feature value. The model-based features [14,3] also create a classification model of the data, but use several properties of this model instead of its performance. Model-based features are e.g. the width or height of an unpruned decision tree that was build on the dataset.

A typical application of meta-learning is the prediction of the best classifier for an unknown dataset. The used knowledge consists of the meta-features of known datasets and the information about what algorithm worked best for each of the datasets. Based on such a meta-dataset, a learning scheme creates a classification model that can be used for predicting the best classifier of an unknown dataset.

A slightly different approach creates a ranking of all considered target algorithms. The correct ranking is known for several datasets and the predicted ranking of the new dataset is typically gained by a nearest-neighbor approach. The distance measure is based on the meta-features. Brazdil et al. [6] presented a method for ranking algorithms that also includes computation times. A multi-criteria measure involves a ratio of accuracies and a ratio of times between two considered algorithms. The run-time is not directly predicted but only influences the ranking score. The strength of the influence is a user defined parameter.

Regression was also used for meta-learning [10,11,15]. Instead of predicting the best algorithm using a classification approach, this method predicts quantitative performance values. Therefore, for each target classifier whose performance should be predicted, a separate regression model has to be trained. Again, the features are the meta-features of the datasets, but the label is the actual performance value of one single target classifier. Various meta-features and regression algorithms have been used to predict different performance measures of classification algorithms, but none of these methods predicted their run-time.



(a) Different parameter combinations require different run-times.
 (b) Additionally, the distribution of the run-time differs for other datasets.

Fig. 1. The parameters *sample ratio* and *pureness* of the Ripper classifier against the required run-time for the two datasets *cloud* and *heart-h*

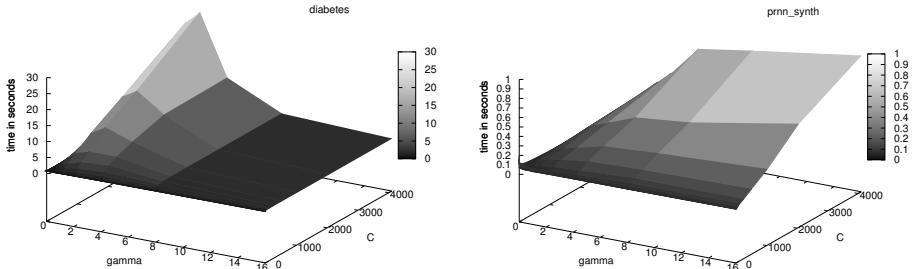
Lindner and Studer [12] used a case-based reasoning approach for algorithm recommendation by meta-learning that also considers time aspects. Training and testing time are treated separately. Each algorithm is generally assigned to one of five categories from “very fast” to “very slow” in order to include requirements of the user into the recommendation. Different run-times of the algorithms for different datasets were not investigated.

In this paper, a regression approach for predicting the time of a grid search for diverse classifiers is presented. The expected run-time is predicted in a quantitative way instead of predicting categories or comparisons between algorithms only. The user gets an actual estimation of the time required for optimizing one particular algorithm on his data. Additionally, the prediction can be used to choose the most suitable algorithm for a problem respecting run-time conditions as well.

3 Run-Time of a Grid Search

Since the performance of most classifiers depends on parameter values, the parameters are usually optimized. A simple and often used method for parameter optimization is a grid search. All predefined combinations of parameter values are evaluated to determine the best of them. The advantage of a grid search is that it usually delivers very good results. However, the drawback of this brute force approach is the rather high run-time compared to other optimization strategies.

For the time prediction of a grid search, one may think of predicting the run-time for one single evaluation of the classifier using one defined parameter combination and afterwards multiply the result with the size of the grid. However, as visible for the *cloud* dataset in Figure 1(a), different parameter combinations require different amounts of time. The plot shows the run-time of training the Ripper classifier for different combinations of its two parameters *sample ratio* and *pureness*. If, for example, the time of the lowest values of the parameters have been used, the total run-time will be significantly underestimated. Also



(a) A high value of γ lead to a shorter run-time for the *diabetes* dataset

(b) In contrast to the *diabetes* dataset, a higher value of γ results in a longer run-time for the *prnn_synth* dataset

Fig. 2. Run-time of training a Support Vector Machine (SVM) with different combinations of its parameters γ and C on the *diabetes* dataset and the *prnn_synth* dataset

other parameters of widely used classifiers obviously influence their run-time, e.g. the maximal depth of a decision tree or the learning rate of a Multilayer Perceptron (MLP).

A solution might be using the estimation for a defined parameter combination and a constant factor in addition to the grid size in order to compensate these dependencies. However, the distributions of the run-times within the parameter space differs for different datasets as well. For example, the increase of run-time for the *heart-h* dataset is significantly different than for the *cloud* dataset, as visible in Figure 1(b).

Furthermore, the measured run-times can differ even more between multiple datasets. Figure 2 shows the run-time for the typically optimized parameters γ and C of a Support Vector Machine (SVM). As visible in Figure 2(a), a higher value of the kernel parameter γ leads to a shorter run-time for the *diabetes* dataset, whereas for the *prnn_synth* dataset in Figure 2(b), a higher value of γ results in a longer run-time. Obviously, the distribution of the run-time depends on the dataset itself. Therefore, meta-features, which describe the data, are additionally necessary for a precise prediction of the run-time.

4 Methodology

For each target classifier, whose run-time should be predicted, a separate regression model is trained. The training data for the learning scheme consists of the knowledge about known datasets. Each instance of the training set describes one dataset. It contains the meta-features of the dataset and the measured run-time of the considered target classifier. The run-time is used as the target variable. After the learning phase, the resulting model can be used to predict the run-time of an unknown dataset. The model is applied on the meta-features of this dataset. The overall approach is illustrated in Figure 3.

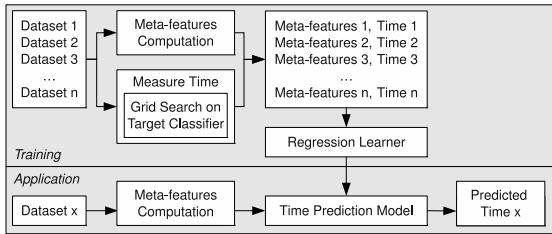


Fig. 3. Training of a time prediction model for one target classifier using a regression learner (top) and application of the model to an unknown dataset (bottom). The target value is the run-time of a parameter optimization.

4.1 Traditional Meta-Features

As a first set of meta-features, we used typical measures from the previously mentioned groups. This set includes the following 34 meta-features:

Simple meta-features: number of samples, number of classes, number of attributes, number of nominal attributes, number of numerical attributes, dimensionality (number of attributes divided by number of samples)

Statistical meta-features: kurtosis, skewness, canonical discriminant correlation, first normalized eigenvalues of canonical discriminant matrix, absolute correlation

Information theoretic meta-features: normalized class entropy, normalized attribute entropy, joint entropy, mutual information, noise-signal-ratio, equivalent number of attributes

Model-based meta-features: For these features, a decision tree is trained without pruning. Different properties of this tree are used as feature values: number of leaves, number of nodes, nodes per attribute, nodes per sample, leaf corroboration.

Additionally the minimum, maximum, mean value and standard deviation of the following measures are used: length of a branch, number of nodes in a level, number of occurrences of attributes in a split.

The previously successfully used concept of landmarking uses only the performance of several classification algorithms. Since these values are obviously unrelated to the run-time of an algorithm, we used landmarking in an adapted way that is described in the next section.

4.2 Time-Based Meta-Features

Landmarking have been successfully used in the past for different meta-learning approaches [15, 120]. The approach uses performance values of simple classifiers for predicting the performance of more sophisticated algorithms. Analogically, we use the run-time of the same simple learners for predicting the run-time

of a sophisticated classifier. The used classifiers are Naive Bayes, One-Nearest Neighbor, and Decision Stumps.

Additionally, we also included several times required for computing the other groups of meta-feature. The measured computation times of the following steps are used:

- calculating the statistical meta-features
- calculating the information theoretic meta-features
- calculating the model-based features including the creation of the decision tree

If the traditional meta-features are calculated anyway, e.g. for predicting the accuracy of classifiers, these measures do not require any additional computational effort.

In a practical scenario, these time-specific meta-features may be able to compensate the difference in performance of the computer the regression model was created on and the computer the regression model is used on. The features take the actual machine where the prediction is performed on into account because they are computed on the users machine. Therefore, more realistic and more useful predictions are possible. Since the approach estimates actual run-times, it depends on the actual hardware and features describing the environmental aspects should be used. To summarize, a time prediction approach should include meta-features that are able to describe two aspects: the dataset and the performance of the users computer. Using traditional meta-features and time-based measures, both requirements are fulfilled.

5 Evaluation

We evaluated the presented approach on real world datasets from the UCI machine learning repository [1] and StatLib [18]. The run-time of a grid search for five different classifiers are investigated. The used classifiers as well as their optimized parameters are listed in Table 1.

The complete evaluation was done using RapidMiner [3]. It is an open source data mining and pattern recognition framework implemented in Java. All times have been measured on an AMD Opteron using a single-threaded program.

The number of datasets is different for the classifiers since we only considered datasets with a run-time of the grid search in a defined interval. We excluded datasets with a run-time smaller than 10 seconds since the time measurements in this small time scales are more error-prone. Especially the landmarking times are too small for reliable results, but very low run-times are not important in a practical sense either. Additionally, datasets with a run-time of the algorithm greater than 24 hours have been neglected as well because of the computational effort. The exact number of datasets used for the single algorithms are shown in Table 2.

Table 1. The investigated classifiers and their parameters that have been optimized by a grid search

| Classifier | Parameter | Interval | Steps | Scale |
|---------------|--|--|----------------------------|--|
| k -NN | k weighted vote | [1, 1000] $\{\text{yes, no}\}$ | 100 | logarithmic |
| SVM | γ C | $[2^{-10}, 2^4]$ $[2^{-2}, 2^{12}]$ | 15 15 | quadratic quadratic |
| MLP | learning rate momentum decay | [0.01, 1.0] [0.0, 1.0] $\{\text{yes, no}\}$ | 10 10 | logarithmic linear |
| Ripper | sample ratio prune benefit pureness criterion | [0.1, 1.0] [0.1, 1.0] [0.1, 0.9] $\{\text{inf. gain, acc.}\}$ | 10 10 9 | linear linear linear |
| Decision Tree | min split size min leaf size min gain max depth confidence | [2, 100] [1, 100] [0.05, 5] [5, 100] [0.05, 5] | 10 10 10 10 10 | logarithmic logarithmic logarithmic logarithmic linear |

The presented approach was evaluated by a leave-one-out cross-validation for every algorithm. We used the regression variant of a Support Vector Machine, the ϵ -SVR, as meta-learning scheme. The parameters γ and C of the ϵ -SVR have been optimized by a grid search. LibSVM [7] was used as implementation.

Table 2. The number of samples (datasets) used within the evaluation for the different target classifiers

| Classifier | k -NN | SVM | MLP | Ripper | Decision Tree |
|------------|---------|-----|-----|--------|---------------|
| Samples | 68 | 123 | 123 | 58 | 118 |

We investigated different subsets of the described meta-features. The basic subsets are as follows:

simple: simple meta-features only

normal: all traditional meta-features

time: time measures of the traditional meta-features

tLM: time-landmarking

All meta-features were calculated using R and were normalized to the interval $[0, 1]$. The sets of features and several combinations of them were evaluated using two common performance measures of regression models: the correlation coefficient and the normalized absolute error.

Table 3. The Pearson product moment correlation coefficients of different sets of meta-features: feature sets that include the proposed time-landmarking features (*tLM*) achieve higher correlation values for four out of five classifiers

| Classifier | simple | normal | time | <i>tLM</i> | simple + time | simple + <i>tLM</i> | time + <i>tLM</i> | simple + time + <i>tLM</i> |
|---------------|--------|--------------|-------|--------------|---------------|---------------------|-------------------|----------------------------|
| k-NN | 0.817 | 0.702 | 0.803 | 0.867 | 0.883 | 0.905 | 0.875 | 0.933 |
| SVM | 0.766 | 0.653 | 0.569 | 0.876 | 0.764 | 0.863 | 0.873 | 0.863 |
| MLP | 0.810 | 0.814 | 0.647 | 0.733 | 0.799 | 0.825 | 0.768 | 0.816 |
| Ripper | 0.737 | 0.814 | 0.544 | 0.700 | 0.731 | 0.718 | 0.756 | 0.713 |
| Decision Tree | 0.850 | 0.844 | 0.575 | 0.821 | 0.833 | 0.878 | 0.864 | 0.874 |

5.1 Correlation

The Pearson product moment correlation coefficient (PMCC) of the actual run-time and the predicted run-time was calculated. The correlation between two variables X and Y is defined as

$$\rho_{X,Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}. \quad (1)$$

The results are values in the interval $[-1, 1]$. A value of one indicates a perfect positive relationship whereas minus one means the inverse, perfect negative relationship. If the correlation is zero, the two input variables are independent.

Table 3 shows the correlation coefficients for all five target classifiers and the investigated sets of meta-features. It is visible that good correlation values could be achieved. The correlation is at least 0.8 for all classifiers. Besides the simple meta-features, the time-landmarking approach seems to be particularly suitable for the task. The traditional meta-features achieved clearly better results only for the Ripper classifier.

5.2 Normalized Absolute Error

In addition to the correlation measure, the normalized absolute error was determined that serves as a comparison to a baseline. The absolute error of the prediction by the presented approach is divided by the absolute error of the prediction by a baseline method:

$$e = \frac{|t_m - t_p|}{|t_m - t_b|} \quad (2)$$

where t_m is the actual measured time, t_p the predicted time of the presented approach, and t_b the time predicted by the baseline method. For the baseline method, the predicted run-time is simply the average run-time of the classifier. Hence, the baseline method predicts the same run-time for every dataset.

If the normalized absolute error is smaller than one, the approach is better than the baseline. A value greater than one would indicate that predicting the

Table 4. The normalized absolute errors of different sets of meta-features. Like for the correlation, features sets including the time-landmarking features (*tLM*) achieve lower error rates for four target classifiers.

| Classifier | simple | normal | time | <i>tLM</i> | simple + time | simple + <i>tLM</i> | time + <i>tLM</i> | simple + time + <i>tLM</i> |
|---------------|--------|--------|-------|------------|---------------|---------------------|-------------------|----------------------------|
| k-NN | 0.619 | 1.107 | 0.673 | 1.283 | 0.596 | 0.601 | 1.142 | 0.611 |
| SVM | 0.562 | 0.772 | 0.727 | 0.482 | 0.553 | 0.521 | 0.512 | 0.467 |
| MLP | 0.458 | 0.516 | 0.756 | 0.671 | 0.480 | 0.440 | 0.625 | 0.448 |
| Ripper | 0.562 | 0.564 | 0.796 | 0.584 | 0.579 | 0.581 | 0.557 | 0.584 |
| Decision Tree | 0.535 | 0.625 | 0.860 | 0.459 | 0.542 | 0.434 | 0.468 | 0.464 |

average run-time is better than using meta-learning. Table 4 shows the normalized absolute errors. For this evaluation, the parameters of the ϵ -SVR were optimized according to this performance measure.

Error rates clearly below the baseline have been reached. Surprisingly, the error rates for the *k*-NN classifier are greater than one for certain meta-feature groups although the achieved correlation values are rather high.

5.3 Correlation between Features and Target Variable

Finally, the correlation between the different timing meta-features and the actual run-time was determined for each target classifier. Thereby, we wanted to evaluate the usefulness of the single time features and we wanted to determine how strong they are connected to the target time. In this part of the evaluation, no meta-learner is trained.

Figure 4 shows the correlation values between the six investigated time-based meta-features and the five considered target classifiers. It is visible that especially the run-times of the Naive Bayes and the One-Nearest Neighbor landmarks are highly correlated to the actual run-time. Surprisingly, the time of the model-based features that include a creation of a decision tree is not related to the time of the decision tree target classifier. The reason for this is that the two

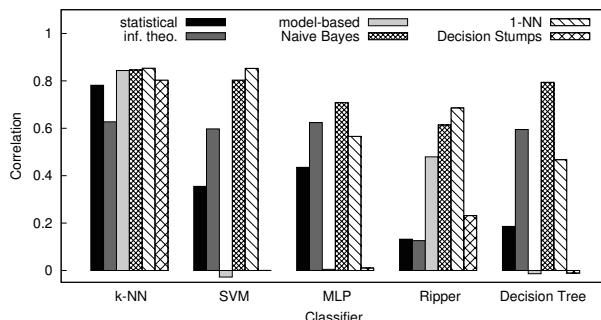


Fig. 4. The correlation between the time-based meta-features and the five target classifiers: Especially the Naive Bayes and the 1-Nearest Neighbor time-landmarkers achieve high correlation values

methods are quite different. In contrast to the target classifier, the decision tree of the model-based features is unpruned and does not include any parameter optimization. Moreover, different implementations were used.

It is also noticeable that all six timing meta-features achieve high correlation values for the k -NN target classifier. One reason for this is probably the simplicity of the algorithm.

6 Conclusion

A method for predicting the run-time of a classification algorithm was presented. Meta-learning was used to estimate the time needed for a grid search over multiple parameters. Therefore, independent regression models have been learned for multiple target classifier. Since the run-time depends on the parameters of the classifier and the dataset itself, meta-features are used for creating the prediction model. In addition to traditional meta-features, measures that are specialized for time prediction were proposed. The run-time of simple learners as well as the time needed for calculating the traditional meta-features were used.

The presented approach was evaluated on real world datasets from the UCI machine learning repository and StatLib. Different subsets and combinations of the meta-features were evaluated according to their suitability for the task. Therefore, the correlation coefficient and the normalized absolute error were used.

The results show that the approach is able to reasonably predict the run-time of different algorithms. Especially the proposed time-landmarking measures improved the results. The investigated correlation values between time-based meta-features and the target run-time also show that the 1-Nearest Neighbor and the Naive Bayes time-landmarkers are suitable for this task.

References

1. Asuncion, A., Newman, D.: UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences (2007), <http://www.ics.uci.edu/~mlearn/MLRepository.html>
2. Bensusan, H., Giraud-Carrier, C.: Casa batló is in passeig de gràcia or how landmark performances can describe tasks. In: Proceedings of the ECML 2000 Workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination, pp. 29–46 (2000)
3. Bensusan, H., Giraud-Carrier, C., Kennedy, C.: A higher-order approach to meta-learning. In: Proceedings of the ECML 2000 Workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination, pp. 109–117 (June 2000)
4. Bensusan, H., Giraud-Carrier, C.G.: Discovering task neighbourhoods through landmark learning performances. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 325–330. Springer, Heidelberg (2000)

5. Bensusan, H., Kalousis, A.: Estimating the predictive accuracy of a classifier. In: De Raedt, L., Flach, P. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, pp. 25–36. Springer, Heidelberg (2001)
6. Brazdil, P., Soares, C., da Costa, J.P.: Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results. Machine Learning 50(3), 251–277 (2003)
7. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001), software, <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
8. Engels, R., Theusinger, C.: Using a data metric for preprocessing advice for data mining applications. In: Proceedings of the European Conference on Artificial Intelligence (ECAI 1998), pp. 430–434. John Wiley & Sons, Chichester (1998)
9. Fürnkranz, J., Petrak, J.: An evaluation of landmarking variants. In: Giraud-Carrier, C., Lavrač, N., Moyle, S., Kavšek, B. (eds.) Proceedings of the ECML/PKDD Workshop on Integrating Aspects of Data Mining, Decision Support and Meta-Learning (IDDM 2001), Freiburg, Germany, pp. 57–68 (2001)
10. Gama, J., Brazdil, P.: Characterization of classification algorithms. In: Pinto-Ferreira, C., Mamede, N. (eds.) EPIA 1995. LNCS, vol. 990, pp. 189–200. Springer, Heidelberg (1995)
11. Köpf, C., Taylor, C., Keller, J.: Meta-analysis: From data characterisation for meta-learning to meta-regression. In: Proceedings of the PKDD 2000 Workshop on Data Mining, Decision Support, Meta-Learning and ILP (2000)
12. Lindner, G., Studer, R.: Ast: Support for algorithm selection with a cbr approach. In: Recent Advances in Meta-Learning and Future Work, pp. 418–423 (1999)
13. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: Yale: Rapid prototyping for complex data mining tasks. In: Ungar, L., Craven, M., Gunopulos, D., Eliassi-Rad, T. (eds.) KDD 2006: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 935–940. ACM, New York (2006)
14. Peng, Y., Flach, P., Soares, C., Brazdil, P.: Improved dataset characterisation for meta-learning. In: Lange, S., Satoh, K., Smith, C. (eds.) DS 2002. LNCS, vol. 2534, pp. 141–152. Springer, Heidelberg (2002)
15. Pfahringer, B., Bensusan, H., Giraud-Carrier, C.: Meta-learning by landmarking various learning algorithms. In: Proceedings of the Seventeenth International Conference on Machine Learning, pp. 743–750. Morgan Kaufmann, San Francisco (2000)
16. Segrera, S., Pinho, J., Moreno, M.: Information-theoretic measures for meta-learning. In: Corchado, E., Abraham, A., Pedrycz, W. (eds.) HAIS 2008. LNCS (LNAI), vol. 5271, pp. 458–465. Springer, Heidelberg (2008)
17. Sohn, S.Y.: Meta analysis of classification algorithms for pattern recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 21(11), 1137–1144 (1999)
18. Vlachos, P.: StatLib Datasets Archive. Department of Statistics, Carnegie Mellon University (1998), <http://lib.stat.cmu.edu>
19. Wolpert, D.H.: The lack of a priori distinctions between learning algorithms. Neural Comput. 8(7), 1341–1390 (1996)

Human-Machine Corpus Analysis for Generation and Interaction with Spoken Dialog Systems

Roland Roller, Tatjana Scheffler, and Norbert Reithinger*

DFKI GmbH, Projektbüro Berlin
Alt-Moabit 91c
10559 Berlin, Germany
`{firstname.lastname}@dfki.de`

1 Introduction

This paper describes a new approach to language generation for simulated users based on the construction of flexible templates extracted from a corpus. In our opinion a realistic user simulation on the speech level is based on two parts: user behavior and language generation. In this work we mainly concentrate on the language generation for simulated user interaction with spoken dialog systems (SDS). The presented approach could be used as part of a user simulation for intensive end-to-end system tests and evaluations and for testing purposes of the speech recognition and natural language understanding modules of an SDS.

We present our semi-automatic analysis of a human-machine corpus, the corpus-based language generation process, which generates realistic user replies on the basis of their usage frequency and verbosity, and a speech enrichment approach to increase the variability of the output. We demonstrate in user simulation experiments realized with synthesized speech, that the generated output is comparable in its variability to the utterances of human testers.

2 Related Work

There has been a lot of work in the area of user simulation for testing and evaluation of SDS. In this work we are focussing on the interaction level of simulated users with the spoken dialog system. [4] describes three different interaction levels: the intention level (usually feature-value pairs) as in [4][11][6][7], the text level [1][8][10], or the speech level. For this, there are two main approaches: using existing audio snippets of real users [9] or synthesized speech [3][5]. Both speech level approaches have their advantages. While recorded audio of real users is more realistic and covers several different users with different voices or accents, realistic interaction behavior, and background noises, the TTS approach is dynamic and can easily generate completely new user replies.

* This research was funded by the IBB through the ProFIT framework, grant #10140648, sponsored by the European Regional Development Fund, and carried out in cooperation with the Quality and Usability Lab at Technical University Berlin. We would like to thank Florian Gödde and Sebastian Möller.

The language generation module (for the text and speech levels) of previous user simulations is mainly realised with a rule-based [3] or template-based approach. Corpus-based language generation can depend on a corpus out of the same domain, a similar domain, or it could try to adapt existing corpus data to a new domain. In [13], the corpus contains no real human-computer interactions, but rather is generated by a user simulation.

Furthermore, several user simulations generate errors, either on the SDS recognition side, or on the user input side. Error simulation on the SDS side involves the generation of lexical errors [2], recognition errors [6], or the manipulation of confidence scores [7, 8]. Simulation on the user side includes spontaneous speech such as e.g. fragments or fillers [3, 5]. The usage of pre-recorded audio of real users in [9] includes probably the most realistic error level, containing fillers, pauses, background noise and flawed pronunciations, but is always restricted to the given sentences.

3 Human-Machine Corpus Analysis

The basis of our work is the VOICE Awards Corpus, a collection of 1970 dialogs with 120 German-language commercially deployed spoken dialog systems from a broad range of domains (see [12] for more detail). Altogether the corpus contains more than 23,000 user inputs. The corpus is hand-annotated with dialog acts, errors, task success, and repetitions.

In order to extract user templates, we developed semi-automatic methods for further annotating the corpus. First, redundant user information were counted and then removed, using Perl scripts with regular expressions: fillers (found with a pre-defined list), repetitions (which are already manually annotated), self-corrections, and polite phrases. Second, we used semi-automatically compiled lists [2] to identify named entities (e.g. street names, cities, etc.) and replaced the value with its category name.

Next we defined regular expressions covering different input styles for recurring input types such as number sequences, dates, time, etc. and replaced this information with its input style name. We also automatically identified the usage of slot values (usage of an option given in an SDS request with several alternatives) within a user reply. We found slot values by calculating the string similarity of the user input with each system request slot, disregarding articles.

Finally, we categorized the user reply verbosity into four types: raw (information), short (phrase), fragment, sentence; depending on its structure and its length. For the language generation process, we group the obtained templates by their dialog act, included content categories, and verbosity. We save these data together with their occurrence frequency in a database.

Table 1 shows the most frequent user replies by dialog act and reply information with their rate of occurrence and total frequency in the corpus. Accepts, rejects and slots, which together account for more than 40% of all user inputs, are generally short (the number of words within a slot name are not considered) and do not come in many variants (Table 2).

Table 1. User Reply Usage. **Slot:** User chooses a given alternative from an alternative question. **Option:** User chooses a given option from an instruction. **Free Input:** User reply to an open question.

| template | occurrence | frequency |
|----------------|------------|-----------|
| accept/reject | 24.49% | 5672 |
| slot | 17.47% | 4047 |
| sequence | 14.27% | 3305 |
| system_command | 10.41% | 2412 |
| single number | 3.65% | 846 |
| bad_input | 3.32% | 770 |
| date | 3.01% | 696 |
| option | 2.91% | 673 |
| city | 2.66% | 617 |
| free input | 2.53% | 586 |
| name | 2.42% | 561 |
| no_input | 2.23% | 516 |

Table 2. User input verbosity: average and maximum number of words per type of template; total number of templates (variety) per type

| template type | avg. words | max. words | variety |
|---------------|------------|------------|---------|
| slot | 1.0239 | 5 | 12 |
| accept | 1.0018 | 3 | 13 |
| reject | 1.0142 | 4 | 16 |
| free input | 3.4976 | 22 | 227 |

The data show that German SDS usually expect similar kinds of information as input and users tend to reply in short, repetitive phrases. The biggest challenge for template-based generation is *free input* (usually after an open question), which contains relatively many words and offers a high amount of variation.

4 User Input Generation

This section describes our corpus-based probabilistic language generation approach. Input for the generation process is (1) the result of our action planner (AP) module, (2) a predefined user task/goal and (3) user templates [12]. The AP calculates the intended user dialog act and the kind of information we should provide, depending on the given context, SDS request and user model. The intended verbosity is part of the user model. The user task defines the goal constraints and the required user information to run a simulation. The defined categories must match the information in the extracted templates (e.g., we must have templates to provide a destination if this is part of the user goal). The person who is setting up the simulation is responsible to define meaningful tasks. The user templates are constructed out of the generalized sentences obtained by the corpus processing procedure discussed above and are grouped by dialog act, included information and verbosity.

Based on the results from the Action Planner, we identify all matching user templates according to the given constraints and choose one of them by its probability (the frequency of a given template divided by the frequency of all matching templates). If there is no matching template set according to the given verbosity, we expand the search space by neglecting the verbosity. In case we

Table 3. User reply variety: Number of templates used

| | task 1 | task 2 | task 3 | task 4 | task 5 |
|------------|--------|--------|--------|--------|--------|
| Users | 34 | 45 | 61 | 30 | 36 |
| Simulation | 34 | 31 | 59 | 25 | 31 |

Table 4. User reply variety: Most frequently used template types with average and maximum word length, and number of different templates used (variants)

| template | Users | | | Simulation | | |
|------------|------------|------------|----------|------------|------------|----------|
| | avg. words | max. words | variants | avg. words | max. words | variants |
| slot | 1.0361 | 4 | 2 | 1.0000 | 1 | 1 |
| accept | 1.0985 | 4 | 3 | 1.0150 | 2 | 6 |
| reject | 1.0000 | 1 | 2 | 1.0095 | 2 | 3 |
| free input | 2.7591 | 19 | 122 | 2.0171 | 7 | 64 |

could not find any fitting template at all, we back off to rule based generation of the user reply template. After selecting a template, we replace the contained information categories with our goal information.

This probabilistic template approach has several advantages. First, the random selection of user templates depending on their occurrence frequency will represent a realistic usage of different input concepts. Furthermore the extraction of user templates is domain independent.

To generate even more dynamic and realistic user input, we enrich the output sentence by further features we have identified in our corpus. So far, we consider three additional input characteristics that human users utilize: (1) polite phrases (used at least once by 0.51% of users in our corpus), (2) self corrections (0.38%), and (3) fillers (1.21%). All generated user inputs are enriched with these features, according to their frequency as seen in the corpus.

5 User Simulation Experiments

To test our generation, we set up experiments with a commercial German SDS from the telecommunication domain. The system provides customers information about their own and other telephone payment plans and gives them opportunity to change certain settings. Our experiments were realized via ISDN line with MaryTTS. In preliminary tests we have shown that Mary-Pavoque reached comparable system-side recognition quality as human users. We defined five tasks of different complexity. We hired 29 student subjects to solve each task once, and ran simulations using the same tasks.

The subjects achieved a total task success of approximately 85.14% in 148 calls, the simulated users 92.74% (N=193). The simulations automatically hung up at a time limit of four minutes. Table 3 shows the number of different user templates used within each task, not including speech enrichment. Table 4 shows

that the simulated user input is comparable in its length and variety to the real users. Only the free input of users offers more flexibility. Even if real user inputs offer a bit more flexibility concerning the selection of different replies, the results show that we can reproduce quite realistic and flexible user input.

6 Conclusion

We describe an approach to language generation for simulated end-to-end user tests based on flexible templates constructed from a corpus. We also augment the user replies by additional features such as self-corrections and fillers. This approach is a dynamic and easy way to cover many different kinds of user utterances with realistic frequencies. Our experiments show that the generated replies are comparable in variety to the utterances of real users.

References

1. Ai, H., Litman, D.: Comparing real-real, simulated-simulated, and simulated-real spoken dialogue corpora. In: Proc. AAAI Workshop on Statistical and Empirical Approaches for SDS (2006)
2. Ai, H., Weng, F.: User simulation as testing for spoken dialog systems. In: Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue, Columbus, Ohio, pp. 164–171 (2008)
3. Chung, G.: Developing a flexible spoken dialog system using simulation. In: Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL 2004), Barcelona, Spain, pp. 63–70 (July 2004)
4. Eckert, W., Levin, E., Pieraccini, R.: User modelling for spoken dialogue system evaluation. In: Proceedings of ASRU, pp. 80–87 (1997)
5. Filisko, E., Seneff, S.: Developing city name acquisition strategies in spoken dialogue systems via user simulation. In: Proceedings of SigDial, Lisbon, Portugal, vol. 6, pp. 144–154 (2005)
6. Georgila, K., Henderson, J., Lemon, O.: User simulation for spoken dialogue systems: Learning and evaluation. In: Proceedings of Interspeech (2006)
7. Griol, D., Hurtado, L.F., Segarra, E., Sanchis, E.: A statistical approach to spoken dialog systems design and evaluation. Speech Communication 50, 666–682 (2008)
8. Jost Schatzmann, B.T., Young, S.: Error simulation for training statistical dialogue systems. In: IEEE Automatic Speech Recognition and Understanding Workshop 2007, Kyoto, Japan (2007)
9. López-Cózar, R., de la Torre, A., Segura, J., Rubio, A.: Assessment of dialog systems by means of a new simulation technique. SpeCom 40, 387–407 (2003)
10. Jung, S., Lee, C., Kim, K., Lee, G.G.: An integrated dialog simulation technique for evaluating spoken dialog systems. Computer Speech and Language, 9–16 (2008)
11. Scheffler, K., Young, S.: Probabilistic simulation of human-machine dialogues. In: Proc. ICASSP, pp. 1217–1220 (2000)
12. Scheffler, T., Roller, R., Reithinger, N.: Speecheval – evaluating spoken dialog systems by user simulation. In: Proceedings of the 6th IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems, Pasadena, CA, pp. 93–98 (2009)
13. Wang, C., Seneff, S., Chung, G.: Language model data filtering via user simulation and dialogue resynthesis. In: Proc. of INTERSPEECH (2005)

Comparison of Laser-Based Person Tracking at Feet and Upper-Body Height

Konrad Schenk, Markus Eisenbach,
Alexander Kolarow, and Horst-Michael Gross*

Neuroinformatics and Cognitive Robotics,
Ilmenau University of Technologies

Abstract. In this paper, a systematic comparative analysis of laser-based tracking methods, at feet and upper-body height, is performed. To this end, we created a well defined dataset, including challenging but realistic person movement trajectories, appearing in public operational environments, recorded with multiple laser range finders. In order to evaluate and compare the tracking results, we applied and adapted a performance metric, known from the Computer Vision area. The dataset in combination with this performance metric enables us to perform systematic and repeatable experiments for benchmarking laser-based person trackers.

Keywords: Laser Range Finder, Person Tracking, People Tracking, Feet Height, Upper-Body Height, MOTP, Benchmark.

1 Introduction

Detecting and tracking persons with laser range finders is a common method in the field of intelligent service robotics. On a robotic platform, the laser range finder is often placed at feet height ($\approx 20\text{cm}$ above ground), to safely navigate around other objects. Additionally, this configuration is also used for person tracking. Another possible configuration is the positioning at upper-body height ($\approx 110\text{cm}$ above ground). This may produce better tracking results. Yet, this is not evaluated, since most common robotic platforms do not have laser range finders at upper-body height. Other research groups focus on people tracking with multiple laser range finders. Multiple lasers can be used for surveillance of large public areas, e.g. airports and for passenger traffic flow control. In such a scenario, a comparison of the tracking quality between feet and upper-body height was not performed. In this paper, we address this comparison.

We compare two state of the art algorithms for person tracking with multiple laser range finders at feet and upper-body height. The algorithms are evaluated in several experiments, using a representative generic dataset. A novel quality metric is used in order to compare alternative tracking algorithms for person

* This research was funded by the German Federal Ministry of Education and Research (BMBF) as part of the project APFel.

tracking using laser range finders. The results can be used for selecting a suitable algorithm for a designated operational environment.

The scenario we focus on, is the surveillance of an airport. The tracking of people is achieved by cameras, but we additionally use the laser range finders to extract a global movement model and a prediction graph between all cameras. Such a graph is important, since the cameras do not overlap, and a global tracking needs additional information for correct interpolation of trajectory pieces, between different cameras.

The remainder of this paper is organized as follows: We present the state of the art for tracking persons with laser range finders at feet and upper-body height in Section 2. The most promising and real-time capable tracking algorithms are also described in this section in more detail. These trackers were then used in our experiments (Section 3). Additionally, we present the experimental setting, our test framework, and the experiments including a detailed comparison between the two setups. We complete with a conclusion and a perspective on further work.

2 Tracker

Laser range finders have been widely used for people tracking. Most applications on mobile robotic platforms use one finder at feet height [2][11][13][17]. Also static installations at feet height were used in research [1]. An elaborated detection mechanism on feet height, which could be used with a generic tracking mechanism, was presented in Arras et al. [3]. Shao et al. [14] utilized a multi-person tracking mechanism using multiple stationary laser range finders at feet height in combination with a sophisticated motion model.

Only little research was conducted on upper-body height. In [6] and [9], hybrid mechanisms for tracking people at feet height and upper-body height were developed for a mobile robot. In [10], multiple static laser range finders were utilized to predict the movements of people in a public area. Glas et al. [8] also employs multiple static laser range finders at upper-body height, in order to track people and improve self localization of mobile robots.

The usage of scans at multiple heights was also investigated in [12][15]. The evaluation of these novel methods may lead to promising results in further work.

For comparison, we have chosen the feet tracker of Shao [14], since it can be implemented with real time capabilities and proves good results. Additionally the use of a background model simplifies the detection of people. The tracking mechanism presented by Glas et al. [8] also provides a good tracking performance. We use a similar, but simpler, real-time multi person tracker for the laser range finders at upper-body height.

In order to track people in laser range data, several tasks need to be addressed. At first, an adequate preprocessing of the data is needed. Afterwards, the tracking itself can be performed. Both tasks are now shortly described.

In the preprocessing, all laser range finders are aligned in a global coordinate system. This way, the range scans from all laser finders can be transformed into

a global set of points. In the next step, we perform a histogram-based background subtraction. This enables us to extract all points referring to dynamic objects. Solely, those points are used to track the people. Afterwards, a mean shift clustering, similar to the work in [7] (which is also used in [14]), is applied, in order to segment the point clouds into clusters. Those clusters are used to detect people afterwards.

Under the assumption, that people's movement is Markovian, both tracking methods apply multiple particle filters for estimating the current state of all people. A simple particle filter, as described in [16], is used in both tracking methods.

2.1 Feet Tracker

The used feet tracker follows the implementation of [14]. Every person is tracked by an individual particle filter. In order to initialize a filter, a person needs to be detected. This is done by spatio-temporal analysis of the clusters obtained by the mean shift algorithm. When a foot is pivoting around the other one, a certain sequence can be observed in the clustered points, which can be seen in Figure 1. At first, every foot has its own cluster of points. After the distance of the moving foot to the standing foot falls below a certain threshold, the points of both feet are merged into one cluster. If the moving foot departs from the standing foot afterwards, the big cluster is split up into two clusters again. Therefore, a person walking one step can be detected at two specific time frames. In order to detect a splitup or a merge, it is sufficient to analyze the overlapping areas of the clusters in two adjacent time frames. After such a sequence is found, a particle filter can be initialized at this position, in order to track the newly detected person.

The state \mathbf{z}_t of one persons movement consists of eight parameters: (x_t^l, y_t^l) and (x_t^r, y_t^r) are the positions of the left and right feet, s_t is the walking stride, T_t the walking period, α_t the walking direction and γ_t the walking phase.

The walking model implements a periodic function and is described in Equations 1 and 2, where δ is the duration between the last and the current time step. Further details and the derivation can be found in [14].

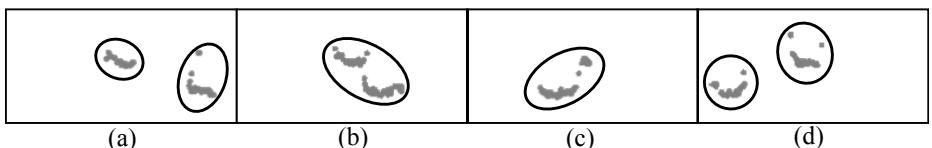


Fig. 1. Merging and splitting of the clusters belonging to the feet of a person. The person is walking from right to left, while the right foot is standing and the left one is passing by. In (a), both feet have their own cluster. In (b) and (c), both feet are close enough to cause their scan points to be merged into one cluster. In (d), the left foot reached a distance high enough from the right foot, so that the big cluster is split up into two clusters again.

$$\begin{aligned}
& \text{if } (\text{mod}(\gamma_t, 2\pi) > \pi) \\
& \quad m_l = s_t |\cos(\gamma_t) - \cos(\gamma_{t-1})| \\
& \quad m_r = 0 \\
& \text{else} \\
& \quad m_l = 0 \\
& \quad m_r = s_t |\cos(\gamma_t) - \cos(\gamma_{t-1})| \\
& \text{end}
\end{aligned} \tag{1}$$

$$\begin{pmatrix} x_t^l \\ y_t^l \\ x_t^r \\ y_t^r \\ s_t \\ T_t \\ \gamma_t \\ \alpha_t \end{pmatrix} = \begin{pmatrix} x_{t-1}^l \\ y_{t-1}^l \\ x_{t-1}^r \\ y_{t-1}^r \\ s_{t-1} \\ T_{t-1} \\ \gamma_{t-1} \\ \alpha_{t-1} \end{pmatrix} + \begin{pmatrix} \cos(\alpha_{t-1}) \cdot m_l \\ \sin(\alpha_{t-1}) \cdot m_l \\ \cos(\alpha_{t-1}) \cdot m_r \\ \sin(\alpha_{t-1}) \cdot m_r \\ 0 \\ 0 \\ 2\pi \frac{\delta}{T_{t-1}} \\ 0 \end{pmatrix} \tag{2}$$

The observation model takes the scan points directly into account. If $Z = \{x^{(j)}, y^{(j)} | j = 1, \dots, N\}$ are the observed foreground points, the weight of a sample can be updated with Equation 3–7. For further details, please refer to [14].

$$o_l = \max \left(\sum_{j=1}^N \exp \left(-\frac{(x_t^l - x^{(j)})^2 + (y_t^l - y^{(j)})^2}{2\delta^2} \right), P_0 \right) \tag{3}$$

$$o_r = \max \left(\sum_{j=1}^N \exp \left(-\frac{(x_t^r - x^{(j)})^2 + (y_t^r - y^{(j)})^2}{2\delta^2} \right), P_0 \right) \tag{4}$$

$$s_d = (x_t^r - x_t^l) \cdot \cos(\alpha_t) + (y_t^r - y_t^l) \cdot \sin(\alpha_t) \tag{5}$$

$$o_b = \exp \left(\frac{-(s_d + s_t \cdot \cos(\gamma_t))^2}{2h^2} \right) \tag{6}$$

$$w_t^{(i)} = o_l \cdot o_r \cdot o_b \tag{7}$$

When two persons are walking in a close proximity, particle filters can switch to people already being tracked. Therefore, a repelling term between the filters is included to avoid hijacking. This ensures, that one person is only tracked by a single particle filter.

2.2 Upper-Body Tracker

For the upper-body tracker we utilize a more common approach, similar to [8]. Every person is tracked by an individual particle filter. The detection of a person is done by evaluating the size of a scan point cluster, using its eigenvalues. A threshold for the biggest eigenvalue describes the minimal human body size.

The upper boundary of the cluster sizes is given by the mean shift algorithm and does not need to be evaluated. In order to track the corresponding person, a particle filter is initialized at the cluster position. A more comprehensive detection method could also be applied, but due to the background subtraction, this simple approach is sufficient.

The state \mathbf{z}_t of a person consists of four parameters. (x_t, y_t) is the position of the person, α_t the movement direction and v_t the velocity along this direction. The motion model implements a simple linear movement, as shown in Equation 8, where δ is the duration between the last and the current time step.

$$\mathbf{z}_t = \begin{pmatrix} x_t \\ y_t \\ \vartheta_t \\ v_t \end{pmatrix} = \mathbf{z}_{t-1} + \begin{pmatrix} \cos(\vartheta_{t-1}) \\ \sin(\vartheta_{t-1}) \\ 0 \\ 0 \end{pmatrix} \cdot \delta \cdot v_{t-1} \quad (8)$$

The observation model for the upper-body tracker also takes the scan points directly into account. If $Z = \{x^{(j)}, y^{(j)} | j = 1, \dots, N\}$ are the observed foreground points, the weight of a sample can be updated with Equation 9.

$$w_t^{(i)} = \sum_{j=1}^N \exp \left(- \frac{\left((x_t^{(i)} - x^{(j)})^2 + (y_t^{(i)} - y^{(j)})^2 \right)}{\zeta^2} \right) \quad (9)$$

To prevent a particle filter to focus on a person already tracked by another one, only points in its proximity are used in the weight update. The proximity is evaluated by Voronoi clustering (see [18]).

Both tracking methods proved to have real-time capabilities and were used to obtain the results described in the next section.

3 Experiments

In the previous section, we introduced the selected algorithms for tracking persons with laser range finders at feet and upper-body height. Here, we compare them in several categories, including different walking speed, linear and non-linear trajectories, speed and direction changes, multiple persons, and accessories carried by people.

3.1 Experimental Setup

The experiments were conducted in a room, measuring $5.5 \times 11\text{m}$. The test persons followed specified lines on the floor. Four laser range finders were used. Two were installed at feet height (20cm) and two at upper-body height (110cm), standing at the same position. They were placed 2m left and right to the center of the trajectory. Another alternative configuration with all laser range finders installed on one side of the room was evaluated. But since no different results

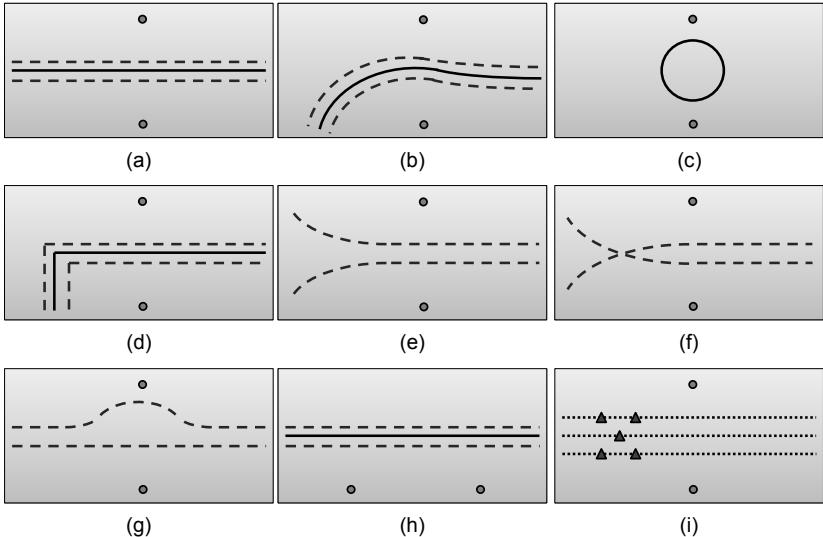


Fig. 2. Specified test trajectories. (a,h,i) Straight line, (b) Curve, (c) Circle, (d) Corner, (e) Join, (f) Cross, (g) Evading – for one (continuous line), two (dashed lines), three (dotted lines) and five persons (triangles on dotted lines). The positions of the laser range finders are marked as filled circles: (a-g,i) normal setup with laser range finders on opposite sides, (h) alternative setup with laser range finders on the same side.

were obtained, we do not describe this subaspect in more detail. Seven kinds of trajectories were included in the experiments: Straight line, curve, circle, corner, join, cross and evading. Figure 2 shows all specified trajectories. The used laser range finder is LMS151 from SICK. It scans 270° with a 0.5° resolution at 50Hz. Small people or children are not harmed by the laser at upper-body height, since it is eye safe.

In the following, we describe the experiments in detail. Each experiment was based on typical situations, observed at airports. The result of one test run is shown in Figure 3.

In experiment 1 "Speed", the person walked with three different speeds (slow, normal, fast) on a straight line (Figure 2a). For each speed, three different persons were recorded three times, resulting in nine trajectories (for all following experiments also nine trajectories were extracted).

Experiment 2 "Non-linear trajectories" examined the tracker for a single person walking on curves. Experiments included a curve (Figure 2b), a circle (Figure 2c) and evading (Figure 2g). This tested the ability of the algorithms to track non-linear movements.

In experiment 3 "Change of direction and speed", sudden changes of the walking direction or speed were evaluated. The direction change was emulated by a single person walking on trajectory (d) of Figure 2. The speed changed when the person stopped at the middle of the trajectory (a), waited some time and moved on. In a third test, the persons were additionally asked to stop and

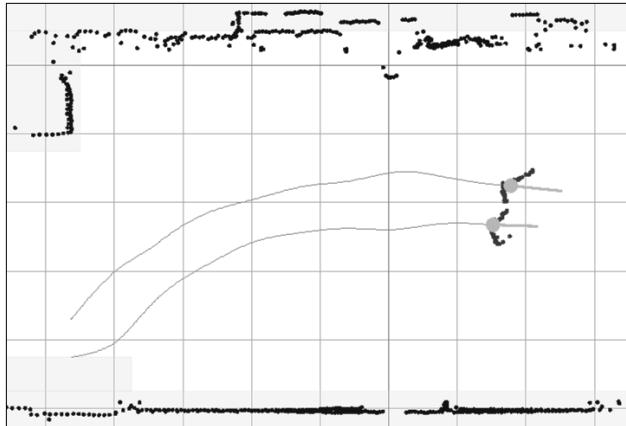


Fig. 3. Single exemplary tracking result of trajectory (b)

turn around in order to look behind. This experiment addressed the ability to track persons with non-linear movements.

Experiment 4 "Multiple persons" examined the tracking precision with an increasing number of people. In the case of multiple persons in the scene, occlusions appeared more often. The tracker had to cope with only partly visible and fully occluded persons. Trajectories (a,b,d) from Figure 2 were used for two and three persons and trajectory (i) for five persons.

In experiment 5 "ID switch", the ability to differentiate trajectories of persons, walking close to each other, was evaluated. Two cases were considered: In the first case, the trajectories of the two persons drew near but did not cross (Figure 2e). In the second case the trajectories intersected (Figure 2f). The tracker needed to follow the two persons without switching the IDs.

Experiment 6 "Accessories" tested the influence of accessories to the tracking result. The following situations were considered: persons carrying and pulling suitcases, pushing a baggage cart, carrying large items in front of the body, using a cane and wearing a shoulder bag or a long skirt. For this experiment, we used trajectory (a) from Figure 2.

3.2 Precision Measure

In order to evaluate the tracking results, we adapt the multi object tracking precision (MOTP), a widely used measure in computer vision. The original approach uses the overlap between the bounding boxes of the tracker's hypothesis and the ground truth bounding boxes of the objects of interest, e.g. persons, in every frame. Details can be found in [45].

In the following, we present our adapted version for laser-based tracking. In contradiction to the bounding box in computer vision, the tracked object is represented by a point, when using laser range finders. Therefore, we can not utilize the overlap with the ground truth. Instead, we use the distance. The

matching value (m) should be 1, if the distance is smaller than a threshold (similar to a minimum overlap), and between 0 and 1 otherwise. We defined the following matching term:

$$m = \min\left(1, \frac{h}{d^3}\right), \quad (10)$$

where d is the distance to the ground-truth in meters, and h is a threshold. We use $h = 0.008$ which values a hypothesis correct, if $d \leq 20$ cm.

The remaining calculations are inherited from the original MOTP: First, every trajectory of the tracker (T) has to be matched to the best suitable ground truth trajectory (G). Every trajectory T can only be matched to one trajectory G and vice versa. If the amount of trajectories T is smaller than the the amount of trajectories G, virtual trajectories T are added with a matching term of $m = 0$. Unmatched trajectories T also get a matching term of $m = 0$. With this combinatoric problem solved, the MOTP can be calculated as

$$MOTP = \frac{\sum_{i=1}^F \sum_{j=1}^{N_i^G} m_{ij}}{\sum_{i=1}^F \max(N_i^T, N_i^G)}, \quad (11)$$

where F is the number of frames, m_{ij} is the matching value between ground truth trajectory j and its belonging trajectory T in frame i (Equation 10). N_i^T and N_i^G are the number of trajectories T and trajectories G in frame i .

In Figure 4, we show exemplary trajectories and their MOTP. The measure with the recommended threshold gives a very intuitive interpretation of the result. If the tracking result is very similar to the ground truth, the MOTP is 1.0 (Figure 4e). One particle filter tracking two persons results in a MOTP near 0.5 (Figure 4a). Similar, two particle filters tracking a single person, also results in a MOTP near 0.5 (Figure 4b), and three filters tracking two persons result in a MOTP near $\frac{2}{3}$ (Figure 4d). The MOTP decreases whenever gaps in the trajectories are present or the trajectory is too far away from the ground truth

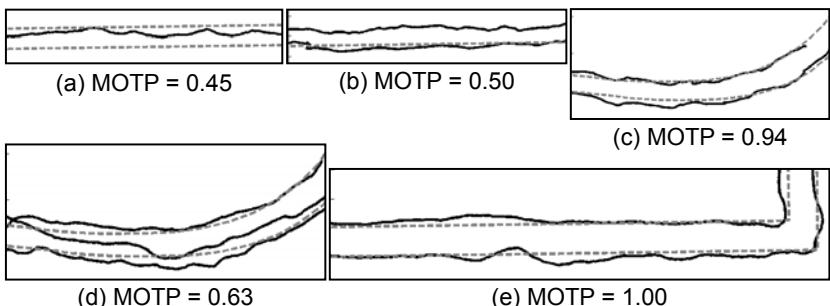


Fig. 4. Spatial plot of example trajectories in topview. Continuous lines represent the calculated trajectories and dashed lines represent the ground truth. Corresponding MOTP are shown below the plots.

(Figure 4c). When a particle filter, belonging to person A, switches to another person B, it has a high distance to its original ground truth. This ID-switch is penalized, since the high distance of person B to the ground truth of A results in a low MOTP.

3.3 Results Feet Tracker

Table 1, 3, 5 and 6 show, that despite the difficult setups, the feet tracker has achieved good results. In contrast, high initial speeds of people can cause the feet tracker to fail, as shown in Table 1. While running, the movement of people's feet differs, compared to normal walking. The motion model of the feet tracker is not designed to track those running movements and therefore fails tracking.

Movements of one or two persons pose no problem. Three or five people walking in close proximity to each other, are resulting in erroneous tracking. This can be ascribed to the initialization of the particle filters. Particle filters, for people entering the scene, are not initialized immediately. As described in Section 2.1, detections are only triggered at specific time frames. Therefore, particle filters for multiple persons, entering the scene at the same time, are not initialized at the same moment. Persons not being tracked by a particle filter, disturb nearby particle filters. Occlusions can also prevent the tracking mechanism from detecting a person for a longer period of time. This can cause identity switches and tracking errors as shown in Table 4.

Moving objects, other than the feet, also disturb the tracking mechanism. The suitcase and the baggage cart, mentioned in Table 6, cause scan points in the proximity of the tracked person. Therefore, the corresponding particle filter is often hijacked, or fails to estimate the movement parameters. This leads to bad tracking results.

3.4 Results Upper-Body Tracker

The upper-body tracker achieves very good results in nearly every experiment. But multiple people walking in close proximity to each other can cause the tracker to fail. This is shown in Table 4. The failures are caused by occlusions and identity switches: If a person is occluded for some time, its particle filter is often hijacked by a nearby person. As shown in Table 6, a baggage cart can disturb the tracking mechanism, too. The cart is detected by the laser range finders at upper-body height and causes a second particle filter to be initialized. Therefore, the baggage cart, and the person pushing it, are tracked, which results in a bad assessment of the tracker in such a case.

3.5 Comparison and Discussion

Both tracking methods performed well, but in nearly every experiment, the upper-body tracker achieved better results. It needs to be mentioned, that the experiments focused on difficult situations, which are not occurring in real applications frequently. Therefore, the results presented in 14 are plausible, despite of the results their tracking method achieved in our experiments.

Table 1. Experiment 1
"Speed"

| Scene | FT | UBT |
|-------------|------|------|
| slow | 1.0 | 1.0 |
| normal | 0.84 | 1.0 |
| fast | 0.50 | 0.88 |
| Avg. | 0.78 | 0.96 |

Table 2. Experiment 2
"Non-linear trajectories"

| Scene | FT | UBT |
|-------------|------|------|
| curve | 0.83 | 1.0 |
| circle | 0.78 | 0.99 |
| evading | 0.97 | 0.95 |
| Avg. | 0.86 | 0.98 |

Table 3. Experiment 3
"Change of direction and speed"

| Scene | FT | UBT |
|-------------|------|------|
| corner | 0.83 | 0.89 |
| stop | 0.81 | 0.91 |
| turn | 0.84 | 0.90 |
| Avg. | 0.83 | 0.90 |

Table 4. Experiment 4
"Multiple persons"

| Scene | FT | UBT |
|-------------|------|------|
| 1 pers. | 0.84 | 1.0 |
| 2 pers. | 0.85 | 0.90 |
| 3 pers. | 0.57 | 0.91 |
| 5 pers. | 0.53 | 0.66 |
| Avg. | 0.70 | 0.87 |

Table 5. Experiment 5
"ID switch"

| Scene | FT | UBT |
|-------------|------|------|
| join | 0.81 | 0.87 |
| cross | 0.81 | 0.97 |
| Avg. | 0.81 | 0.92 |

Table 6. Experiment 6
"Accessories"

| Scene | FT | UBT |
|-------------|------|------|
| carry case | 0.78 | 0.96 |
| pull case | 0.32 | 1.0 |
| carrying | 0.86 | 1.0 |
| bag | 0.86 | 1.0 |
| skirt | 0.86 | 0.95 |
| cane | 0.88 | 1.0 |
| cart | 0.45 | 0.69 |
| Avg. | 0.72 | 0.94 |

The results are obtained by using the MOTP metric (see Section 3.2). FT is referring to the results of the feet tracker, while UBT refers to the upper-body tracker.

The main difference between both tracking methods is the motion model. The feet tracker implements a motion model, dedicated for walking movements of human feet, with eight parameters. The upper-body tracker uses a simpler linear motion model with four parameters. Therefore, it is more difficult for the feet tracker to estimate the correct motion parameters, since a larger parameter space has to be covered. Hence, the feet model is more likely to fail in case of disturbances, like occlusions in combination with other nearby objects, or changes in movement. This seems to be the main cause for the slightly better performance of the upper-body tracker.

Different movement speeds only affect the feet tracker. As shown in Table 1, the specialized motion model of the feet tracker is only able to describe regular walking movements, which are not present in the recordings of fast speed. A running movement differs from the regular and causes failures in tracking.

As Table 2 and 3 show, non-linearities and changes in movement do not pose a problem for both tracking methods. The use of particle filters implements a certain random element compensating for these variations in movement.

Since a certain sequence of observations is required to trigger a detection within the feet tracker, particle filters can only be initialized at specific time frames. The upper-body tracker can detect a person at every time step, as long

as no occlusions are present. Therefore, every person entering the scene is tracked immediately by the upper-body tracker and does not influence other particle filters. Thus, the feet tracker mainly fails in tracking multiple people, due to unfavorable conditions during initialization. The upper-body tracker primarily fails in tracking multiple people, due to identity switches, caused by occlusions. This leads to the results, shown in Table 4.

As Table 5 proves, both tracking mechanisms do not switch their targets in experiment 5. Nevertheless, identity switches may happen in more complex situations, as described in the previous paragraph.

Both tracking methods show problems with objects near the tracked persons (see Table 6). The baggage cart in experiment 6 causes scan points on both heights. It hijacks the particle filter at feet height, or causes false detections on the upper-body height. However, objects in close proximity to the person at upper-body height did not cause any problems, since their scan points were joined into the cluster of the person. Most accessories, like suitcases, are only visible at feet height. Therefore, they do not influence the upper-body tracker.

The upper-body tracker benefits from a simpler movement model and less disturbances, caused by non human objects. The feet tracker profits from less occlusions, due to the geometry of people. Another advantage of the feet tracker is the simpler installation, since no tripod or stand is required.

4 Conclusion

For the first time, a qualitative and quantitative comparison of two laser-based tracking methods was performed and their flaws and advantages were discussed in detail. In order to ensure a fair comparison, we recorded a huge dataset of laser range scans at feet and upper-body height. With a total of 55 minutes of scandata, containing 696 single recordings with more than one thousand individual person movement trajectories, we obtained more than 650 000 single laser range scans. The MOTP metric was adapted, in order to evaluate laser-based tracking results. The dataset in combination with the new metric provides a benchmarking framework, which can be used for choosing a suitable tracking mechanism for a designated operational environment.

In our scenario, the surveillance of an airport, the upper-body tracker is in advantage, since accessories like suitcases and skirts do not influence the tracking. However, a combination of scans at upper-body and feet height, like in [69], might improve the detection and tracking of people.

In our future work, we intend to offer an extended version of the dataset in combination with an evaluation framework, utilizing the introduced metric. Thus, we enable other research groups to benchmark their tracking methods for comparison. The extended dataset will include more setups, in order to widen the scope, beyond regular movements and airport scenarios. Furthermore, other tracking methods and the influence of their parameters will be evaluated.

References

1. Almeida, A., Almeida, J., Araujo, R.: Real-time tracking of moving objects using particle filters. In: ISIE, pp. 1327–1332 (2005)
2. Arras, K., Grzonka, S., Luber, M., Burgard, W.: Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities. In: ICRA, pp. 1710–1715 (2008)
3. Arras, K., Mozos, O., Burgard, W.: Using boosted features for the detection of people in 2d range data. In: ICRA, pp. 3402–3407 (2007)
4. Bernardin, K., Elbs, A., Stiefelhagen, R.: Multiple object tracking performance metrics and evaluation in a smart room environment. In: 6th IEEE International Workshop on Visual Surveillance (2006)
5. Bernardin, K., Stiefelhagen, R.: Evaluating multiple object tracking performance: The clear mot metrics. EURASIP IVP (2008)
6. Carballo, A., Ohya, A., Yuta, S.: Fusion of double layered multiple laser range finders for people detection from a mobile robot. In: MFI, pp. 677–683 (2008)
7. Comaniciu, D., Meer, P.: Distribution free decomposition of multivariate data. PAA 2(1), 22–30 (1999)
8. Glas, D.F., et al.: Simultaneous people tracking and localization for social robots using external laser range finders. In: IROS, pp. 846–853 (2009)
9. Hashimoto, M., Konda, T., Bai, Z., Takahashi, K.: Laser-based tracking of randomly moving people in crowded environments. In: ICAL, pp. 31–36 (2010)
10. Kanda, T., et al.: Who will be the customer?: A social robot that anticipates people's behavior from their trajectories. In: UbiComp, pp. 380–389 (2008)
11. Lee, J.H., Tsubouchi, T., Yamamoto, K., Egawa, S.: People tracking using a robot in motion with laser range finder. In: IROS, pp. 2936–2942 (2006)
12. Mozos, O., Kurazume, R., Hasegawa, T.: Multi-part people detection using 2d range data. IJSR 2, 31–40 (2010)
13. Schulz, D., et al.: People tracking with mobile robots using sample-based joint probabilistic data association filters. IJRR 22(2), 99–116 (2003)
14. Shao, X., et al.: Tracking a variable number of pedestrians in crowded scenes by using laser range scanners. In: SMC, pp. 1545–1551 (2008)
15. Spinello, L., Luber, M., Arras, K.O.: Tracking people in 3d using a bottom-up top-down people detector. In: ICRA, pp. 1304–1310 (2011)
16. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press, Cambridge (2005)
17. Topp, E.A., Christensen, H.I.: Tracking for following and passing persons. In: IROS, pp. 70–76 (2005)
18. Voronoi, G.: Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les paralléloèdres primitifs. Journal für die reine und angewandte Mathematik 134, 198–287 (1908)

Refinements of Restricted Higher-Order Anti-Unification for Heuristic-Driven Theory Projection

Martin Schmidt, Helmar Gust, Kai-Uwe Kühnberger, and Ulf Krumnack

University of Osnabrück, Institute of Cognitive Science,
Albrechtstr. 28, 49076 Osnabrück, Germany
`{martisch,hgust,kkuehnbe,krumnack}@uos.de`

Abstract. Empirical research supports the belief that structural commonalities between two domains are the main guidance for the construction of analogies. Restricted higher-order anti-unification has been shown suitable to find structural commonalities and generate mappings between domains in the symbolic analogy model Heuristic-Driven Theory Projection (HDTDP). This paper will describe how to enforce and integrate restrictions on mappings between symbols from a many-to-many up to a one-to-one symbol correspondence. We will also discuss how sorts together with sortal ontologies can be incorporated into anti-unification within HDTDP and thereby restrict possible mappings between domains.

1 Introduction

Analogy making is a high-level cognitive process [13] that occurs in every day life. It is considered a core component of cognition [7]. Analogies use already known information from a *source domain* to acquire knowledge in new situations in the *target domain*. They require an abstract mapping to be established between these domains. Empirical research supports the belief that structural commonalities between the two domains are the main guidance for the construction of this mapping. Many computational models, most prominently Gentner's structure mapping engine (SME) [2], have been proposed. Altogether, many different mechanisms to analyze and extract structural commonalities have been developed [5,9]. The proposed theories for analogical reasoning are mostly psychologically or neurally inspired. Heuristic-driven Theory Projection (HDTDP) [8] is however, a logic based analogy model. The domains are given as input to HDTDP in a first-order logic formalization. These formulas constitute the domain's axiomatization. Restricted higher-order anti-unification is used to capture even deep structural commonalities between formulas undetected by other frameworks. While HDTDP tries to construct cognitively plausible analogies it does not model the cognitive process of analogy making in humans.

2 Restricted Higher-Order Anti-Unification

Anti-unification, originally introduced in the context of induction, is a formal counterpart of unification. While a unifier for a given set of terms T is a term u that is an instance of every term of T , an anti-unifier is a term g that is an anti-instance of every term of T . While unification aims to find the most general unifier, anti-unification searches for the most specific anti-unifier.

1. A *renaming* $\rho^{F,F'}$ replaces a variable $F : \sigma_1 \times \dots \times \sigma_n \rightarrow \sigma \in \mathcal{V}_n$ by a variable $F' : \sigma_1 \times \dots \times \sigma_n \rightarrow \sigma \in \mathcal{V}_n$:

$$F(t_1 : \sigma_1, \dots, t_n : \sigma_n) : \sigma \xrightarrow{\rho^{F,F'}} F'(t_1 : \sigma_1, \dots, t_n : \sigma_n) : \sigma$$

2. A *fixation* ϕ_f^F replaces a variable $F : \sigma_1 \times \dots \times \sigma_n \rightarrow \sigma \in \mathcal{V}_n$ by a function symbol $f : \sigma_1 \times \dots \times \sigma_n \rightarrow \sigma \in \text{Func}_{\Sigma}$:

$$F(t_1 : \sigma_1, \dots, t_n : \sigma_n) : \sigma \xrightarrow{\phi_f^F} f(t_1 : \sigma_1, \dots, t_n : \sigma_n) : \sigma$$

3. An *argument insertion* $\iota_{G,i}^{F,F'}$ with $0 \leq i \leq n$, $F : \sigma_1 \times \dots \times \sigma_n \rightarrow \sigma \in \mathcal{V}_n$, $G : \sigma_i \times \dots \times \sigma_{i+k-1} \rightarrow \sigma_g \in \mathcal{V}_k$ with $k \leq n - i$ and $F' : \sigma_1 \times \dots \times \sigma_{i-1} \times \sigma_g \times \sigma_{i+k} \times \dots \times \sigma_n \rightarrow \sigma \in \mathcal{V}_{n-k+1}$ is defined as:

$$F(t_1 : \sigma_1, \dots, t_n : \sigma_n) : \sigma \xrightarrow{\iota_{G,i}^{F,F'}}$$

$$F'(t_1 : \sigma_1, \dots, t_{i-1} : \sigma_{i-1}, G(t_i : \sigma_i, \dots, t_{i+k-1} : \sigma_{i+k-1}) : \sigma_g, t_{i+k} : \sigma_{i+k}, \dots, t_n : \sigma_n) : \sigma$$

4. A *permutation* $\pi_{\alpha}^{F,F'}$ with $F : \sigma_1 \times \dots \times \sigma_n \rightarrow \sigma \in \mathcal{V}_n$ and $F' : \sigma_1 \times \dots \times \sigma_n \rightarrow \sigma \in \mathcal{V}_n$ together with a bijective function $\alpha : \{1, \dots, n\} \rightarrow \{1, \dots, n\}$ which is not the identity function, rearranges the arguments of a term:

$$F(t_1 : \sigma_1, \dots, t_n : \sigma_n) : \sigma \xrightarrow{\pi_{\alpha}^{F,F'}} F'(t_{\alpha(1)} : \sigma_{\alpha(1)}, \dots, t_{\alpha(n)} : \sigma_{\alpha(n)}) : \sigma$$

5. An *sort constriction* $\kappa_{G,G'}^{F,F'}$ with $F : \sigma_1 \times \dots \times \sigma_i \times \dots \times \sigma_n \rightarrow \sigma_F \in \mathcal{V}_n$, $F' : \sigma_1 \times \dots \times \sigma_{i'} \times \dots \times \sigma_n \rightarrow \sigma_F \in \mathcal{V}_n$ and $G : \sigma_{g1} \times \dots \times \sigma_{gk} \rightarrow \sigma_i \in \mathcal{V}_k$, $G' : \sigma_{g1} \times \dots \times \sigma_{gk} \rightarrow \sigma_{i'} \in \mathcal{V}_k$ where $\sigma_{i'} < \sigma_i$ is defined as:

$$F(t_1 : \sigma_1, \dots, G(t_{g1} : \sigma_{g1}, \dots, t_{gk} : \sigma_{gk}) : \sigma_i, \dots, t_n : \sigma_n) : \sigma_F \xrightarrow{\kappa_{G,G'}^{F,F'}}$$

$$F'(t_1 : \sigma_1, \dots, G'(t_{g1} : \sigma_{g1}, \dots, t_{gk} : \sigma_{gk}) : \sigma_{i'}, \dots, t_n : \sigma_n) : \sigma_F$$

Fig. 1. Basic Substitutions in restricted higher-order anti-unification

First-order anti-unification fails to detect commonalities when function symbols differ and creates a very general anti-unifier in the form of a variable. Higher-order anti-unification is therefore necessary to anti-unify complex structures, but we loose the uniqueness of the most specific anti-unifier and most specific anti-unifier are in general not well-defined. Restricted higher-order anti-unification

was proposed as a solution for computing suitable anti-unifier to build an explicit generalization during the mapping phase of the analogy framework HDTP. Similar in spirit is the approach presented in [11] for ontology repair systems.

For restricted higher-order anti-unification classical many-sorted first-order terms are extended by introducing variables that can take arguments: for every natural number n we assume an infinite set \mathcal{V}_n of variables with arity n and a finite set of n -ary function symbols \mathcal{C}_n . Here we explicitly allow the case $n = 0$ with \mathcal{V}_0 being the set of first-order variables and \mathcal{C}_0 being 0-ary function symbols (constants). Variables will be written as uppercase letters while function symbols are lowercase. $\sigma_F : F(\sigma_1 : t_1, \dots, \sigma_n : t_n)$ is then a higher order term with sort σ_F . We will omit to formalize how to handle predicate symbols explicitly since they are processed analogously to function symbols. Furthermore, we will not discuss anti-unification with sorts for now. Moreover, we do not annotate sorts in examples for ease of explanation. How sorts can be handled by restricted higher-order anti-unification is discussed later in this paper.

Given the term algebra $\text{Term}^*(\Sigma, \mathcal{V}^*)$, we define the set of basic substitutions for restricted higher-order anti-unification without sort constriction as given in figure [1.4.). The additional basic substitution for sort constriction 5. will be explained in section [6].

We write the chaining of basic substitutions $\tau_1, \tau_2, \dots, \tau_n$ as $[\tau_1, \tau_2, \dots, \tau_n]$ which has the application structure $\text{apply}(\text{apply}(\text{apply}(s, \tau_1), \dots), \tau_n)$ on a term s . Every chain of basic substitutions $[\tau_1, \tau_2, \dots, \tau_n]$ is a substitution. We will write $s \rightarrow t$ if there exists a substitution that transforms s into t , and $s \xrightarrow{\sigma} t$ if σ is such a substitution.

We omit explicit mentioning of variable arity in substitutions within examples from now on, because they are obvious from the argument-structures of the terms involved. Examples for every basic substitution within a generalization can be found in figure [2] along with the resulting terms. A generalization for a pair of terms s and t consists of two basic substitution chains τ and ν together with an anti-unifier g such that $g \xrightarrow{\tau} s$ and $g \xrightarrow{\nu} t$. Thereby, the terms s and t are both instances of the term g and g is an anti-instance for each of the terms s and t . We will call the generalization a least general generalization if the anti-unifier used is a most specific anti-unifier.

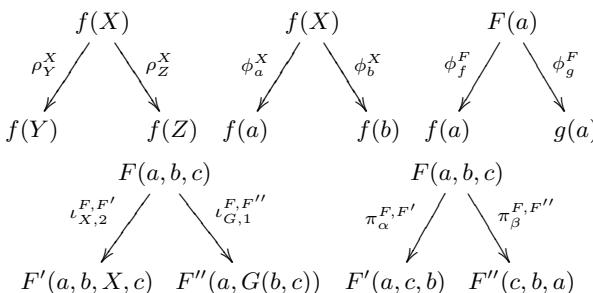


Fig. 2. Anti-Unification of terms

While a form of higher-order anti-unification, restricted higher-order anti-unification is still sufficiently restricted to be of practical use. Anti-unifiers are well defined, because anti-unifiers always exist for a pair of terms, there are finitely many of them up to renaming of variables and are always less complex than the anti-unified terms as shown in [10]. However, one drawback in comparison with first-order anti-unification is that most specific anti-unifiers are not unique. As a result, least general generalizations are also not unique anymore. An example are the terms $f(g(a, b, c), d)$ and $f(d, h(a))$ for which $f(X, Y)$, $F(d, G(a))$ and $F'(G(a), d)$ are most specific anti-unifiers. All generalizations containing these anti-unifiers are therefore least general generalizations. Note that the substitutions are not explicitly stated because there exists an infinite number of valid basic substitution chains to these anti-unifiers. This is due to the unrestricted use of the basic renaming and permutation substitution.

3 Complexity of Substitutions

Multiple least general generalizations are not necessarily disadvantageous. In analogy making several different mappings with different degrees of plausibility may coexist. Thus, we need a complexity measure for ranking generalizations and therefore have a criterion to rank alternative least general generalizations. Substitutions will need to be considered in the complexity measure because we want to promote their reuse which stands for common structure in domains when we anti-unify sets of pairs of terms.

The complexity of a basic substitution τ is defined as:

$$\mathcal{C}(\tau) = \begin{cases} 0 & \text{if } \tau = \rho \quad (\text{renaming}) \\ 1 & \text{if } \tau = \phi_f \quad (\text{fixation}) \\ k + 1 & \text{if } \tau = \iota_{V,i} \text{ and } V \in \mathcal{V}_k \quad (\text{insertion}) \\ 1 & \text{if } \tau = \pi_\alpha \quad (\text{permutation}) \end{cases}$$

For a composition of basic substitutions we define $\mathcal{C}([\tau_1, \dots, \tau_m]) = \sum_{i=1}^m \mathcal{C}(\tau_i)$.

The complexity of a substitution is meant to reflect its computational processing effort. The complexity values for basic substitutions have proven to generate plausible analogies when used by HDTTP. The complexity of a generalization can be defined in a straightforward manner by defining it as the sum of complexities of its substitutions.

In [12] it was discussed that the only situation in which renaming is needed to produce an anti-unifier is when there are two unifiable anti-instances of terms that have variables in the same argument positions. Otherwise the substitution chains become unnecessary longer. Therefore, we can take an altered view on which substitutions we prefer in generalizations. Chains of basic substitutions should only have all basic renaming substitutions before all other basic substitutions and are restricted to the minimum amount needed as discussed in [12].

Given the restrictions on the use of renaming we can now define the notion of preferred generalizations as: A preferred generalization is a least general generalization for the pair of terms such that there is no other least general

generalization that has less complexity and the two substitutions needed only contain the minimal amount of basic renaming substitutions required at their beginning. In contrast to least general generalizations there are only a finite number of preferred generalizations (up to renaming of the anti-unifier and the basic substitution chains) for a pair of terms. These preferred generalizations are a core building block for mappings between domains constructed in the mapping process of HDTP.

4 Reuse of Substitutions

Anti-unification within HDTP is used to produce a set of anti-unifiers for a set of terms of source and target domains by building generalizations. The resulting substitutions constitute the mapping between the domains. Subterms within the source and target domains can appear more than once and therefore the same substitutions might be used more than once to produce a set of generalizations. It is cognitively plausible that already used substitutions and therefore constructed mappings do not add complexity on reuse. Let g_1, g_2, \dots, g_n be generalizations constructed after each other in this order. Any basic substitution τ that is used in g_j with the same parameters as used in g_i has the complexity of 0 if $1 \leq i < j \leq n$. We could simply define reuse as above, however this will yield a form of reuse that is not desired if we do not assume additional restrictions for substitutions in generalizations. The root of the problem is that the variable F in the basic substitutions $\phi_f^F, \iota_I^{F,F'}$ and $\phi_\alpha^{F,F'}$ is not restricted in relation to the other parameters such as F' , I or α . Therefore, these basic substitutions can be used as if they did an additional basic renaming substitution.

$$\begin{aligned}
 \text{(a)} \quad F &\xrightarrow{\iota_{X,0}^{F,F'}} F'(X) \xrightarrow{\iota_{Y,1}^{F',F''}} F''(X, Y) \xrightarrow{\iota_{Z,2}^{F'',F'''}} F'''(X, Y, Z) \\
 \text{(b)} \quad G &\xrightarrow{\iota_{X,0}^{G,G'}} G'(X) \xrightarrow{\iota_{Y,1}^{G',G''}} G''(X, Y) \xrightarrow{\iota_{Z,2}^{G'',G'''}} G'''(X, Y, Z) \\
 \text{(c)} \quad G &\xrightarrow{\rho_F^G} F \xrightarrow{\iota_{X,0}^{F,F'}} F'(X) \xrightarrow{\iota_{Y,1}^{F',F''}} F''(X, Y) \xrightarrow{\iota_{Z,2}^{F'',G'''}} G'''(X, Y, Z)
 \end{aligned}$$

Fig. 3. Example for use of insertion as renaming

Figure 3 depicts three substitutions that illustrate how insertion can be used as a replacement for a basic renaming substitution. Let us assume that the basic substitutions made in (a) do not add complexity on the next reuse. Now we want to find a substitution with minimal complexity for $G \rightarrow G'''(X, Y, Z)$. One possibility is depicted in (b) and the basic substitution chain shown there has the complexity of 3, because no basic substitution from (a) is used again. The substitution $[\rho_F^G, \iota_{X,0}^{F,F'}, \iota_{Y,1}^{F',F''}, \iota_{Z,2}^{F'',G'''}]$, which corresponds to (c), reuses the basic substitutions $\iota_{X,0}^{F,F'}$ and $\iota_{Y,1}^{F',F''}$ and therefore only has complexity 1. This is certainly not the intended case of reuse as F and G share no connection here, but rather are just variables. This problem is not restricted to insertion because permutations and fixations can be used to achieve the same effect.

We observe that F'' and F''' have an explicit relation once $\iota_{Z,2}^{F'',F'''}$ is used to connect them. The basic substitution $\iota_{Z,2}^{F'',F'''}$ states that a term with variable F''' as function symbol is obtained if we insert the variable Z at position 2 into a term with variable F'' as function symbol. It is only plausible that no other variable than F'' has this connection with F''' using the parameters Z and 2 for the insertion. The basic substitution $\iota_{Z,2}^{F'',G'''}$ from figure 3(c) therefore violates this restriction. We can require analogously restrictions for the other basic substitutions.

The parameter F' in $\iota_{V,P}^{F,F'}$ and $\pi_\alpha^{F,F'}$, as well as the function symbol f in ϕ_f^F , can be inferred from the other parameters under our desired restrictions. To achieve these restrictions we introduce a function \mathcal{S} , which takes these basic substitutions with the inferable parameter F' respectively f missing and maps them to the corresponding variable F' in case of insertion and permutation and to the corresponding function symbol f in case of fixation.

There is one more property of the currently possible substitution chains that is not desirable. Whereas we have restricted that for example ϕ_s^X and ϕ_p^X can not be valid substitutions at the same time, we have not ruled out the case that ϕ_s^X and ϕ_s^Y can not be valid at the same time. This is not a problem for reuse and complexity measures as using both variants just means less possible reuse cases and thereby the same or more complexity. This in turn means this will not be the case in the preferred generalizations. $Y \xrightarrow{[\rho_X^Y, \phi_s^X]} s$ will just be preferred over $Y \xrightarrow{[\phi_s^Y]} s$ if ϕ_s^X can be reused and therefore make the longer substitution chain complexity free. Nevertheless, they are allowed and could be used in generalizations and effectively duplicate some functionality that the renaming substitution should be explicitly doing. To avoid the possibility of chains that are more complex than needed we will require \mathcal{S} to have kind of an inverse function \mathcal{S}^{-1} . We say "kind of" here because it is not an inverse function in the classical mathematical sense. The function needs to consider the type of substitution and some additional parameters to infer the original variable that was changed. For example if we have $\mathcal{S}(\phi^X) = s$ we require $\mathcal{S}^{-1}(\phi_s) = X$. Likewise the parameter F in $\iota_{V,P}^{F,F'}$ and $\pi_\alpha^{F,F'}$, as well as the function symbol F in ϕ_f^F can be inferred from the other parameters with this special inverse function. This will disallow for example $\pi_\alpha^{F,G}$ and $\pi_\alpha^{F',G}$ to be valid at the same time, because $\mathcal{S}^{-1}(\pi_\alpha^{F,G})$ can not evaluate F and F' at the same time. This does not rule out that the substitution $\pi_\beta^{F',G}$ and $\mathcal{S}^{-1}(\pi_\beta^{F,G}) = F'$ can coexist with $\mathcal{S}^{-1}(\pi_\alpha^{F,G}) = F'$.

In order to restrict the use of parameters in insertion, permutation and fixation substitutions, \mathcal{S} should have the aforementioned inverse function \mathcal{S}^{-1} . Note that a symmetric relation between F and F' in $\iota_{V,P}^{F,F'}$ and $\pi_\alpha^{F,F'}$ as well as F and f in ϕ_f^F is established now. This we might see as a one-to-one correspondence. We do not however require a relation between F and F' in a renaming substitution $\rho_F^{F'}$. The renaming substitutions will therefore not be restricted by \mathcal{S}^{-1} yet. These are

unrestricted: but we will discuss their restriction in context of mappings between symbols in the next section.

5 Restricting Mapping between Symbols

In analogy making restrictions on mapping symbols between domains are often imposed so as to disallow arbitrary mapping from symbols to symbols. The up to now described form of anti-unification allows for multiple mappings of one symbol to symbols within the other domain. However, this may not be desired as discussed in [13]. Gentner’s Structure-Mapping Theory is based on the finding that people prefer alignments that are structurally consistent [4]. This means that there should be a one-to-one correspondence between elements in source and target domain. The opposite of this is a many-to-many mapping of symbols that is possible under the up to now described anti-unification.

To achieve a more restrictive mapping we first have to impose an additional property on anti-unifiers for preferred generalizations of two formulas. Every anti-unifier used has to contain only variables that are immediately renamed in the substitution chains to the instances, which are the anti-unified formulas. This means variables in the anti-unifier can not be used in the original formulas or any anti-instances of these, possible by substitution chains, that do not contain renaming substitutions. This enforces the constraint that the generalized domains, which consists solely of anti-unifiers, do not contain variables occurring in the aligned domains. Symbols that occur and match in both terms however are not affected by this and can still occur in the anti-unifier. Note, that the number of anti-unifiers (up to renaming) in anti-unification is not constrained by this property. We therefore preserve as much structure in anti-unifiers as before and this has no effect on complexity of least general generalizations. The notion of ”containing only the minimal amount of basic renaming substitutions” for preferred generalizations however has to be interpreted now as containing the minimal amount of renaming substitutions to achieve the aforementioned new property for anti-unifiers. To be more specific each substitution chain will need exactly one renaming substitution for each variable name occurring in the anti-unifier within the generalization. Figure 4 depicts preferred generalizations for the anti-unification of $f(a, b)$ with $f(c, d)$ under the old and new extended properties for preferred generalizations for mapping domains in HDTP.

This gives us a new restriction point to regulate possible mappings between symbols. We can impose restrictions on the use of renaming substitutions in substitution chains from the variables occurring in the anti-unifiers to their instances. Anti-unification of two differing symbols yields a variable in the anti-unifier. By definition of our revised preferred generalizations each substitution chain belonging to this variable contains a renaming for this variable to a new variable, which after application of possibly more substitutions will be fixated into the corresponding symbol. We will simply restrict the number of possible different renaming substitutions from such variables in an anti-unifier to the corresponding variables in the instances and therefore also for the associated symbols.

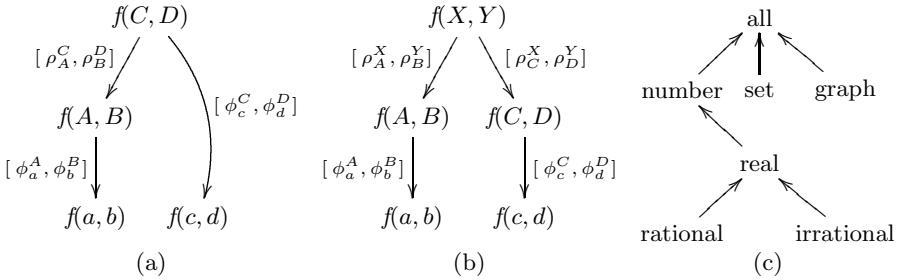


Fig. 4. Substitution chains for preferred generalization under old (a) and new (b) definition and a sortal ontology in (c)

For one-to-one mappings we require that a maximum of one renaming of the form ρ_A^V to a variable A exists in the set of substitutions of the generalizations that make up the mapping between domains. Therefore, ρ_A^W and ρ_A^Z can not coexist in the substitutions chains from the anti-unifiers to one of the input domains. To have a full one-to-one mapping constraint for alignment of domains we furthermore require that each variable in the set of generalizations only has exactly one specific renaming to a variable in each of the aligned domains. This does not mean that this specific substitution can not be used multiple times. It just guarantees that ρ_A^X and ρ_B^X can not be part of a mapping to a specific domain at the same time, which achieves the same restrictions as the usual convention to apply substitutions exhaustively but in global form. The two mentioned constraints basically achieve something similar for renaming what our functions \mathcal{S} and \mathcal{S}^{-1} already achieved for all substitutions besides renaming.

We may therefore just extend the scope of \mathcal{S} and \mathcal{S}^{-1} to act also on renaming substitutions. We have to make decisions whether a renaming is valid in a substitution chain, depending on the domain it corresponds to in a generalization. Therefore, we have to introduce a second parameter to \mathcal{S} which represents one of the two to be aligned domains of the to be anti-unified terms. Assume $f(a, b)$ belongs to the domain \mathcal{D}_α and $f(c, d)$ belongs to the domain \mathcal{D}_β in the generalizations. Every substitution τ part of the left side substitution chains to domain \mathcal{D}_α for the generalizations would be governed by the restrictions imposed by $\mathcal{S}(\tau, \mathcal{D}_\alpha)$ and every substitution ν in the substitution chains to \mathcal{D}_β by $\mathcal{S}(\nu, \mathcal{D}_\beta)$.

We can now reformulate the restrictions for renaming that lead to one-to-one mappings of symbols as properties of the function \mathcal{S} . First, by definition of a function itself it is guaranteed that $\mathcal{S}(\tau, \mathcal{D}) = X$ and $\mathcal{S}(\tau, \mathcal{D}) = Y$ with $X \neq Y$ cannot be the case at the same time for a basic renaming substitution τ . This covers one of the constraints from earlier and does not allow for multiple mappings from one variable in the general domain to multiple variables in each of the aligned domains. However, because the second parameter specifies a domain, the variable in the generalized domain can still be renamed to a different variable in the substitutions to each of the aligned domains. This is done with the renaming substitutions ρ_A^X and ρ_C^X in figure 4(b). We assume that the left hand

side substitution chain belongs to \mathcal{D}_α and the right to \mathcal{D}_β . Then $\mathcal{S}(\rho^X, \mathcal{D}_\alpha) = A$ and $\mathcal{S}(\rho^X, \mathcal{D}_\beta) = C$ can still be valid at the same time. The other constraint needed for one-to-one mappings can be encoded by requiring S to have an inverse function S^{-1} also for renaming substitutions.

We therefore have established a one-to-one mapping from each variable occurring in the generalized domain to a variable belonging to each of the aligned domains. Not restricting S to have an applicable inverse function S^{-1} for the renaming substitution can lead to many-to-many mappings. Whether a renaming substitution can be used because its parameters adhere to the properties governed by S and S^{-1} in the specific mapping mode is checked within the anti-unification algorithm [12] employed by HDTP when generating new anti-unifiers.

Because S and S^{-1} restrict only substitutions, there is a special case of mappings which they do not govern. When a symbol like a maps to the same symbol name a in the other domain no restrictions can be made. a as such will be in the anti-unifier and there will be no renaming substitutions which can conflict with a later mapping from e.g. a to b . Thereby, we might gain a non one-to-one mapping, even when adhering to the restrictions laid down by S and S^{-1} . To avoid this loophole the anti-unification algorithm will internally use variables in the anti-unifier with renaming and fixation substitutions to the symbols in the domains even if their names match. Complexities will not be assigned to the substitutions used for this construction.

Under one-to-one mapping restrictions a preferred generalization for two terms might not always exist. In the simplest case we can't anti-unify the pair of constants a and b and the pair a and c at the same time without violating one-to-one mapping restrictions. In the other extreme many-to-many mapping is not wished for because it can produce mappings that basically establish the notion that two symbols in the same domain serve the same purpose. If that would be the case, then the question is why the formalization used different names for the same concepts in the first place. What is more reasonable is that the mapping is just too overzealous, generating mappings between symbols trying to preserve as much term structure as possible. But the undesired side effect is that relations between symbols are completely intermixed. A third kind of mapping constraint on symbols namely many-to-one mapping can be established. This can be enforced by requiring that S has the partial inverse S^{-1} for renaming substitutions only for one of the two domains that can occur in the second argument. This is enforced variable by variable basis.

6 Anti-Unification with Sorts

Sorts describe the type of an entity at a general level and can be interpreted as high-level concepts e.g. *object*, *massterm*, *time* or *number*. They reflect that we do not want to consider the universe as a homogeneous collection of objects. Their purpose in HDTP is to help restrict the possible mappings during the analogical mapping process by using background knowledge about relations between classes of symbols in the involved domains.

We can assume that a strict partial order $<$ on sorts exist. Moreover, a concept that includes every symbol occurring in the domain in its extension should exist, what in natural language would be the category "thing". We will refer to this sorts as σ_{all} . Because σ_{all} is a supersort of every other sort it is the least upper bound for the set of all other sorts. A strict partial order of sorts together with a least upper bound for all sorts can be viewed as a compact and simple ontology of sorts. Given some sorts and their relations derived from the general field of mathematics, we can construct the ontology in figure 5(c). Here $\sigma_{real} < \sigma_{number}$ holds and therefore the sort *number* is a supersort of sort *real* and the sort *real* is a subsort of the sort *number*.

Given a sortal ontology which defines a supersort relation we define the additional basic substitution for restricted higher-order anti-unification with sortal ontologies as shown in figure 5(5.). It can be shown, that with this additional substitution, anti-unifiers still always exist for a pair of terms, are always less complex than the anti-unified terms and are finite up to the renaming of variables. This was the case also in the original definition of restricted higher-order anti-unification.

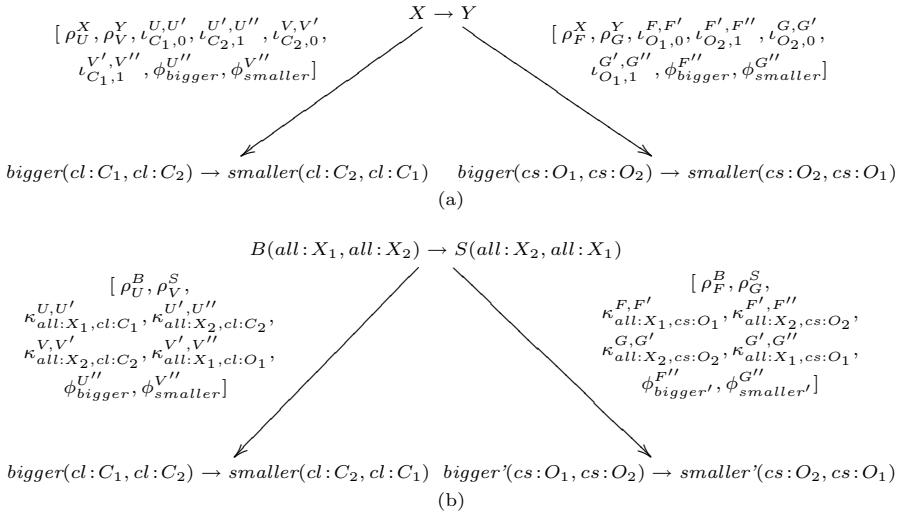


Fig. 5. Example for anti-unifier with differing sorts

Figure 5 depicts the generalization of two formulae without (a) and with the possibility of sort constriction substitutions (b). Here we assume a sortal ontology where the *all* sort is the only supersort of the *construction* (*cs*) sort from a *object construction domain* and *collection* (*cl*) sort from a *object collection domain* (see [6] for details). Term structure is lost and only a very general most specific anti-unifier $X \rightarrow Y$ is possible in example (a). In contrast, by generalization of the sorts of the arguments of the predicates *bigger* and *smaller* to the *all* sort, term structure can be preserved in (b).

All basic substitutions were given a complexity measure. Defining no complexity measure for the sort constriction substitution might have the effect of not preserving sorts to help preserve term structure during anti-unification. However, in extreme cases we may preserve sorts for minor term structure changes. For example given a equally large set of arguments for two terms that are to be anti-unified, if the signatures of these two terms match by permutation we could argue that it is more structure preserving. We would just need to permute the arguments of the terms instead of changing the sorts of all arguments in both terms to e.g. the all sort in order to gain an anti-unifier. Preferred Generalizations are defined via the complexity of substitution chains associated with an anti-unifier. In order for sorts to have an influence on the amount and term structure of preferred generalizations, we will have to define a complexity for the sort constriction substitution. We can add weights in sortal ontologies to have a complexity for sort constrictions depending on the minimal sum of weights of the vertices that need to be traversed between two sorts.

How weights in the graph of the sortal ontology are chosen will determine how sorts restrict and guide the anti-unification process, which is based on choosing generalizations with minimal complexity. As discussed earlier, we want to rather permute a set of arguments with their sorts to make them fit the signature of another function than to change the sort of each argument. Therefore, the complexity of a set of sort constrictions should be higher than that of a permutation which has the complexity of 1. On the other side we would like to prefer sort constriction instead of argument insertion to preserve as much term structure as possible. The complexity of a single sort constriction to the most general supersort σ_{all} should therefore be less than the complexity of argument insertion of a variable with arity one, which has a complexity of 1. Every sortal ontology should therefore ideally fulfill the criteria that the sort constriction from σ_{all} to any sort has less complexity than 1 while still maximizing all weights in the sortal ontology. Any sortal ontology that does not fulfill this could be normalized to fulfill this criteria by scaling down or up all weights by a constant factor. Another way to define complexity regardless of weights in the sortal ontology but adhere to principles balancing term structure and sort preservation as discussed above, is to assign each sort constriction substitution a fixed complexity of 1. However, this will result in more generalizations having the same complexity where the involved anti-unifiers only differ in the sorts used.

For efficiency reasons the current implementation of HDTP limits sortal ontologies to tree structures. These are subsets of semi-lattices and therefore all theoretical considerations are still valid. Computing the least general supersorts in a sortal ontology that has a tree structure corresponds to the well known lowest common ancestor problem and efficient algorithms to solve this problem exist. Therefore, we can conclude that an implementation of sort matching with its current limitation of sortal ontologies to trees can be viewed as being a computationally lightweight addition to HDTP's mapping process.

7 Conclusion

We showed how restricted higher-order anti-unification can be further refined to enable HDTP to enforce one-to-one up to many-to-many mapping constraints between symbols. Furthermore, it was shown how terms with sorts can be handled by anti-unification to guide the alignment of terms. A more detailed approach how weights in sortal ontologies should be assigned and what effect this has on restricted higher-order anti-unification with sortal ontologies should be the topic of further research.

References

1. Chalmers, D.J., French, R.M., Hofstadter, D.R.: High-level perception, representation, and analogy: A critique of artificial intelligence methodology. *Journal of Experimental and Theoretical Artificial Intelligence* 4(3), 185–211 (1992)
2. Falkenhainer, B., Forbus, K.D., Gentner, D.: The structure-mapping engine: Algorithm and examples. *Artificial Intelligence* 41(1), 1–63 (1989)
3. Forbus, K.D., Gentner, D., Markman, A.B., Ferguson, R.W.: Analogy just looks like high-level perception: why a domain-general approach to analogical mapping is right. *Journal of Experimental & Theoretical Artificial Intelligence* 10, 231–257 (1998)
4. Gentner, D.: Structure-mapping: A theoretical framework for analogy. *Cognitive Science* 7(2), 155–170 (1983)
5. Gentner, D.: The mechanism of analogical learning. In: Vosniadou, S., Ortony, A. (eds.) *Similarity and Analogical Reasoning*, pp. 199–241. Cambridge University Press, New York (1989)
6. Guhe, M., Pease, A., Smaill, A., Schmidt, M., Gust, H., Kühnberger, K.U., Krumnack, U.: Mathematical reasoning with higher-order anti-unification. In: Proceedings of the 32nd Annual Conference of the Cognitive Science Society, pp. 1992–1997. Cognitive Science Society, Austin (2010)
7. Gust, H., Krumnack, U., Kühnberger, K.U., Schwering, A.: Analogical reasoning: A core of cognition. *Zeitschrift für Künstliche Intelligenz (KI)*, Themenheft KI und Kognition 1, 8–12 (2008)
8. Gust, H., Kühnberger, K.U., Schmid, U.: Metaphors and heuristic-driven theory projection (HDTP). *Theoretical Computer Science* 354(1), 98–117 (2006)
9. Indurkhya, B.: Metaphor and cognition. Kluver, Dordrecht (1992)
10. Krumnack, U., Schwering, A., Gust, H., Kühnberger, K.U.: Restricted higher-order anti-unification for analogy making. In: Orgun, M.A., Thornton, J. (eds.) *AI 2007. LNCS (LNAI)*, vol. 4830, pp. 273–282. Springer, Heidelberg (2007)
11. McNeill, F., Bundy, A.: Dynamic, automatic, first-order ontology repair by diagnosis of failed plan execution. *IJSWIS* (International Journal on Semantic Web and Information Systems) special issue on Ontology Matching 3, 1–35 (2007)
12. Schmidt, M.: Restricted Higher-Order Anti-Unification for Heuristic-Driven Theory Projection, PICS (Publications of the Institute of Cognitive Science), vol. 31. University of Osnabrück, Osnabrück (2010)
13. Vosniadou, S., Ortony, A.: *Similarity and analogical reasoning*. Cambridge University Press, Cambridge (1989)

Linkless Normal Form for \mathcal{ALC} Concepts and TBoxes

Claudia Schon

University of Koblenz-Landau, Germany

Abstract. In this paper we introduce a normal form for \mathcal{ALC} concepts and TBoxes called *linkless normal form*. We investigate properties of concepts given in this normal form such as an efficient satisfiability test and the calculation of uniform interpolants. We further show a way to approximate a TBox by a concept in linkless normal form, which allows us to check certain subsumptions efficiently. This makes the linkless normal form interesting from the viewpoint of knowledge compilation. Furthermore, we show how to use the approximation of a TBox in linkless normal form to efficiently construct an approximation of a uniform interpolant of a TBox w.r.t. a given signature.

1 Introduction

Knowledge compilation is a technique originally developed for dealing with the computational intractability of propositional reasoning. It has been used in various AI systems for compiling knowledge bases offline into representations, which can be queried more efficiently. An overview of techniques for propositional knowledge bases is given in [7].

Several techniques for Description Logics, such as structural subsumption, normalization and absorption, are related to knowledge compilation. To perform a subsumption check on two concepts, structural subsumption algorithms [2] transform both concepts into a normal form and compare the structure of these normal forms. In contrast to structural subsumption, our approach is able to handle general negation. Absorption [8] and normalization [4] have the aim of increasing the performance of tableau based reasoning procedures. Unlike those approaches, we extend the use of preprocessing, allowing an efficient consistency test without requiring a tableau procedure.

With regards to Description Logics, knowledge compilation has first been investigated in [16], where \mathcal{FL} concepts are approximated by \mathcal{FL}^- -concepts. Recently, [5] introduced a normal form called *prime implicate normal form* for \mathcal{ALC} concepts which allows a polynomial subsumption check. So far, however, the prime implicate normal form has not been extended for TBoxes. Another approach to precompile both \mathcal{ALC} concepts and TBoxes is presented in [9] and [8]. There, the result of the precompilation is represented as a graph structure. Using this graph, certain subsumptions can be checked in polynomial time. However, a disadvantage of the precompilation of concepts into these graphs is that the

graph provides no possibility to see the result of the precompilation as a concept. In this paper we remedy this situation by presenting concepts as the result of the precompilation process. This clarifies the whole precompilation process and emphasizes certain properties of precompiled concepts. For example, it will be simple to develop an operator to calculate uniform interpolants of precompiled concepts w.r.t. a given signature.

In this paper we will consider the Description Logic \mathcal{ALC} [2] and adopt the notion of linkless formulas, as introduced in [4][13]. First, we present the basics of the Description Logics \mathcal{ALC} and \mathcal{ALE} . Then we define some normal forms used to introduce the idea of our precompilation. Afterwards we will discuss properties of precompiled concepts and introduce a method to efficiently check certain subsumptions using precompiled concepts.

2 Preliminaries

At first we introduce syntax and semantics of the Description Logics \mathcal{ALE} and \mathcal{ALC} [3]. Complex \mathcal{ALE} concepts C and D are formed from atomic concepts and atomic roles according to the following syntax rule:

$$C, D \rightarrow A \mid \top \mid \perp \mid \neg A \mid C \sqcap D \mid \exists R.C \mid \forall R.C$$

where A is an atomic concept and R is an atomic role. \mathcal{ALC} has the additional rules $C, D \rightarrow \neg C \mid C \sqcup D$. Next we consider the semantics of \mathcal{ALC} concepts. An interpretation \mathcal{I} is a pair $\langle \Delta^{\mathcal{I}}, .^{\mathcal{I}} \rangle$. $\Delta^{\mathcal{I}}$ is a nonempty set (the domain of the interpretation) and $.^{\mathcal{I}}$ is an interpretation function assigning to each atomic concept A a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and to each atomic role R a binary relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. We extend the interpretation function to complex concepts by the following inductive definitions:

$$\begin{aligned} \top^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\ \perp^{\mathcal{I}} &= \emptyset \\ (\neg C)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\ (C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\ (C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\ (\exists R.C)^{\mathcal{I}} &= \{a \in \Delta^{\mathcal{I}} \mid \exists b (a, b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\} \\ (\forall R.C)^{\mathcal{I}} &= \{a \in \Delta^{\mathcal{I}} \mid \forall b (a, b) \in R^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\} \end{aligned}$$

A concept C is satisfiable if there is an interpretation \mathcal{I} with $C^{\mathcal{I}} \neq \emptyset$. We call such an interpretation a model for C . A terminological axiom has the form $C \sqsubseteq D$ or $C \equiv D$ where C, D are concepts and an axiom $C \sqsubseteq D$ ($C \equiv D$) is satisfied by an interpretation \mathcal{I} if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ ($C^{\mathcal{I}} = D^{\mathcal{I}}$). A TBox consists of a finite set of terminological axioms and is called satisfiable if there is an interpretation satisfying all its axioms. Given an axiom $C \sqsubseteq D$ and a TBox \mathcal{T} we often want to know if $C \sqsubseteq D$ w.r.t. \mathcal{T} , which we denote by $C \sqsubseteq_{\mathcal{T}} D$. $C \sqsubseteq_{\mathcal{T}} D$ holds iff $C \sqsubseteq D$

is true in all models of \mathcal{T} . Another way to show that $C \sqsubseteq_{\mathcal{T}} D$ holds is to show that $(C \sqcap \neg D)^{\mathcal{I}} = \emptyset$ for every model \mathcal{I} of \mathcal{T} .

In the following, unless stated otherwise, by the term *concept*, we denote \mathcal{ALC} concepts given in negation normal form (NNF), i.e., negation occurs only in front of atomic concepts. By the term *role restriction* we denote a concept of the form $QR.C$ with $Q \in \{\exists, \forall\}$ and by *concept literal*, we denote an atomic concept or a negated atomic concept. Further by *literal* we denote a concept literal or a role restriction and \overline{C} means the complement of a concept literal C . By concepts occurring on the topmost level of a concept C , we understand each literal occurring in C , that is not in the scope of a role restriction. Further we say that two concepts C_1 and C_2 occur on the same level in concept C if they occur in the scope of the same role restrictions in C .

A concept C is in disjunctive normal form (DNF) iff C has the form $C = (\bigsqcup_{i=1}^n (\prod_{j=1}^m L_{i,j}))$ where $L_{i,j}$ is a literal and $L_{i,j} \neq L_{i,k}$ for all i, j, k , $j \neq k$. Note that this definition of DNF only affects the topmost level of a concept. Each concept can be transformed into DNF using the distributive law.

In the sequel we will analyze conjunctive paths through a concept.

Definition 1. For a concept C , the set of its paths is defined as follows:

$$\begin{aligned} \text{paths}(\perp) &= \emptyset \\ \text{paths}(\top) &= \{\emptyset\} \\ \text{paths}(C) &= \{\{C\}\}, \text{ if } C \text{ is a literal} \\ \text{paths}(C_1 \sqcup C_2) &= \text{paths}(C_1) \cup \text{paths}(C_2) \\ \text{paths}(C_1 \sqcap C_2) &= \{X \cup Y \mid X \in \text{paths}(C_1) \text{ and } Y \in \text{paths}(C_2)\} \end{aligned}$$

For example the concept: $C = (\exists R.(D \sqcup E) \sqcup \neg A) \sqcap \forall R.D \sqcap \forall R.E \sqcap B$ has two different paths $p_1 = \{\exists R.(D \sqcup E), \forall R.D, \forall R.E, B\}$ and $p_2 = \{\neg A, \forall R.D, \forall R.E, B\}$.

Definition 2. Let C be a concept. We call distinct concepts D and E conjunctively combined in C if C has a path containing both D and E or if C contains $QR.F$, $Q \in \{\exists, \forall\}$ and D and E are conjunctively combined in F .

By $|C|$ we denote the number of subconcepts occurring in C . For a TBox \mathcal{T} , $|\mathcal{T}|$ is the sum of all $|C|$ where C is the left or right hand side of an axiom in \mathcal{T} . By $\text{depth}(C)$ we denote the maximal depth of nested role restrictions occurring in C , e.g. $\text{depth}(\exists R.\forall S.(A \sqcup \exists S.B)) = 3$. The size of a concept C , denoted by $\text{size}(C)$, is the number of atomic concepts, role restrictions, negations and connectives used in C . For example the size of $A \sqcup \exists R.(B \sqcap \neg A)$ is 7. Note that the size of a concept C is in the same order of magnitude as the number of subconcepts of C .

3 Normal Forms

In the precompilation introduced in this paper, we will first precompile the topmost level of a given concept and in the next step, we will recursively perform

the precompilation on subconcepts occurring in the scope of a role restriction. In the following definition, A, B, B_1, B_2, C_1 and C_2 denote concepts.

Definition 3. A concept C is in \forall -normal form (\forall -NF) if its topmost level does not contain conjunctively combined concepts of the form $\forall R.B_1$ and $\forall R.B_2$ for the same role R . Further C is in \exists -normal form (\exists -NF) if C is in \forall -NF and each $\exists R.B$ occurring on the topmost level of C is conjunctively combined with at most one role restriction of the form $\forall R.A$. If C is in \exists -NF and for all concepts of the form $\exists R.C_1$ and $\forall R.C_2$ occurring conjunctively combined on the topmost level of C , C_1 is equivalent to $C_1 \sqcap C_2$, we say that C is in propagated \exists -NF. Furthermore C is in completely propagated \exists -NF if C is in propagated \exists -NF and for all $Q R.B$ occurring in C , $Q \in \{\exists, \forall\}$, B is in complete propagated \exists -NF as well.

Note that Def. 3 restricts occurrences of $\exists R.A$ in C . This means that for example the concept $D = (\exists R.B \sqcap \forall R.F) \sqcup (\exists R.B \sqcap \forall R.\neg E)$ is in \exists -NF, because the claimed condition holds for each occurrence of $\exists R.B$ in D .

For example the concept

$$C = \exists R.(B \sqcup E) \sqcap \forall R.\neg B \sqcap (E \sqcup D \sqcup \forall R.F)$$

is not in \forall -NF, since the two universal role restrictions $\forall R.\neg B$ and $\forall R.F$ are conjunctively combined. C can be transformed into \forall -NF. The resulting concept is:

$$\exists R.(B \sqcup E) \sqcap ((\forall R.\neg B \sqcap (E \sqcup D)) \sqcup \forall R.(\neg B \sqcap F)).$$

Transformation to \exists -NF leads to:

$$(\exists R.(B \sqcup E) \sqcap \forall R.\neg B \sqcap (E \sqcup D)) \sqcup (\exists R.(B \sqcup E) \sqcap \forall R.(\neg B \sqcap F)).$$

A completely propagated \exists -NF of concept C is:

$$(\exists R.((B \sqcup E) \sqcap \neg B) \sqcap \forall R.\neg B \sqcap (E \sqcup D)) \sqcup (\exists R.((B \sqcup E) \sqcap \neg B \sqcap F) \sqcap \forall R.(\neg B \sqcap F)) \quad (1)$$

One way to transform a concept into \forall -NF is using the idea of path dissolution [14]. Usually path dissolution is used to remove unsatisfiable paths from a propositional logic formula. In the following we give the intuition how to use path dissolution for the transformation of concepts into \forall -NF. For better understanding of the idea of path dissolution, please note that for propositional logic formulas a path is satisfiable if it does not contain complementary literals. Furthermore two propositional logic formulas are equivalent if they have the same set of satisfiable paths. Given a propositional logic formula F and a set of literals $L = \{L_1, \dots, L_n\}$, $n \geq 1$, the term *conjunctive path extension* of L in F ($CPE(L, F)$) denotes a formula in NNF whose paths are exactly those paths of F containing an element of L . Further the term *conjunctive path complement* of L in F ($CC(L, F)$) denotes a formula in NNF whose paths are exactly those paths of F containing no element of L . Neither $CPE(L, F)$ nor $CC(L, F)$ have to be in DNF. [14] gives an algorithm to compute both $CPE(L, F)$ and $CC(L, F)$ for a given formula F and a set of literals L .

Path dissolution can be also used to remove conjunctively combined universal role restrictions $\forall R.D$ and $\forall R.E$ from a concept C . For this, we use a bijection between concepts and propositional logic formulas. This bijection, called *prop*, maps each atomic concept A to a propositional logic variable a , further $\sqcap (\sqcup)$ to $\wedge (\vee)$, $\top (\perp)$ to *true (false)* and $QR.C$ to a propositional logic variable $Q_{\neg r_c}$ with $Q \in \{\exists, \forall\}$. In order to remove one occurrence of conjunctively combined universal role restrictions $\forall R.D$ and $\forall R.E$ from a concept C , we first determine the smallest subconcept $G \sqcap H$ of C modulo commutativity, which contains the conjunctive combination of $\forall R.D$ and $\forall R.E$. Modulo commutativity of \sqcap means, that there is a subconcept $D_1 \sqcap \dots \sqcap D_n$ of C with $n \geq 2$ and there are distinct D_i, D_j in $\{D_1, \dots, D_n\}$ with $G = D_i$ and $H = D_j$. W.l.o.g. we assume that $\forall R.D$ occurs in G and $\forall R.E$ occurs in H . Next we use the bijection and construct the propositional logic formula $prop(G \sqcap H) = G' \wedge H'$. We construct $CPE(\{\forall_{\neg r_d}\}, G')$, $CC(\{\forall_{\neg r_d}\}, G')$, $CPE(\{\forall_{\neg r_e}\}, H')$ and $CC(\{\forall_{\neg r_e}\}, H')$. It is obvious, that

$$\begin{aligned} G' \wedge H' \equiv & (CC(\{\forall_{\neg r_d}\}, G') \wedge CC(\{\forall_{\neg r_e}\}, H')) \\ & \vee (CC(\{\forall_{\neg r_d}\}, G') \wedge CPE(\{\forall_{\neg r_e}\}, H')) \\ & \vee (CPE(\{\forall_{\neg r_d}\}, G') \wedge CC(\{\forall_{\neg r_e}\}, H')) \\ & \vee (CPE(\{\forall_{\neg r_d}\}, G') \wedge CPE(\{\forall_{\neg r_e}\}, H')) \end{aligned} \quad (2)$$

Note that only the last disjunct of formula (2) contains conjunctively combined occurrences of $\forall_{\neg r_d}$ and $\forall_{\neg r_e}$ and further every path in the last disjunct contains both $\forall_{\neg r_d}$ and $\forall_{\neg r_e}$. Next we use the bijection to map the right side of formula (2) back to a concept N . Since every path in $(CPE(\forall_{\neg r_d}, H) \wedge CPE(\forall_{\neg r_e}, G))$ contains both $\forall_{\neg r_d}$ and $\forall_{\neg r_e}$, we can combine them to $\forall R.(D \sqcap E)$ in the concept N . Next we substitute the result of this for $G \sqcap H$ in C . After this step, the number of conjunctively combined concepts of the form $\forall R.C_1$ and $\forall R.C_2$ in C decreased by one.

In this way, all conjunctively combined concepts of the form $\forall R.C_1$ and $\forall R.C_2$ in C can be removed step by step, leading to a concept in \forall -NF. Note that the result of this transformation does not necessarily have to be in DNF. Only in the worst case, the result is in DNF which means an exponential blowup occurred. In [14] many optimizations are introduced which help to keep the result of dissolution as succinct as possible.

In a similar way, we can use dissolution to transform a concept into \exists -NF.

The \forall -normal form and propagated \exists -normal form introduced here are closely related to the normalization rules used in [1] to compute the least common subsumer of \mathcal{AEL} concept descriptions. Another related approach is the normal form used for the calculation of uniform interpolants in [19]. The complete propagated \exists -NF is closely related to the notion of standard formula of degree d in multi-modal logic introduced in [12]. However only in the worst case, the blowup produced by transforming a concept into complete propagated \exists -NF corresponds to the blowup produced to transform a formula into a standard formula of degree d or into the normal form used in [19]. The reason for that is the fact, that concepts given in complete propagated \exists -NF are allowed to have a NNF structure

and are not supposed to be transformed to DNF. In contrast to that, the normal form used in [19] is based on the DNF.

Now we are able to introduce the linkless normal form.

4 Linkless Concepts

The core of our precompilation technique is the removal of so called links [14]. Intuitively a link is a contradictory part of a concept, which can be removed from the concept preserving equivalence.

Definition 4. *For a given concept C a link is a set of two complementary concept literals occurring in a path of C . A concept C is called top-level linkless if there is no path in C containing a link.*

The idea of links was first introduced for propositional logic formulas. If a formula contains a link, this means that the formula has a contradictory path. Further if all paths of a formula contain a link, the formula is unsatisfiable. The special structure of linkless formulas in propositional logic allows us to decide satisfiability in constant time and it is possible to enumerate models very efficiently. We can remove links from a formula with the help of *path dissolution* [14] by eliminating paths containing a link. The result of removing all links from a propositional logic formula F is called *full dissolvent* of F . Further path dissolution simplifies away all occurrences of *true* and *false* in a formula. In the worst case, the removal of links can cause an exponential blowup. Path dissolution can be used for Description Logics as well. We use the bijection between concepts and propositional logic formulas introduced in Section 3.

Definition 5. *Let C be a concept mapped to $\text{prop}(C)$. Then $\text{fulldissolvent}(C)$ is the concept obtained by mapping the full dissolvent of $\text{prop}(C)$ back to a concept using prop^{-1} .*

From the fact, that path dissolution preserves equivalence in the propositional case [14] it follows, that $\text{fulldissolvent}(C) \equiv C$. Note that if $\text{prop}(C)$ is unsatisfiable, $\text{fulldissolvent}(C) = \perp$. In general, a path p is inconsistent if the conjunction of its elements is inconsistent. For a concept given in propagated \exists -NF, a path p is inconsistent iff p contains a link or p contains $\exists R.A$ with an inconsistent concept A . Our aim is now to develop a normal form for concepts, which has the same nice properties as the linkless normal form known for propositional logic formulas. The idea of this normal form is to remove links from a concept not only from the topmost level of the concept but from *all levels* of the concept. For our precompilation we assume that the input concept is in propagated \exists -NF and in the first step of our precompilation, we remove all links from the concept. The concept resulting from this step can still be inconsistent. Take $\exists R.(\neg B \sqcap B) \sqcap \forall R.B$ as an example. Therefore, in the second step of the precompilation we precompile all subconcepts occurring in the scope of an existential role restriction. Further we precompile all subconcepts occurring in the scope of an universal role restriction. This last step is necessary when we want to

check subsumptions. Checking subsumptions can introduce new existential role restrictions we need to be able to combine with universal role restrictions occurring in the precompiled concept very efficiently. Therefore it is advantageous to have precompiled versions of subconcepts occurring in the scope of universal role restrictions.

The result of the precompilation is given in the next definition.

Definition 6. A concept C is in linkless normal form (*linkless NF*) if it is in propagated \exists -NF, top-level linkless and for all $QR.B$ occurring in C , B is in linkless NF and further C is simplified according the following simplifications:

$$\top \sqcap D \rightarrow D \quad \top \sqcup D \rightarrow \top \quad \perp \sqcap D \rightarrow \perp \quad \perp \sqcup D \rightarrow D \quad \exists R.\perp \rightarrow \perp$$

A concept is given in linkless NF is also called linkless. The following algorithm calculates the linkless NF for a given concept:

Algorithm 1. Let C be a concept. The concept $\text{linkless}(C)$ can be recursively calculated as follows:

1. Transform C into propagated \exists -NF.
2. Substitute C by $\text{fulldissolvent}(C)$.
3. Simplify the result according to the simplifications given in Def. 6.
4. For all role restrictions $QR.B$ on the topmost level of C , replace B by $\text{linkless}(B)$.

Note that the precompilation, i.e. the application of algorithm 1 preserves equivalence. Therefore every concept can be transformed into an equivalent linkless NF. Like in the propositional case, the removal of links can cause an exponential blowup. The linkless NF of the example concept, which is given in propagated \exists -NF in (1) is:

$$(\exists R.(E \sqcap \neg B) \sqcap \forall R.\neg B \sqcap (E \sqcup D)) \sqcup (\exists R.(E \sqcap \neg B \sqcap F) \sqcap \forall R.(\neg B \sqcap F)) \quad (3)$$

5 Properties of Linkless Concepts

From the structure of linkless concepts follows for linkless concepts C_1 and C_2 , that the concepts $C_1 \sqcup C_2$, $\forall R.C_1$ and $\exists R.C_1$ are linkless as well. It is easy to see that linkless concepts are not closed under negation and conjunction.

The consistency of linkless concepts can be tested in constant time.

Theorem 1. A linkless concept C can only be inconsistent if $C = \perp$.

The proof of Theorem 1 can be found in [15] and uses the fact, that the simplifications given in Def. 6 are performed during the precompilation.

5.1 Tractable Subsumption Checking

In general, a subsumption $C \sqsubseteq E$ holds iff $C \sqcap \neg E$ is unsatisfiable. To simplify notation we consider subsumptions $C \sqsubseteq \neg D$, which hold iff $C \sqcap D$ is unsatisfiable. Given a linkless concept C a subsumption $C \sqsubseteq \neg D$ can be answered in time linear to $\text{size}(C) \cdot \text{size}(D)$ if D has a certain structure. The next definition specifies, for which concepts D we can check subsumptions efficiently.

Definition 7. A consistent \mathcal{ALE} concept D is called a q-concept if D is incomplete propagated \exists -NF and for all $QR.B$ occurring in D , B is consistent.

Since it is reasonable to expect concept D in a subsumption check $C \sqsubseteq \neg D$ to be rather small, a possible exponential blowup produced by the transformation of D into propagated \exists -NF is not too harmful.

Given a linkless concept C , in order to check a subsumption $C \sqsubseteq \neg D$, we have to check the satisfiability of $C \sqcap D$. However linkless concepts are not closed under conjunction. Therefore, we have to define an operator, which allows us to conjunctively combine the linkless concept C with the q-concept D resulting in a linkless concept. The operator used here is an enhancement of the conditioning operator introduced in [6] in propositional logic. Intuitively, conditioning a linkless concept C by a q-concept D means, that we assume D to be true and simplify C according to this assumption. In the following we understand a q-concept D to be the set of its conjuncts.

Definition 8. Let C be a linkless concept and D be a q-concept. Then C conditioned with D , denoted by $C|D$, is defined as:

1. If C is a concept literal:

$$C|D = \begin{cases} \top, & \text{if } C \in D \\ \perp, & \text{if } \overline{C} \in D \\ C, & \text{otherwise} \end{cases}$$

2. If C has the form $C_1 \sqcap C_2$:

$$C|D = \begin{cases} \perp, & \text{if } C_1|D = \perp \text{ or } C_2|D = \perp \\ C_i|D, & \text{if } C_j|D = \top, (i, j \in \{1, 2\}, i \neq j) \\ C_1|D \sqcap C_2|D, & \text{otherwise} \end{cases}$$

3. If C has the form $C_1 \sqcup C_2$:

$$C|D = \begin{cases} \top, & \text{if } C_1|D = \top \text{ or } C_2|D = \top \\ C_i|D, & \text{if } C_j|D = \perp, (i, j \in \{1, 2\}, i \neq j) \\ C_1|D \sqcup C_2|D, & \text{otherwise} \end{cases}$$

4. If C has the form $\forall R.E$:

$$C|D = \begin{cases} \perp, & \text{if there is } \exists R.B' \in D \text{ with } E|B' = \perp. \\ \forall R.(E|B), & \text{if there is } \forall R.B \in D \text{ and there is no } \exists R.B' \in D \\ & \text{with } E|B' = \perp. \\ \forall R.E, & \text{if there is no } \forall R.B \in D \text{ and there is no } \exists R.B' \in D \\ & \text{with } E|B' = \perp. \end{cases}$$

5. If C has the form $\exists R.E$:

$$C|D = \begin{cases} \perp, & \text{if there is } \forall R.B \in D \text{ and } E|B = \perp. \\ \exists R.(E|B), & \text{if there is } \forall R.B \in D \text{ and } E|B \neq \perp. \\ \exists R.E, & \text{otherwise} \end{cases}$$

Conditioning a concept C by a q-concept D has complexity $O(\text{size}(C) \cdot \text{size}(D))$. Note that $C|D$ is linkless.

Lemma 1. Let C be a linkless concept and D a q-concept. Then $(C|D) \sqcap D \equiv C \sqcap D$. Further $C|D$ is satisfiable iff $C \sqcap D$ is satisfiable.

Corollary 1. Let C be a linkless concept and D be a q-concept. Then it can be decided in time linear to $\text{size}(C) \cdot \text{size}(D)$ if $C \sqsubseteq \neg D$ holds.

The proof of Lemma 10 can be found in [15]. Corollary 11 is a direct consequence of Lemma 10, the fact that $C \sqsubseteq \neg D$ iff $C \sqcap D$ is unsatisfiable and the complexity of conditioning in the propositional case [6]. In (3) our example concept C is given in linkless NF. We now want to check if the subsumption $C \sqsubseteq E \sqcup \exists R.F$ holds. Negating the right side of the subsumption, leads to the q-concept $\neg E \sqcap \forall R.\neg F$. With the help of Def. 8, we can calculate: $C \sqcap \neg E \sqcap \forall R.\neg F = \exists R.(E \sqcap \neg B) \sqcap \forall R.\neg B \sqcap D$. Since this concept is satisfiable, the subsumption $C \sqsubseteq E \sqcup \exists R.F$ does not hold.

5.2 Uniform Interpolation

Another interesting transformation for precompiled theories mentioned in [7] is uniform interpolation. With regard to ontologies, uniform interpolation has many applications [12], e.g. re-use of ontologies, predicate hiding and ontology versioning. Intuitively, a uniform interpolant of a concept C w.r.t. a set of atomic concepts Φ is a concept D that does not contain any atomic concepts from Φ and is indistinguishable from C regarding the superconcepts and subsumers that do not use symbols from Φ . So the idea of uniform interpolation is to forget all symbols given in Φ without changing the meaning of C .

Definition 9. Let C be a concept and Φ a set of atomic concepts. Then the concept D is called uniform interpolant of C w.r.t. Φ , or Φ -interpolant of C for short, iff the following conditions hold:

- D contains only atomic concepts which occur in C but not in Φ .
- $\models C \sqsubseteq D$.
- For all concepts E not containing symbols from Φ it holds: $\models C \sqsubseteq E$ iff $\models D \sqsubseteq E$.

We will now present an operator to compute a uniform interpolant of a linkless concept w.r.t. a set of concept symbols.

Definition 10. Let C be a linkless concept and Φ be a set of atomic concepts. Then $UI(C, \Phi)$ is the concept obtained by substituting each occurrence of A and $\neg A$ in C by \top iff $A \in \Phi$.

The next theorem states that uniform interpolants of linkless concept w.r.t. a set of concept symbols can be calculated efficiently.

Theorem 2. Let C be a linkless concept and Φ a set of atomic concepts. Then the following hold:

1. $UI(C, \Phi)$ is a Φ -interpolant of C ,
2. $UI(C, \Phi)$ can be calculated in time linear in the size of C and
3. if $UI(C, \Phi)$ is simplified according to the simplifications given in Def. 10, then $UI(C, \Phi)$ is linkless.

The second and the third assertion follow directly from the way $UI(C, \Phi)$ is constructed. The proof of the first assertion can be found in [15].

Given for example the set $\Phi = \{E, D\}$, we can calculate the Φ -interpolant of the linkless concept C from the example given in [3]: $UI(C, \Phi) = (\exists R.(\top \sqcap \neg B) \sqcap \forall R.\neg B \sqcap (\top \sqcup \top)) \sqcup (\exists R.(\top \sqcap \neg B \sqcap F) \sqcap \forall R.(\neg B \sqcap F))$ which can be simplified according to the simplifications given in Def. 6 to the concept $(\exists R.\neg B \sqcap \forall R.\neg B) \sqcup (\exists R.(\neg B \sqcap F) \sqcap \forall R.(\neg B \sqcap F))$.

6 Linkless TBoxes

In order to extend the linkless NF for TBoxes, we first have to consider TBox approximations. We transform a TBox $\mathcal{T} = \{A_1 \sqsubseteq B_1, \dots, A_n \sqsubseteq B_n\}$ into a metaconstraint [10] by first transforming each assertion $A_i \sqsubseteq B_i$ into an equivalent assertion $\top \sqsubseteq \neg A_i \sqcup B_i$ and then conjoining all assertions. This leads to the one single assertion $\top \sqsubseteq C_{\mathcal{T}}$ with $C_{\mathcal{T}} = \prod_{A_i \sqsubseteq B_i \in \mathcal{T}} (\neg A_i \sqcup B_i)$, stating that every element in the domain has to belong to the concept $C_{\mathcal{T}}$. Now subsumptions w.r.t. the TBox \mathcal{T} can be checked using $C_{\mathcal{T}}$. If we want to know if a concept D is satisfiable w.r.t. \mathcal{T} , we can decide this by checking the satisfiability of $D \sqcap C_{\mathcal{T}} \sqcap \forall U.C_{\mathcal{T}}$. Where U is the transitive closure of the union of all roles occurring in \mathcal{T} . Since $C_{\mathcal{T}} \sqcap \forall U.C_{\mathcal{T}}$ is not an \mathcal{ALC} concept, we need an approximation of it [19]. This approximation can be transformed into linkless NF in order to get an approximation of a linkless version of the TBox.

Definition 11. For a TBox \mathcal{T} and $n \geq 0$ the n -th approximation of \mathcal{T} is defined as:

$$C_{\mathcal{T}}^{(n)} = \prod_{k=0}^n \prod_{R_1, \dots, R_k \in \mathcal{R}} \forall R_1 \dots \forall R_k.C_{\mathcal{T}}$$

with \mathcal{R} the set of all roles occurring in \mathcal{T} .

Given for example the TBox $\mathcal{T} = \{A \sqsubseteq (B \sqcup \exists R.D), B \sqsubseteq \forall R'.D\}$ we get

$$C_{\mathcal{T}}^{(0)} = C_{\mathcal{T}} = (\neg A \sqcup B \sqcup \exists R.D) \sqcap (\neg B \sqcup \forall R'.D)$$

$$C_{\mathcal{T}}^{(1)} = C_{\mathcal{T}}^{(0)} \sqcap \forall R.C_{\mathcal{T}} \sqcap \forall R'.C_{\mathcal{T}}$$

$$C_{\mathcal{T}}^{(2)} = C_{\mathcal{T}}^{(1)} \sqcap \forall R.\forall R.C_{\mathcal{T}} \sqcap \forall R.\forall R'.C_{\mathcal{T}} \sqcap \forall R'.\forall R.C_{\mathcal{T}} \sqcap \forall R'.\forall R'.C_{\mathcal{T}}$$

The next theorem follows directly from Lemma 9 in [17] and states how the approximation $C_{\mathcal{T}}^{(n)}$ of a TBox \mathcal{T} can be used to check subsumptions w.r.t. \mathcal{T} .

Theorem 3. Let \mathcal{T} be a TBox and D a concept. If $n \geq 2^{|D|+|\mathcal{T}|}$, then D is satisfiable w.r.t \mathcal{T} iff $D \sqcap C_{\mathcal{T}}^{(n)}$ is satisfiable.

Since subsumption checks can be transformed into a satisfiability test, we can use $C_{\mathcal{T}}^{(n)}$ to check subsumptions as well. Now we extend our normal form to TBoxes. The simple idea is to make $C_{\mathcal{T}}^{(n)}$ linkless for a certain number n in order to be able to check subsumptions w.r.t \mathcal{T} . We choose $n \geq 2^{|D|+|\mathcal{T}|}$. The higher we choose n the higher the size of the q-concept can be. It is reasonable to assume that q-concepts are small compared to the size of the TBox. Therefore this is rather nonrestrictive.

In the previous section we considered uniform interpolation. For TBoxes, uniform interpolation is even more interesting. In many applications, only a small subset of the signature of a given TBox is used. This leads to the idea of uniform interpolation where all atomic concepts that are not of interest are removed preserving the meaning of the original TBox within the atomic concepts of interest. However in general, uniform interpolation for \mathcal{ALC} TBoxes need not exist [19].

Definition 12. Let \mathcal{T} be a TBox and Φ a set of atomic concepts. Then the TBox \mathcal{T}' is called a uniform interpolant of \mathcal{T} w.r.t. Φ or short Φ -interpolant of \mathcal{T} iff the following conditions hold:

- \mathcal{T}' contains only atomic concepts occurring in \mathcal{T} but not in Φ .
- $\mathcal{T} \models \mathcal{T}'$.
- For all concept inclusions $C \sqsubseteq D$ not containing symbols from Φ : $\mathcal{T} \models C \sqsubseteq D$ implies $\mathcal{T}' \models C \sqsubseteq D$.

From Theorem 3 and Proposition 5.1 in [19] follows:

Corollary 2. Let \mathcal{T} be a TBox, Φ a set of atomic concepts, D a q -concept containing only atomic concepts occurring in \mathcal{T} but not in Φ and $n \geq 2^{|D|+|\mathcal{T}|}$. Then D is satisfiable w.r.t. \mathcal{T} iff D is satisfiable w.r.t. $UI(linkless(C_{\mathcal{T}}^{(n)}), \Phi)$.

We call $UI(linkless(C_{\mathcal{T}}^{(n)}), \Phi)$ n th approximation of a Φ -interpolant of \mathcal{T} . If we know the maximal size of concepts we want to test the satisfiability w.r.t. the Φ -interpolant of \mathcal{T} , we can choose n as suggested in [19] accordingly and use the n th approximation of the Φ -interpolant instead of the Φ -interpolant itself.

Note that, given a linkless version of $C_{\mathcal{T}}^{(n)}$, the UI operator introduced in the previous section can be used to calculate the approximation of the Φ -interpolant in time linear to the size of the linkless $C_{\mathcal{T}}^{(n)}$.

7 Conclusion / Future Work

This paper presents a precompilation of \mathcal{ALC} concepts and TBoxes into a normal form called linkless normal form, which allows for an efficient satisfiability test, subsumption test and uniform interpolation. In future work, we would like to extend our normal form to handle more expressive description logics. We expect that especially transitive roles will prove challenging. Another interesting point for future work is to try out other target languages known from the field of knowledge compilation for propositional logic. This could be done using the propagated \exists -NF as a basis and then transforming the result into a target language known from propositional logic. We expect a comparison of the results for different target languages to be very interesting.

References

1. Baader, F., Küsters, R., Molitor, R.: Computing least common subsumers in description logics with existential restrictions. In: Dean, T. (ed.) IJCAI, pp. 96–103. Morgan Kaufmann, San Francisco (1999)

2. Baader, F., Nutt, W.: Basic description logics. In: Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.) [3], pp. 43–95
3. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook. Cambridge University Press, Cambridge (2003)
4. Balsiger, P., Heuerding, A.: Comparison of theorem provers for modal logics - introduction and summary. In: de Swart, H. (ed.) TABLEAUX 1998. LNCS (LNAI), vol. 1397, pp. 25–26. Springer, Heidelberg (1998)
5. Bienvenu, M.: Prime implicants and prime implicants in modal logic. In: Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI 2007), pp. 397–384 (2007)
6. Darwiche, A.: Decomposable negation normal form. Journal of the ACM, 48(4) (2001)
7. Darwiche, A., Marquis, P.: A knowledge compilation map. Journal of Artificial Intelligence Research 17, 229–264 (2002)
8. Furbach, U., Günther, H., Obermaier, C.: A knowledge compilation technique for ALC TBoxes. In: Lane, C., Guesgen, H. (eds.) FLAIRS Conference. AAAI Press, Menlo Park (2009)
9. Furbach, U., Obermaier, C.: Precompiling ALC Tboxes and query answering. In: Proceedings of the 4th Workshop on Contexts and Ontologies (2008)
10. Horrocks, I.: Implementation and optimization techniques. In: Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.) [3], pp. 306–346
11. Konev, B., Walther, D., Wolter, F.: Forgetting and uniform interpolation in large-scale description logic terminologies. In: Boutilier, C. (ed.) IJCAI, pp. 830–835 (2009)
12. Kracht, M.: Tools and techniques in Modal Logic. Elsevier Science, Amsterdam (1999)
13. Murray, N., Rosenthal, E.: Tableaux, path dissolution, and decomposable negation normal form for knowledge compilation. In: Cialdea Mayer, M., Pirri, F. (eds.) TABLEAUX 2003. LNCS, vol. 2796, pp. 165–180. Springer, Heidelberg (2003)
14. Murray, N., Rosenthal, E.: Dissolution: Making paths vanish. Journal of the ACM 40(3), 504–535 (1993)
15. Schon, C.: Linkless normal form for ALC concepts. Reports of the Faculty of Informatics 12/2010, Universität Koblenz-Landau (2010),
<http://www.uni-koblenz.de/FB4/Publications/Reports>
16. Selman, B., Kautz, H.: Knowledge compilation and theory approximation. J. ACM 43(2), 193–224 (1996)
17. ten Cate, B., Conradie, W., Marx, M., Venema, Y.: Definitorially complete description logics. In: Doherty, P., Mylopoulos, J., Welty, C. (eds.) Proc. of KR 2006, pp. 79–89. AAAI Press, Menlo Park (2006)
18. Tsarkov, D., Horrocks, I.: Fact++ description logic reasoner: System description. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 292–297. Springer, Heidelberg (2006)
19. Wang, K., Wang, Z., Topor, R., Pan, J., Antoniou, G.: Concept and role forgetting in ALC ontologies. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayanan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 666–681. Springer, Heidelberg (2009)

Shape Retrieval with Qualitative Relations: The Influence of Part-Order and Approximation Precision on Retrieval Performance and Computational Effort

Arne Schuldt

Centre for Computing and Communication Technologies (TZI),
University of Bremen, Am Fallturm 1, 28359 Bremen, Germany

Abstract. Manifold approaches exist in the field of similarity-based shape retrieval. Although many of them achieve good results in reference tests, there has been less focus on systematically examining the factors influencing both retrieval performance and computational effort. Such an investigation, however, is important for the structured development and improvement of shape descriptors. This paper contributes a thorough investigation of the influence of the shape part-order and approximation precision. Firstly, two shape descriptors based on qualitative spatial relations are introduced and evaluated. These descriptors are particularly suited for the intended investigation because their only distinction is that one of them preserves the part-order, the other abandons it. Secondly, the recall and precision values are related to the degree of approximation in three-dimensional recall-precision-approximation diagrams. This helps choose an appropriate approximation precision. Finally, it turns out that remarkable retrieval results can be achieved even if only qualitative position information is considered.

1 Introduction

Similarity-based shape retrieval poses a demanding task in computer vision. 2D object shapes originate from projections of 3D objects. Latecki et al. [10] point out that, consequently, the shape of the 2D projection may change because

- the observer of the original 3D object changes his point of view,
- the original 3D object performs non-rigid motion, or
- digitisation or segmentation cause noise.

Existing shape descriptions range from compact [7, 2, 5, 14] to complex [13, 8, 11, 15]. While complex shape descriptors focus on maximising the retrieval performance, compact descriptors aim at minimising the computational effort.

Although manifold shape descriptors exist, less effort has been spent on a systematic investigation of the factors influencing both retrieval performance and computational effort. Examining such factors, however, could lead to important insights for creating both effective and efficient shape descriptions. In this context, the main research questions addressed by this paper are:

1. What is the influence of the part-order on shape retrieval?
2. What is the influence of the approximation precision on shape retrieval?

The first question is motivated by earlier experiments [15] on the influence of preserving and abandoning the order of parts (line segments in the case of polygons) in the shape representation. The second question is due to the insight that often a comparatively coarse shape abstraction suffices for object recognition [1]. As the precision may have a significant influence on the computational effort, it is crucial to determine an appropriate degree of approximation.

Obviously, one cannot expect exact quantitative answers (such as “the part-order has an influence of 15.7% on the retrieval performance”) to the above research questions that hold for all conceivable shape descriptors. However, if a significant influence can be verified for a specific shape descriptor, this may outline a clear tendency towards a general qualitative answer (such as “the part-order has a positive influence on the retrieval performance”). In particular, this paper investigates two shape descriptions that are based on cognitively motivated qualitative spatial relations. These relations characterise the relative positions of the polygon segments of the shape’s contour outline qualitatively. This leads to the following additional research question to be answered in this paper:

3. How significant is qualitative position information for shape retrieval?

The remainder of this paper is structured as follows. Section 2 introduces the two shape descriptors that serve as the subject of investigation throughout this paper. Section 3 discusses the influence of the shape precision in order to identify an appropriate degree of approximation. Section 4 presents extensive experiments that examine the influence of part-order and precision on retrieval performance and computational effort. Subsequently, Section 5 gives a discussion of the evaluation results. Finally, Section 6 summarises the findings and gives an outlook on future research.

2 Shape Description with Qualitative Spatial Relations

The first research question asks whether sticking to the order of the shape parts is advantageous or disadvantageous. To answer this question, it is important to compare two shape descriptions that only distinguish with respect to this property. Section 2.1 introduces the qualitative bipartite arrangement relations which are the foundation for the shape descriptors investigated in this paper. Subsequently, Section 2.2 introduces how part-order preserving matrices of bipartite arrangements can be applied to characterise polygons. Finally, Section 2.3 defines part-order abandoning histograms of bipartite arrangements which are a very compact shape description.

2.1 Qualitative Bipartite Arrangement Relations

Gottfried [6] proposes the positional-contrast framework of qualitative spatial relations. An important building block are the 23 bipartite arrangement relations, in short \mathcal{BA}_{23} (Figure 1). These relations describe the relative position of

two line segments with respect to each other. To this end, a double cross [17] is induced on one of the line segments, \vec{bc} in the example (Figure 1 left). It consists of three auxiliary lines that tessellate the two-dimensional plane into six sectors that can be perceived easily by humans. One auxiliary line characterises the left/right dichotomy, the other two distinguish front, during, and back. The start and end points of the other line segment \vec{ad} can then be located in any of the six sectors. If symmetries and intersections are left out, the 23 \mathcal{BA}_{23} relations remain (Figure 1 right).

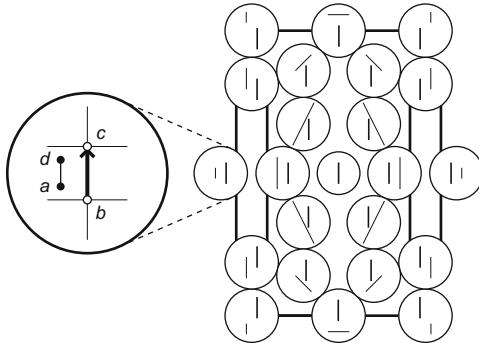


Fig. 1. The double cross induced on \vec{bc} divides the two-dimensional plane into six sectors (left). Any other line segment can start and end in any of these sectors. This leads to the 23 \mathcal{BA}_{23} relations (right).

This representation exhibits important properties with respect to the three challenges for similarity-based shape retrieval (Section 1). On the one hand, the representation is invariant against the affine transforms scale, translation, and rotation. This is accomplished by the double cross as an intrinsic reference system which is induced on one of the characterised line segments and thus exposed to the same transforms. Consequently, the representation is to a certain extent invariant against changes in the point of view and non-rigid object motion. On the other hand, compared to exact quantitative representations, the \mathcal{BA}_{23} relations have a rather coarse resolution. Even if qualitatively characterised points move (due to any of the three challenges mentioned), they will frequently not leave their double cross sector and thus not lead to another relation.

2.2 Part-Order Preserving Bipartite Arrangement Matrix

The \mathcal{BA}_{23} relations characterise the relative position of two line segments. Generally, real-world shapes are more complex than just two line segments. To this end, Gottfried's positional-contrast [6] introduces a matrix in which all line segments of a polygon are related to all others. Consequently, such matrices have a quadratic space complexity, $O(n^2)$. A row in the matrix is referred to as the *course* of the polygon. Each row characterises the qualitative positions of all

other line segments with respect to a reference segment. Each column characterises the position of the reference segment with respect to all other segments.

The off-line complexity for computing a \mathcal{BA}_{23} matrix is $O(n^2)$ as each of the n polygon segments has to be related to all n others. The distance of two matrices is the percentage of non-matching entries whereby only identical entries match. Each pair of matrices has to be compared n times due to cyclic permutation (the permutation with the minimum difference determines the distance). Hence, the complexity for on-line comparison is $O(n^3)$. Therewith, this shape description can be characterised as complex.

Note that only matrices of the same size can be compared to each other because the matching process becomes even more expensive otherwise. However, there is nothing to be said against approximating triangles with more than three points if this helps reduce the overall computational complexity.

2.3 Part-Order Abandoning Bipartite Arrangement Histogram

While the \mathcal{BA}_{23} is a complex shape descriptor that aims at maximising the retrieval performance, other applications require minimal computational effort. Think, for instance, of the real-time constraints in the RoboCup domain [3]. The complexity arises from the fact that the order with which segments occur in the polygon is preserved in the matrix. The complexity can be significantly reduced if this part-order is abandoned. This can be achieved by computing a histogram of the frequencies with which the \mathcal{BA}_{23} relations occur in the matrix. While this representation considers which relations occur, their particular order is neglected. Such a histogram has 23 entries, independently from the number of polygon segments characterised. The space complexity it is thus $O(1)$.

The off-line computational complexity for computing a histogram is still $O(n^2)$ as it requires computing the matrix first. However, the on-line complexity is significantly reduced because two \mathcal{BA}_{23} histograms can be compared with constant complexity, $O(1)$. Therewith, abandoning the part-order has actually a significant influence on the asymptotic computational complexity.

3 Appropriate Precision of Approximation

So far, this paper has implicitly assumed that shapes are characterised by polygons of their contour points. This is motivated by Attneave's finding that the contour points are of particular importance for recognition while the remaining pixels can be neglected [1]. However, even the contour points are not of equal importance. With a cat approximated by only 38 points (Figure 2), Attneave demonstrates that even comparatively few points suffice for object recognition [1]. In particular, points laying on straight lines are redundant for object perception [12], while points of high curvature are important [1].

This raises the question how important points can be identified. The discrete curve evolution approach [9] iteratively deletes the contour points with the least influence on the appearance of the shape (based on lengths and angles). However, ordering the contour points based on their importance does not answer the

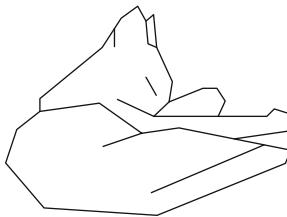


Fig. 2. Although this object has been approximated by Attneave [1] with only 38 connected points, it can still be recognised as a sleeping cat

question how many of them are actually required. Instead, McNeill and Vijayakumar [11] propose to simply choose a fixed number of arbitrary equally-spaced contour points. Proceeding this way has the following advantages:

1. The approximation can be computed quickly because the original polygon is simply split into equally-spaced points.
2. There is no need to decide which and how many points are relevant for the appearance of an individual shape.
3. Choosing a fixed number of contour points for all shapes simplifies the on-line matching problem (as assumed for the \mathcal{BA}_{23} matrix).

Although this approximation strategy relieves one from finding out how many points are necessary for an individual shape, the question remains how many points are required overall. Figure 3 depicts the contour outline of a beetle with different precision of approximation starting with a triangle. In the first row of shapes, it is virtually impossible to recognise the beetle. In the second row, the beetle becomes gradually recognisable. In the third row, the contour is sufficiently precise to identify that it is a beetle outline. The increased precision in the fourth row does not provide much more information. As a hypothesis, it can therefore be assumed that approximating this shape with 30 equally-spaced points should suffice for shape retrieval. A more general answer will be derived based on extensive retrieval experiments.

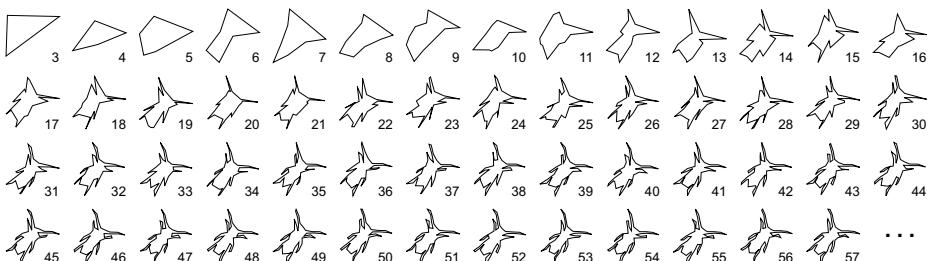


Fig. 3. The contour outline of a beetle from the MPEG-7 data set [10] approximated with increasing precision. The index denotes the number of polygon points.

4 Evaluation

As a foundation for answering the research questions posed in Section 2, extensive experiments have been conducted. The foundation for this evaluation is the shape test data set of the MPEG-7 CE-Shape-1 reference test [10]. This well-known data set comprises 1,400 shapes that are organised in 70 categories with 20 instances each. Example instances for each category are depicted in Figure 4. Section 4.1 evaluates the retrieval performance of the \mathcal{BA}_{23} histogram and the \mathcal{BA}_{23} matrix in an established reference test. Section 4.2 relates these results to the computational effort to be spent for shape retrieval with both approaches.

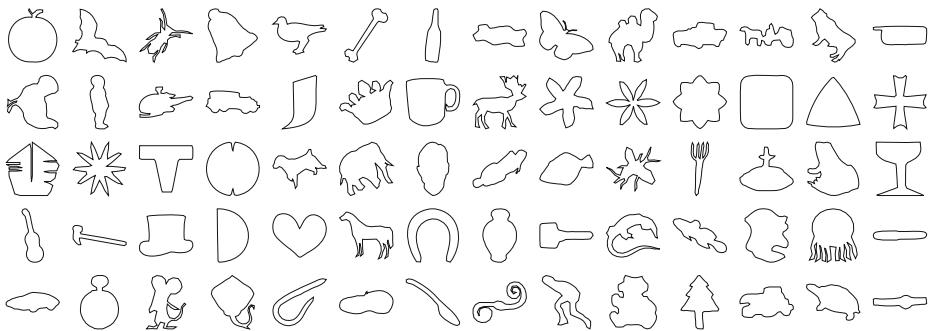


Fig. 4. Example instances for all 70 categories of the test data set for the MPEG-7 CE-Shape-1 reference test [10], each example represents 20 instances

4.1 Shape Retrieval Performance

The so-called bullseye test [10] is a well-established method for measuring the retrieval performance of shape descriptors. The approach enables the comparison of shape descriptors with completely different mathematical foundations simply based on their retrieval performance. It is usually carried out by the developers of the respective shape description themselves in order to ensure the best parameter choice. The bullseye test is defined as follows [10]. Each of the 1,400 shapes (Figure 4) is used as a query, one after another. All other shapes are then ordered based on their similarity with the query. The correct matches within the first 40 results are summed up for all queries. This number is related to $1,400 \cdot 20 = 28,000$ which is the total number of possible correct matches (each of the 70 classes has 20 instances). It is worth mentioning that the classes are grouped semantically. Therefore, retrieval results of 100.0% are unlikely solely based on shape knowledge.

The bullseye values for the \mathcal{BA}_{23} histogram in relation to the degree of approximation are depicted on the left hand side of Figure 5. In addition to the aggregated bullseye values, the three-dimensional recall-precision-approximation diagram in Figure 6 relates the recall and precision values to the degree of

approximation. Both diagrams show that, initially, the retrieval performance increases with the number of polygon points. Having reached a precision of about 30 points, however, the retrieval performance fluctuates only slightly. The best bullseye value of 41.8 % is achieved with 37 polygon points.

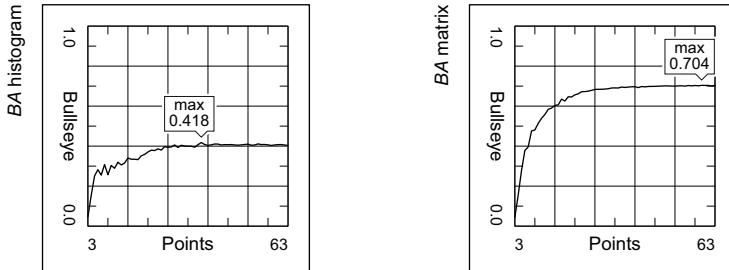


Fig. 5. Retrieval performance in the bullseye test in relation to the degree of approximation. The \mathcal{BA}_{23} histogram (without part-order) achieves a result of up to 41.8 % (left) while the \mathcal{BA}_{23} matrix (with part-order) achieves even up to 70.4 % (right).

Compared to the \mathcal{BA}_{23} histogram, the \mathcal{BA}_{23} matrix is a more complex shape representation. Hence, it is not surprising that it achieves better retrieval results. As depicted on the right hand side of Figure 5, the best bullseye value achieved is 70.4 %. The better retrieval performance also becomes clear when comparing the more detailed recall-precision-approximation diagrams for \mathcal{BA}_{23} histogram (Figure 6) and matrix (Figure 7). To recapitulate, the effort for comparing two \mathcal{BA}_{23} matrices depends on the number of polygon points. Even more important than the pure retrieval performance is therefore the degree of approximation required. Like for the \mathcal{BA}_{23} histogram, the results improve significantly until about 30 polygon points. The best result is achieved with 59 points. However, a bullseye value that is only about one percentage point below can be achieved already with 35 polygon points. For a result of only about two percentage points below the best result, 27 polygon points suffice.

4.2 Computational Effort for Shape Retrieval

To recapitulate, the foundation for the evaluation is the test data set of 1,400 shapes. This means that, for one query, the query shape has to be compared to 1,400 other shapes. As every instance serves as a query, one after another, this makes $1,400^2 = 1,960,000$ for the whole experiment. Each \mathcal{BA}_{23} histogram consists of 23 real numbers, independently from the precision of approximation. Consequently, $23 \cdot 1,400 = 32,200$ real numbers have to be compared for one query and $23 \cdot 1,960,000 = 45,080,000$ for the whole experiment (Figure 8 left). Each qualitative relation in a \mathcal{BA}_{23} matrix is represented by an integer number. This means, that for a whole experiment, $1,960,000 \cdot n^3$ integers have to be compared with n being the number of polygon points. The computational effort

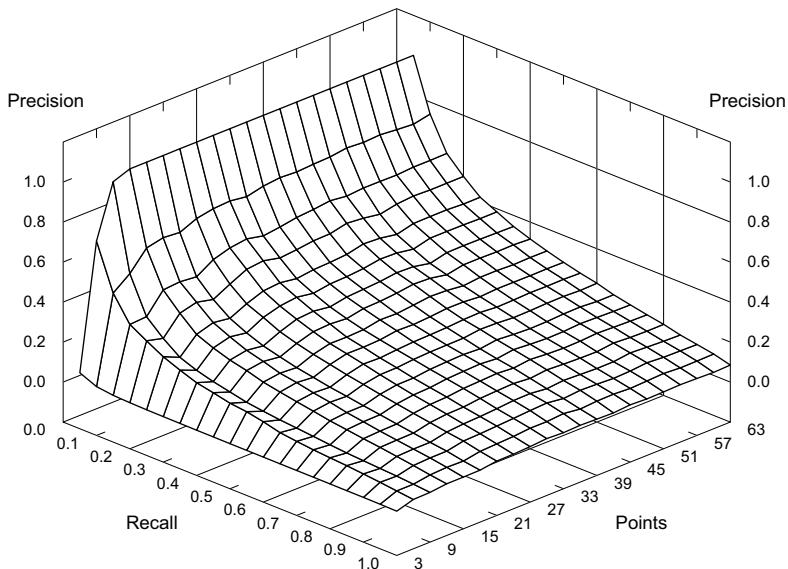


Fig. 6. Recall and precision values for the \mathcal{BA}_{23} histogram (without part-order) in relation to the approximation from 3 to 63 polygon points

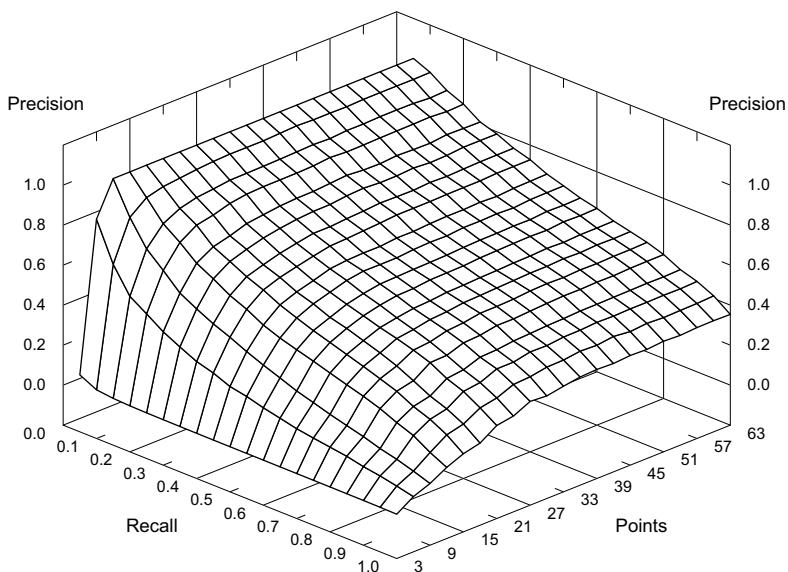


Fig. 7. Recall and precision values for the \mathcal{BA}_{23} matrix (with part-order) in relation to the approximation from 3 to 63 polygon points

ranges from 52,920,000 to 490,092,120,000 integer comparisons for polygons with 3 and 63 points, respectively (Figure 8 right). In contrast to the \mathcal{BA}_{23} histogram, the precision of approximation has thus a significant influence on the effort to be spent for comparing two \mathcal{BA}_{23} matrices.

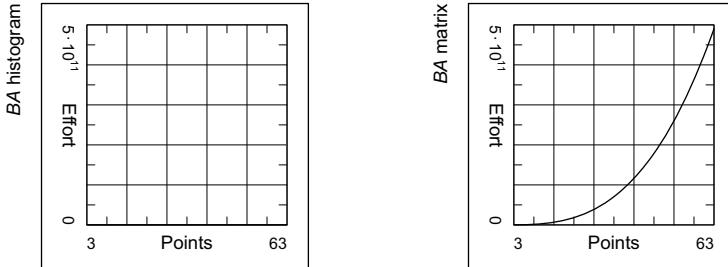


Fig. 8. The computational effort to be spent for the experiments conducted in relation to the degree of approximation. For the \mathcal{BA}_{23} histogram, the computational effort does not depend on the number of polygon points (left). The asymptotic complexity of the \mathcal{BA}_{23} matrix is $O(n^3)$ with n being the number of polygon points.

The retrieval task can be parallelised by splitting up each query into individual comparisons of two shapes. These comparisons can then be delegated to different operating system threads and processors. All experiments have been conducted on a computer with eight dual-core AMD Opteron 8218 processors with 2.6 GHz each and 64 GB RAM in total. The times for conducting the experiments (i.e., 1,960,000 comparisons each) are as follows. For the \mathcal{BA}_{23} histogram, an experiment took about 7.608 seconds. Depending on the number of polygon points, the time consumed by \mathcal{BA}_{23} matrix experiments ranges from 8.040 seconds for triangles to 544.387 seconds for polygons with 63 points. This means that an individual query (i.e., 1,400 comparisons) took about 5.434 milliseconds for the \mathcal{BA}_{23} histogram and between 5.743 and 388.848 milliseconds for \mathcal{BA}_{23} matrices. Note that this relationship does not correspond to the theoretically expected values. This can be explained by the administrative overhead (e.g., thread scheduling and result management) that has more influence for compact descriptors and becomes neglectable for more complex ones.

5 Research Questions Revisited

Based on the comprehensive evaluation, it is now possible to answer the research questions raised in Section 1. Section 5.1 addresses the influence of the shape part-order while the appropriate degree of approximation is dealt with in Section 5.2. Finally, Section 5.3 compares the retrieval performance of the qualitative descriptions to existing ones.

5.1 Influence of the Shape Part-Order

The shape part-order has a significant influence on the retrieval performance of the investigated shape descriptor. The performance of the part-order preserving \mathcal{BA}_{23} matrix in the reference test is 29 percentage points or 68% better than that of the part-order abandoning \mathcal{BA}_{23} histogram. This result is even clearer than that of an earlier experiment [15] with another pair of shape descriptions in which the improvement was only 17 percentage points or 38%. Although the exact figures are clearly not transferable to other shape descriptors, there is a strong indication for the significance of the part-order for shape recognition.

However, there is a price to be paid for preserving the part-order, namely computational effort. Comparing two \mathcal{BA}_{23} histograms is very cheap and does not depend on the number of polygon points. For the improved retrieval performance of the \mathcal{BA}_{23} matrix, considerably higher costs arise. Therefore, it is important to identify an appropriate degree of polygonal approximation.

5.2 Influence of the Approximation Precision

Also, the precision of polygonal approximation has a significant influence on the shape retrieval performance. On the one hand, it is not surprising that an approximation that too coarse (e.g., a triangle) is not meaningful enough. On the other hand, the evaluation shows that the number of required polygon points is actually limited. Initially, the retrieval performance increases with an improved approximation. The gradual increase can be explained by the fact that the individual shape classes successively reach the minimum precision required by them. From about 30 points on, there is no further significant improvement. This finding is backed by the exemplary approximation of the beetle (which is one of the more detailed classes) in Figure 3. In this example, also about 30 equally-spaced points suffice to make the beetle visually recognisable.

The limit for the approximation precision is very valuable for taming the computational effort. For the \mathcal{BA}_{23} matrix, the correlation of the computational complexity and the number of n polygon points is n^3 . Note, however, that as soon one has decided for a particular degree of approximation, the correlation of the computational complexity and the number of shapes in the search space is linear (of course, the effort for the matrix is higher than for the histogram). As explained in Section 4.2, the retrieval task can be highly parallelised. Therefore, it depends on the number of shapes to be compared and on the computer power available whether the descriptors are applicable for real-time applications, such as RoboCup vision or interactive systems.

A thorough examination of the precision-recall-approximation diagrams (Figures 6 and 7) reveals that neither the \mathcal{BA}_{23} histogram nor the matrix can retrieve the identity shape for low degrees of approximation precision (i.e., the precision value for a recall value of 0.05 is not 1.0 as one might expect). This can be explained by the fact that the qualitative position relations are not sufficiently distinctive at this coarse level of approximation.

5.3 Significance of Qualitative Position Information

Finally, the question has to be answered how the retrieval performance of the \mathcal{BA}_{23} histogram and the \mathcal{BA}_{23} matrix relate to other approaches described in the literature. This can be accomplished based on the bullseye values.

The \mathcal{BA}_{23} histogram pertains to the class of compact shape descriptions. Quantitative methods in this class are the numeric shape descriptors compactness, radius ratio, and aspect ratio from text books [25] which achieve between 16.8 and 24.1 % [14]. The Hu moments [7] which can also be computed for polygons [16] achieve bullseye values of 34.1 % [14]. The \mathcal{BA}_{23} histogram with its 41.8 % thus outperforms these quantitative methods. It is itself slightly outperformed by the qualitative scope histogram which achieves 45.5 % [14].

The \mathcal{BA}_{23} matrix clearly outperforms all compact descriptions above. As a complex shape descriptor, it has to be compared to other approaches of its class. The curvature scale space approach [13] has a bullseye value of 75.4 % [10]. The correspondence of visual parts [8] achieves 76.5 % [10]. The Procrustes distance even achieves 79.2 % [11]. Although the 70.4 % of the \mathcal{BA}_{23} matrix are slightly below those quantitative approaches, it is a remarkable result as it is achieved based only on positional-contrast. Future investigations will thoroughly examine which additional information has to be considered to improve these results.

6 Summary and Outlook

To summarise, this paper has systematically investigated influence factors for shape retrieval. In particular, it can be learnt that, on the one hand, preserving the part-order has a positive effect on the retrieval performance. On the other hand, it also requires additional computational effort which can be limited with an appropriate approximation precision. To this end, the recall-precision-approximation diagram is an important contribution because it helps identify an adequate degree of approximation. This type of diagram is introduced in this paper; usually, such extensive examinations were not made. Finally, the paper shows that remarkable retrieval results can be achieved even if only qualitative position information is incorporated. Therewith, this paper lays important foundations for the development of novel shape descriptions.

Directions for future research are threefold. The first question is whether a minimal approximation with which shapes can be recognised by humans can be identified based on the insights gained in this paper. The second question is whether more sophisticated algorithms for approximation improve the retrieval results (thereby potentially requiring more expensive matching methods). Finally, the retrieval performance of matrices and histograms of other qualitative spatial relations and distance measures has to be examined. More complex distance measures might, for instance, exploit the conceptual neighbourhood of the qualitative relations [4,6].

References

1. Attneave, F.: Some Informational Aspects of Visual Perception. *Psychological Review* 61(3), 183–193 (1954)
2. Duda, R.O., Hart, P.E.: *Pattern Classification and Scene Analysis*. John Wiley & Sons, New York (1973)
3. Fabisch, A., Laue, T., Röfer, T.: Robot Recognition and Modeling in the RoboCup Standard Platform League. In: *Humanoids 2010*, Nashville, TN, USA (2010)
4. Freksa, C.: Temporal Reasoning Based on Semi-Intervals. *Artificial Intelligence* 54(1), 199–227 (1992)
5. Garson, G.D., Biggs, R.S.: *Analytic Mapping and Geographic Databases*. Sage Publications, Newbury Park (1992)
6. Gottfried, B.: Qualitative Similarity Measures — The Case of Two-Dimensional Outlines. *Computer Vision and Image Understanding* 110(1), 117–133 (2008)
7. Hu, M.-K.: Visual Pattern Recognition by Moment Invariants. *IRE Transactions on Information Theory* 8(2), 179–187 (1962)
8. Latecki, L.J., Lakämper, R.: Shape Similarity Measure Based on Correspondence of Visual Parts. *PAMI* 22(10), 1185–1190 (2000)
9. Latecki, L.J., Lakämper, R.: Polygon Evolution by Vertex Deletion. In: Nielsen, M., Johansen, P., Fogh Olsen, O., Weickert, J. (eds.) *Scale-Space 1999*. LNCS, vol. 1682, pp. 398–409. Springer, Heidelberg (1999)
10. Latecki, L.J., Lakämper, R., Eckhardt, U.: Shape Descriptors for Non-rigid Shapes with a Single Closed Contour. In: *CVPR 2000*, pp. 424–429. IEEE Computer Society, Hilton Head Island (2000)
11. McNeill, G., Vijayakumar, S.: 2D Shape Classification and Retrieval. In: *IJCAI 2005*, pp. 1483–1488. Professional Book Center, Edinburgh (2005)
12. Mitzias, D.A., Mertzios, B.G.: Shape Recognition with a Neural Classifier Based on a Fast Polygon Approximation Technique. *Pattern Recognition* 27, 627–636 (1994)
13. Mokhtarian, F., Abbasi, S., Kittler, J.: Robust and Efficient Shape Indexing through Curvature Scale Space. In: *BMVC 1996*, pp. 53–62. British Machine Vision Association, Edinburgh (1996)
14. Schuldt, A., Gottfried, B., Herzog, O.: Retrieving Shapes Efficiently by a Qualitative Shape Descriptor: The Scope Histogram. In: Sundaram, H., Naphade, M., Smith, J.R., Rui, Y. (eds.) *CIVR 2006*. LNCS, vol. 4071, pp. 261–270. Springer, Heidelberg (2006)
15. Schuldt, A., Gottfried, B., Osterhagen, O., Herzog, O.: On the Importance of Preserving the Part-Order in Shape Retrieval. In: *SIGIR 2007*, pp. 771–772. ACM Press, Amsterdam (2007)
16. Steger, C.: On the Calculation of Arbitrary Moments of Polygons. Technical Report FGBV-96-05, Informatik IX, Technische Universität München (1996)
17. Zimmermann, K., Freksa, C.: Qualitative Spatial Reasoning Using Orientation, Distance, and Path Knowledge. *Applied Intelligence* 6, 49–58 (1996)

Classification of Semantic Concepts to Support the Analysis of the Inter-cultural Visual Repertoires of TV News Reviews

Martin Stommel, Martina Duemcke, and Otthein Herzog

TZI Center for Computing and Communication Technologies,
University Bremen, Am Fallturm 1, 28359 Bremen, Germany
mstommel,herzog@tzi.de, mduemcke@googlemail.com

Abstract. TV news reviews are of strong interest in media and communication sciences, since they indicate national and international social trends. To identify such trends, scientists from these disciplines usually work with manually annotated video data. In this paper, we investigate if the time-consuming process of manual annotation can be automated by using the current pattern recognition techniques. To this end, a comparative study on different combinations of local and global features sets with two examples of the pyramid match kernel is conducted. The performance of the classification of TV new scenes is measured. The classes are taken from a coding scheme that is the result of an international discourse in media and communication sciences. For the classification of studio vs. non-studio, football vs. ice hockey, computer graphics vs. natural scenes and crowd vs. no crowd, recognition rates between 80 and 90 percent could be achieved.¹

1 Analysis of Visual Repertoires in Media and Communication Sciences

The development of our society as documented in TV news reports is subject to research in media and communication sciences. While the contents of a news report itself is of high importance, media and communication scientists are aware of more subtle but also crucial sources of information: The structure of the scene setup may for example suggest a certain social role of the actors. The meaning of a scene also does not only depend on the video data but also on the cultural background of the viewer. And often it is more conclusive to identify issues that have been omitted compared to those actually addressed.

TV news are suited well to study such questions. The constant process of production, repetition and summarisation of TV news and news reviews results in video representations of the most relevant events of our society in very concise form [2]. The symbolic value as well as the high spread of these representations make them interesting for comparison across countries or years.

¹ A long version of this article has been published as technical report [1].



Fig. 1. Image samples from our four classes. Column 1: Studio vs. no studio. Column 2: Football vs. ice hockey. Column 3: Computer graphics vs. natural scene. Column 4: Crowd vs. no crowd.

The analysis usually includes a lot of manual video annotation. Research efforts in different countries resulted in a coding sheet that states the most important items for annotation [3]. Additional items are included to handle specific research questions. To reduce the influence of personal background and understanding, the annotation is conducted by specialists that have been trained for a high inter-coder reliability, i.e. a high agreement in the annotations. The inter-coder reliability, measured as Krippendorff's alpha, reaches an agreement of more than 70 percent, under good conditions. The annotation is used to compare the depictions of people and events over different countries or years.

In this paper, we study if the process can be facilitated by using current pattern recognition techniques. We chose four items with low symbolic connotation from the annotation scheme. The items are studio/non-studio, football/ice hockey, computer graphics/natural scenes and crowd/no crowd. Figure II shows samples of the class. The pyramid match kernel is trained to classify these items based on a set of local and global detectors and descriptors. Using the optimal feature configurations, we achieve excellent recognition rates for all classes.

2 Computational Approaches

This section briefly summarises computational approaches. For a more detailed discussion please see our technical report [1].

Computational approaches for the classification of TV material usually feature a multi-stage architecture of preprocessing, feature extraction and classification. For the case of TV material, Dorai and Venkatesh [4] distinguish between a high level that addresses the narrative form, and a low level describes formal properties of single frames or shots. Named objects or scene types at an intermediate level are often denoted as semantic concept [5]. Hauptmann et al. [6] and Garg et al. [7] provide theoretical and practical results on the number of semantic concepts and scene appearances. In most cases, semantic concepts are represented

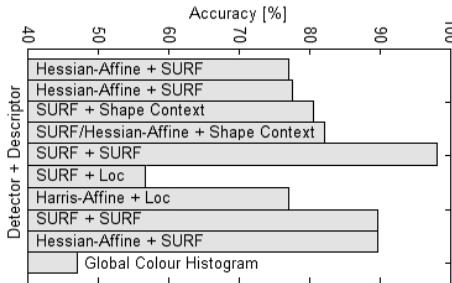


Fig. 2. Overview of the most important classification results

by sets of local feature vectors (e.g. [8]) which achieve a certain invariance against illumination and deformation. For computational reasons, bag-of-feature representations trained by machine learning algorithms [9] are very popular. Random subdivisions of the feature space [10] offer surprisingly good results at extremely low computational cost. However, experiments on different types of constellation models indicate advantages for the use of geometry [11] depending on the level of abstraction [12].

3 Experimental Results

Figure 2 shows a subset of the results for our experimental setup². The full results can be seen in our technical report [1]. For the classification of studio scenes using the original Pyramid Match Kernel, we achieve an accuracy of up to 77 per cent for the SURF descriptor in combination with MSER (first bar of the figure) or one of the corner detectors. Texture and edges therefore seem more important for the studio class than colour. Faces also appear as a good

² We evaluate two versions of Grauman and Darrell's Pyramid Match Kernel [13][14] in combination with four interest point detectors, four feature descriptors, and three global features. Local features are computed at interest points detected by Speeded Up Robust Features (SURF) [15], Maximally Stable Extremal Regions (MSER) [16], and Harris corner points obtained in the Harris-Affine or Hessian-Affine version [17]. These local detectors are combined with four feature descriptors. The descriptors are SURF [15], the image coordinate of a feature point, Steerable Filters [18] and Shape Context [19]. As global features we use global and local colour histograms. The presence or absence of faces is used as a third global feature [20]. The aim of this setup is to benefit from complementary information, e.g. colour and texture. The classification is conducted on single frames that are representatively chosen. Every frame stands for a shot in a TV news review and is annotated by the corresponding category. The sample sizes are each 200 frames for studio and no studio, each 50 frames for football, ice hockey, computer graphics, natural, each 40 frames for crowd and no crowd. The images are taken from separate shots of ARD, ZDF, ABC and CBS TV news reviews from 1999–2009. The images are randomly split into equally sized training and test samples. Special care is taken that no frames of the same video are present in the training and test set at the same time.

feature and it seems that the classifier recognises studio frames by the anchor person. However, most combinations yield only recognition rates slightly better than random.

The hierarchical clustering introduced later [14] leads to a significant improvement for almost all feature types (e.g. bar 2, 3). The best results are now obtained for feature configurations including the shape context. In the following, all results are obtained using the hierarchical clustering in the preprocessing. The combination of multiple detectors increases the accuracy to more than 81 per cent (bar 4). However, the increase in accuracy is balanced by the computational cost to handle a higher number of interest points. The combination of multiple descriptors instead of multiple detectors decreases the accuracy. The trade-off between the fusion of complementary information and numerical stability therefore still is a non-trivial problem.

The best feature combinations for the sports classification reaches an accuracy of 98 per cent (bars 5, 6). The highly dynamic scenes are handled best by the SURF detector and descriptor, while the feature location proves inappropriate here. The predominance of the either white or green field surrounded by the audience is reflected in the good results for the colour histograms.

The good contrast of the computer generated TV news shots seems to match the MSER detector combinations best with an accuracy of 72 per cent on the average. The high accuracy of 77% of the location descriptor combined with the Harris-Affine interest operator (bar 7) can be explained by the static nature of the video type. Computer animations are also frequently repeated without significant change since they form a distinguishing feature of a TV news show.

The results for the recognition of crowds are good for most local features including local colour histograms. A maximum of more than 89 per cent is reached for the SURF descriptor combined with either the SURF (bar 8) or Hessian-Affine (bar 9) interest point detector. The clear advantage over the results for the global colour histogram (bar 10) indicates that geometry is a crucial feature for this class. The face detector performs bad in the recognition of a crowd. Although many faces are present, faces are often occluded or too small to be detected. Also, the skin colour analysis might be disturbed by badly illuminated faces and faces that blur with the background.

To conclude, our experiments show that the best classifier setups achieve a high accuracy of 77% to 98% depending on the class. These results might encourage a stronger use of computer vision methods in media interpretation.

References

1. Stommel, M., Duemcke, M., Herzog, O.: Classification of Semantic Concepts to Support the Analysis of the Inter-Cultural Visual Repertoires of TV News Reviews. Technical Report 58, Center for Computing and Communication Technologies, University Bremen, Germany (2011)
2. Ludes, P.: Visual Hegemonies: An Outline = Volume 1 of The World Language of Key Visuals: Computer Sciences, Humanities, Social Sciences. LIT, Muenster (2005) (Translations into Portuguese in 2007 and Chinese in 2008)

3. Hanitzsch, T.: Codebook for Content Analysis Foreign TV News Project. Worlds of Journalisms Project (February 2010)
4. Dorai, C., Venkatesh, S.: Bridging the Semantic Gap in Content Management Systems: Computational Media Aesthetics. In: Computational Semiotics (COSIGN), pp. 94–99 (2001)
5. Smeaton, A.F., Over, P., Kraaij, W.: High level feature detection from video in TRECVID: a 5-year retrospective of achievements. In: Divakaran, A. (ed.) Multi-media Content Analysis, Theory and Applications. Springer, Heidelberg (2008)
6. Hauptmann, A., Lin, W.H., Yan, R.: How Many High-level Concepts Will Fill the Semantic Gap in News Video Retrieval? In: Proceedings of ACM International Conference on Image and Video Retrieval, pp. 627–634 (2007)
7. Garg, R., Du, H., Seitz, S.M., Snavely, N.: The Dimensionality of Scene Appearance. In: IEEE International Conference on Computer Vision, ICCV (2009)
8. Ke, Y., Sukthankar, R.: PCA-SIFT: A More Distinctive Representation for Local Image Descriptors. In: Computer Vision and Pattern Recognition (CVPR), vol. 2, pp. 506–513 (2004)
9. Jain, A.K., Duin, R., Mao, J.: Statistical Pattern Recognition: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(1), 4–37 (2000)
10. Stommel, M., Herzog, O.: Sift-based object recognition with fast alphabet creation and reduced curse of dimensionality. In: Int'l Conf. on Image and Vision Computing New Zealand, IVCNZ (2009)
11. Crandall, D.J., Felzenszwalb, P.F., Huttenlocher, D.P.: Spatial Priors for Part-Based Recognition Using Statistical Models. In: Computer Vision and Pattern Recognition (CVPR), pp. 10–17 (2005)
12. Stommel, M., Kuhnert, K.D.: Visual Alphabets on Different Levels of Abstraction for the Recognition of Deformable Objects. In: Hancock, E.R., Wilson, R.C., Windeatt, T., Ulusoy, I., Escolano, F. (eds.) SSPR&SPR 2010. LNCS, vol. 6218, pp. 213–222. Springer, Heidelberg (2010)
13. Grauman, K., Darrell, T.: The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features. In: IEEE International Conference on Computer Vision (ICCV), vol. 2, pp. 1458–1465 (2005)
14. Grauman, K., Darrell, T.: Approximate correspondences in high dimensions. In: Advances in Neural Information Processing Systems, NIPS (2006)
15. Bay, H., Ess, A., Tuytelaars, T., van Gool, L.: SURF: Speeded Up Robust Features. *Computer Vision and Image Understanding (CVIU)* 110(3), 346–359 (2006)
16. Forssen, P.E.: Maximally stable colour regions for recognition and matching. In: Computer Vision and Pattern Recognition, CVPR (2007)
17. Mikolajczyk, K., Schmid, C.: An Affine Invariant Interest Point Detector. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 128–142. Springer, Heidelberg (2002)
18. Freeman, W.H., Adelson, E.H.: The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13, 891–906 (1991)
19. Belongie, S., Mori, G., Malik, J.: Matching with shape contexts. In: IEEE Workshop on Content-based access of Image and Video-Libraries (CBAIVL), vol. 13, pp. 20–26 (2000)
20. Vezhnevets, V., Sazonov, V., Andreeva, A.: A Survey on Pixel-Based Skin Color Detection Techniques. In: Proc. Graphicon-2003, vol. 13, pp. 85–92 (2003)

Shaking Hands in Latent Space

Modeling Emotional Interactions with Gaussian Process Latent Variable Models

Nick Taubert, Dominik Endres, Andrea Christensen, and Martin A. Giese

Section for Computational Sensomotorics, Dept. of Cognitive Neurology,
University Clinic, CIN, HIH and University of Tübingen, Frondsbergstr 23,
72070 Tübingen, Germany

{nick.taubert,dominik.endres}@klinikum.uni-tuebingen.de,
{andrea.christensen,martin.giese}@uni-tuebingen.de

Abstract. We present an approach for the generative modeling of human interactions with emotional style variations. We employ a hierarchical Gaussian process latent variable model (GP-LVM) to map motion capture data of handshakes into a space of low dimensionality. The dynamics of the handshakes in this low dimensional space are then learned by a standard hidden Markov model, which also encodes the emotional style variation. To assess the quality of generated and rendered handshakes, we asked human observers to rate them for realism and emotional content. We found that generated and natural handshakes are virtually indistinguishable, proving the accuracy of the learned generative model.

1 Introduction

Accurate probabilistic models of interactive human motion are important for many applications, including computer animation, motion recognition and emotional feature analysis. Gaussian processes provide a powerful framework for the modeling of human motion since they permit to approximate complex trajectories with high accuracy, at the same time guaranteeing successful generalization from few training examples [11]. Gaussian process latent variable models (GP-LVM) have been proposed for the modeling of the motion of individual humans [3]. The resulting low-dimensional representations are suitable for feature extraction and the modeling of style. The GP-LVM can also be extended towards hierarchical architectures [4], making it possible to model the conditional dependencies induced by the coordinated movements of multiple actors / agents in an interactive setting.

The modeling of emotional styles is a classical problem in computer graphics, see e.g. [2,9,10]. The modeling of interactions between multiple characters has often been based on physical interaction models [6]. In this paper we take an approach from machine learning and try to learn the joint statistics of the interactive movements (represented as joint angles from motion capture) using a hierarchical Bayesian approach. This approach was applied to emotional handshakes between two individuals. We validated the realism of the movements of the developed statistical model by psychophysical experiments and find that the generated patterns are virtually indistinguishable from natural interactions.

2 Model

In order to learn the interactions between pairs of actors we devised a hierarchical model based on GP-LVMs with radial basis function (RBF) kernels [7] and hidden Markov models (HMM) [5]. Our model is comprised of three layers (see fig. II, left), which were learned in a layer-wise bottom-up fashion:

GP-LVM-single: the bottom layer. Observed joint angles \mathbf{y} of *one individual* actor were mapped onto a 3-dimensional latent variable \mathbf{x} . The \mathbf{y} were treated as i.i.d. across actors, trials, emotional styles and time. This approach forced the GP-LVM to learn a latent representation which captures the variation w.r.t. these variables (in particular, variation across emotional style and time).

GP-LVM-interaction: the interaction layer. For pairs of joint angles of interacting actors (say, actors 1 and 2), we computed the corresponding latent representation $(\mathbf{x}_1, \mathbf{x}_2)$ with the learned bottom layer model. This latent representation $(\mathbf{x}_1, \mathbf{x}_2)$ forms the 6-dimensional observation variable in the interaction layer, which maps $(\mathbf{x}_1, \mathbf{x}_2)$ onto a 3-dimensional latent variable \mathbf{i} . The mapping is represented by a GP-LVM. Similar to the bottom layer, the $(\mathbf{x}_1, \mathbf{x}_2)$ were treated as i.i.d. across *pairs* of actors, trials and time, sorted by emotional styles. Consequently, \mathbf{i} is a latent representation of the *interaction* which captures the variability w.r.t. emotional style and time.

Latent variables and kernel parameters were optimized with scaled conjugate gradients (SCG) [4].

HMM-dynamic: the top layer. Left-to-right HMMs (7 states, Gaussian observation models, initial mean 0 and diagonal covariance 0.3) learned the temporal evolution of \mathbf{i} , i.e. the dynamic. We trained one HMM per emotional style, across all pairs of actors and their trials.

Our model is fully generative. Since we learned one HMM per emotional style in layer **HMM-dynamic**, we can switch between styles simply by choosing the appropriate HMM and GP-LVM-interaction. We generate new interaction sequences in the latent space of GP-LVM-interaction by running the HMM forward, to compute a state-probability weighted mean from the means of the emission models. This weighted mean generates smooth input sequences in the latent space of the interaction layer. Using the learned probabilistic generative model we then back-project [7] to the joint angles at the lowest level of the hierarchy.

3 Results and Conclusion

We learned emotional handshakes represented as joint angles (in radians) derived from motion capture data. Movements were executed three times with five different emotional styles for each couple: *neutral, fearful, happy, angry* and *sad*. We fitted a commercial character model with 38 joint angles to the data from each subject, thus obtaining the training data for the model.

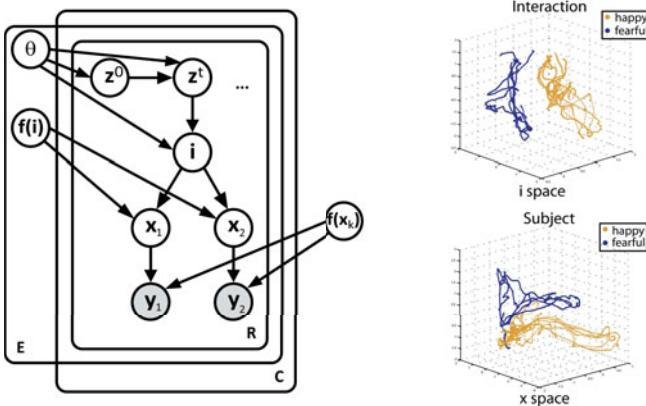


Fig. 1. *Left:* Graphical model representation. A couple C consists of two actors $\in \{1; 2\}$, which performed R trials of handshakes with emotional style E . $y_{1,2}$: observed joint angles and their latent representations $x_{1,2}$ for each actor. The $x_{1,2}$ are mapped onto the $y_{1,2}$ via a function $f(x)$ which has a Gaussian process prior. i : latent interaction representation, mapped onto the individual actors' latent variables by a function $f(i)$ which also has a Gaussian process prior. The dynamics of i are described by a HMM with hidden states z^t and parameters Θ . *Right:* Handshake trajectories in the latent spaces of layer **GP-LVM-single** (bottom panel) and layer **GP-LVM-interaction** (upper panel). The separation between emotional styles (happy and fearful) is clearly visible. For details, see section 2

To illustrate that we succeeded in learning latent representations which encode emotional style variations, see Fig. II right. The upper and lower panels show the trajectories of one actor for a fearful and a happy handshake in the latent representations of layers **GP-LVM-single** and **GP-LVM-interaction**, respectively. The two trajectories are clearly separated.

We designed a psychophysical study to test whether the accuracy of the developed probabilistic model is good enough for computer animation. Nine participants (4 female, mean age: 31 years, 7 months) took part in the first experiment. All were naïve with respect to the purpose of the study.

Rendered video clips showed two uniform, androgynous gray avatars without facial expressions to keep the focus on the bodily movements. In each trial two videos representing the same emotion were displayed side by side on a computer screen. The two videos in each trial could either both display natural handshakes, or a natural and a generated movement. Viewing time was not restricted, allowing participants to search for subtle differences between the animations. Participants classified the *emotion* of the stimulus and the *naturalism* of both displayed movie clips.

Participants failed to reliably assess naturalism, see table II top. They had a strong bias to classify every movement as being natural, which is indicated by a high hit rate (correctly identified natural movements) of 67.5%, a high false alarm rate (generated movements classified as natural) of 42.5% and a low sensitivity measure ($d' = 0.64$).

Table 1. Classification Results. *Upper part:* discrimination performance for natural versus synthesized handshake movements. Columns represent the original movement on which the animation based, rows show judgments of the participants (N=9) in percent. *Lower part:* emotion classification of natural and synthesized handshakes separately. Intended affect is shown in columns, percentages of subjects' (N=12) responses in rows. Bold entries on the diagonal mark rates of correct classification. *class. rate* overall mean correct classification rates for generated and natural movements.

| judgment | animation | | | | | | | | | |
|------------------|--------------------|--------------|-------------|--------------|--------------|------------------|------------|--------------|-------------|-------------|
| | generated movement | | | | | natural movement | | | | |
| not natural | 57.5 | | | | | 32.5 | | | | |
| natural | 42.5 | | | | | 67.5 | | | | |
| intended emotion | | | | | | | | | | |
| judgment | neutral | sad | happy | fearful | angry | neutral | sad | happy | fearful | angry |
| neutral | 70.83 | 4.17 | 12.5 | 0 | 4.17 | 91.67 | 0 | 0 | 0 | 0 |
| sad | 8.33 | 95.83 | 0 | 4.17 | 0 | 4.17 | 100 | 0 | 12.5 | 0 |
| happy | 4.17 | 0 | 87.5 | 0 | 0 | 4.17 | 0 | 91.67 | 0 | 37.5 |
| fearful | 16.67 | 0 | 0 | 91.67 | 12.5 | 0 | 0 | 0 | 87.5 | 0 |
| angry | 0 | 0 | 0 | 4.17 | 83.33 | 0 | 0 | 8.33 | 0 | 62.5 |
| class. rate | 85.83 | | | | | 86.67 | | | | |

In contrast, participants classified the expressed emotions of the handshake movements with very high accuracy. Confusions occurred mainly for emotions that are comparable in their motion energy; i.e. 'angry' and 'happy' or 'sad' and 'fearful'. These results confirm that emotions can be reliably detected from bodily movements and are in line with findings for emotional gait [SI].

Displaying animations of natural and generated handshakes side by side allows participants to directly match stimulus features. Specifically, if there are differences in the naturalism between those video clips, it would be more likely that the observer detects them compared to a sequential display of the movies. Since participants were not even able to discriminate natural movements from completely generated ones in a task where they had the opportunity of a direct comparison, it may be concluded that the generated movements look indeed very natural. On the other hand, this form of presenting the videos next to each other has the disadvantage that the affect classification might be confounded. The perception of expressed emotion could be mainly driven by one of the two videos that are simultaneously presented. To validate that the generated movements are perceived as emotional as the natural ones we conducted a second control experiment. In this experiment participants observed the identical animations as described above but now sequentially.

The classification rates of twelve participants (6 female, mean age 29 years, 9 month) are depicted in table II, bottom. Both kinds of animations conveyed enough information about the emotional context of the handshake to recognize the intended affects more than 85 % of all trials. The recognizability was highly significant for both video types, as revealed in a contingency-table analysis

testing the null hypothesis that the variables 'intended emotion' and 'perceived emotion' are independent (generated movements: $\chi^2 = 1398$, d.f. = 16, $p < 0.001$; natural movements: $\chi^2 = 1488$, d.f. = 16, $p < 0.001$). Further, the percentages of correct classification did not differ between synthesized and natural animations (paired t-test, $t_{59} = -0.163$, $p = 0.87$).

Conclusion: The results of these psychophysical experiments demonstrate clearly that the generated movements are not distinguishable by human observers from animations derived from original motion capture data. Furthermore, the fact that emotions were well identified shows that the animations were sufficient to convey very subtle information about style changes. For the future, we plan to exploit the modular architecture of our model for feature analysis and to build algorithms that automatically generate emotional interactive action sequences.

Acknowledgements. This work was supported by EU projects FP7-ICT-215866 SEARISE, FP7-249858-TP3 TANGO, FP7-ICT-248311 AMARSi and the DFG. We thank E. Huis in 't Veld for help with the movement recordings.

References

1. Atkinson, A.P., Dittrich, W., Gemmell, A., Young, A.: Emotion perception from dynamic and static body expressions in point-light and full-light displays. *Perception* (33), 717–746 (2004)
2. Brand, M., Hertzmann, A.: Style machines. In: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2000, pp. 183–192. ACM Press, New York (2000), <http://dx.doi.org/10.1145/344779.344865>
3. Lawrence, N.D.: Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of Machine Learning Research* 6, 1783–1816 (2005)
4. Lawrence, N.D., Moore, A.J.: Hierarchical gaussian process latent variable models. In: Proceedings of the 24th International Conference in Machine Learning, pp. 481–488. Omnipress (2007)
5. Li, X., Parizeau, M., Plamondon, R.: Training hidden markov models with multiple observations-a combinatorial method. *IEEE Trans. Pattern Anal. Mach. Intell.* 22(4), 371–377 (2000)
6. Nguyen, N., Wheatland, N., Brown, D., Parise, B., Liu, C.K., Zordan, V.B.: Performance capture with physical interaction. In: Symposium on Computer Animation, pp. 189–195 (2010)
7. Rasmussen, C.E., Williams, C.K.I.: Gaussian processes for machine learning. *Journal of the American Statistical Association* 103, 429 (2008)
8. Roether, C., Omlor, L., Christensen, A., Giese, M.A.: Critical features for the perception of emotion from gait. *Journal of Vision* 6, 10.1167/9.6.15 (2009)
9. Rose, C., Bodenheimer, B., Cohen, M.F.: Verbs and adverbs: Multidimensional motion interpolation using radial basis functions. *IEEE Computer Graphics and Applications* 18, 32–40 (1998)
10. Unuma, M., Anjyo, K., Takeuchi, R.: Fourier principles for emotion-based human figure animation. In: Proceedings of Computer Graphics, SIGGRAPH 1995 (1995)
11. Wang, J.M., Fleet, D.J., Member, S., Hertzmann, A.: Gaussian process dynamical models for human motion. *IEEE Trans. Pattern Anal. Machine Intell* (2007)

Value-Difference Based Exploration: Adaptive Control between Epsilon-Greedy and Softmax

Michel Tokic^{1,2} and Günther Palm¹

¹ Institute of Neural Information Processing, University of Ulm, 89069 Ulm, Germany

² Institute of Applied Research, University of Applied Sciences,
Ravensburg-Weingarten, 88241 Weingarten, Germany

Abstract. This paper proposes “Value-Difference Based Exploration combined with Softmax action selection” (VDBE-Softmax) as an adaptive exploration/exploitation policy for temporal-difference learning. The advantage of the proposed approach is that exploration actions are only selected in situations when the knowledge about the environment is uncertain, which is indicated by fluctuating values during learning. The method is evaluated in experiments having deterministic rewards and a mixture of both deterministic and stochastic rewards. The results show that a VDBE-Softmax policy can outperform ε -greedy, Softmax and VDBE policies in combination with on- and off-policy learning algorithms such as Q -learning and Sarsa. Furthermore, it is also shown that VDBE-Softmax is more reliable in case of value-function oscillations.

1 Introduction

Balancing the ratio between exploration and exploitation is one of the most challenging tasks in reinforcement learning with great impact on the agent’s learning performance. On the one hand, too much exploration prevents the agent from maximizing short-term reward because selected *exploration* actions may yield negative reward from the environment. On the other hand, *exploiting* uncertain environment knowledge prevents from maximizing long-term reward since selected actions may remain suboptimal. This problem is well known as the *dilemma of exploration and exploitation* [1].

A straightforward—and often very successful—approach is to balance exploration/exploitation by the ε -greedy method [2]. With this method, the amount of exploration is globally controlled by a parameter, ε , that determines the randomness in action selections. In contrast to others, one advantage of ε -greedy is the fact that no memorization of exploration specific data is required, such as counters [3] or confidence bounds [4, 5], which makes the method particularly interesting for very large or even continuous state-spaces. Compared to other more complex methods, ε -greedy is often hard to beat [6] and reported to be often the method of first choice as stated by Sutton [7]. In practice, however, a drawback of ε -greedy is that it is unclear which setting of ε leads to good results for a given learning problem. For this reason, the experimenter has to rigorously

hand tune ε for obtaining good results, which can be a very time-consuming task in practice depending on the complexity of the target application.

One method that aims at overcoming the above mentioned limitation of ε -greedy is “Value-Difference Based Exploration” (VDBE) [8]. In contrast to pure ε -greedy, VDBE adapts a state-dependent exploration-probability, $\varepsilon(s)$, based on fluctuations in the temporal-difference error instead of requiring to tune a global parameter by hand. However, since the original article on VDBE demonstrated the method on a multi-armed bandit task [9], results from applying the method in multi-state MDPs are still due. For this reason, open questions are: (1) is the method also able to outperform other basic exploration strategies in multi-state MDPs and (2) how do *on-* and *off-policy* learning methods affect learning performance?

This paper gives answers to these questions: Results are reported on evaluating ε -greedy, Softmax and VDBE policies on two different examples. In this context, it is shown that value-function oscillations (e.g. caused by function approximation or by learning algorithms such as Sarsa) can lead to a constant level of exploration when using VDBE and thus to bad learning performance. For this reason, an extension of VDBE to the so-called *VDBE-Softmax* method is proposed that extends Wierings’ *Max-Boltzmann Exploration* rule [10] in an adaptive manner. In Section 4, all four policies (ε -greedy, Softmax, VDBE and VDBE-Softmax) are evaluated on the cliff-walking problem [1] using deterministic rewards. In Section 5, all four policies are evaluated in the here presented bandit-world task having both deterministic and stochastic rewards. The results show that VDBE and VDBE-Softmax policies are able to outperform ε -greedy- and Softmax policies under the condition of using Q -learning and constant values for the exploration parameter. Finally, we show that VDBE diverges in case of oscillations in the value function, but in turn converges to near optimal results when the proposed VDBE-Softmax method is used.

2 Methodology

The reinforcement learning (RL) framework is considered where an agent interacts with a Markovian decision process (MDP) [1]. At each discrete time step $t \in \{0, 1, 2, \dots\}$ the agent is in a certain state $s_t \in \mathcal{S}$. After the selection of an action, $a_t \in \mathcal{A}(s_t)$, the agent receives a reward signal from the environment, $r_{t+1} \in \mathbb{R}$, and passes into a successor state s' . The decision which action is chosen in a certain state is characterized by a policy $\pi(s) = a$, which could also be stochastic $\pi(a|s) = Pr\{a_t = a|s_t = s\}$. A policy that maximizes the cumulative reward is denoted as π^* .

In reinforcement learning, one way of learning policies is learning a value-function that denotes how “valuable” it is to select action a in state s . Here, a state-action value, $Q(s, a)$, denotes the expected discounted reward for following π when starting in state s and selecting action a :

$$Q^\pi(s, a) = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a \right\}, \quad (1)$$

where γ is a discount factor such that $0 < \gamma \leq 1$ for episodic learning tasks and $0 < \gamma < 1$ for continuous learning tasks.

2.1 Learning the Q Function by *On-* and *Off-Policy* Methods

Value functions are learned by sampling observations of the interaction between the agent and its environment. For this, the branch of *temporal-difference learning* offers two commonly used algorithms which are namely Sarsa for on-policy control [11]:

$$\begin{aligned}\Delta_{\text{Sarsa}} &\leftarrow [r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \\ Q(s_t, a_t) &\leftarrow Q(s_t, a_t) + \alpha \Delta_{\text{Sarsa}} ,\end{aligned}\quad (2)$$

and Q -learning for off-policy control [2]:

$$\begin{aligned}b^* &\leftarrow \operatorname{argmax}_{b \in \mathcal{A}(s_{t+1})} Q(s_{t+1}, b) \\ \Delta_{\text{Qlearning}} &\leftarrow [r_{t+1} + \gamma Q(s_{t+1}, b^*) - Q(s_t, a_t)] \\ Q(s_t, a_t) &\leftarrow Q(s_t, a_t) + \alpha \Delta_{\text{Qlearning}} ,\end{aligned}\quad (3)$$

where α is a stepsize parameter [12]. The only technical difference between both algorithms is the inclusion of successor-state information used for the evaluation of action a_t taken in state s_t while learning the value function. Sarsa includes the discounted value of the selected action in the successor state, $Q(s_{t+1}, a_{t+1})$, for which reason it is called to be an *on-policy* method. In contrast, Q -learning includes the discounted value of the optimal action in the successor state, $Q(s_{t+1}, b^*)$, for which reason it is called to be an *off-policy* method.

Although the convergence of Sarsa depends on the stochasticity in action selections, it is well known that the algorithm outperforms Q -learning in many cases even though no convergence proof exists for Sarsa. However, if the stochasticity in action selections becomes zero (i.e. greedy), Sarsa technically becomes the same as Q -learning, and thus also convergent under several conditions [13].

2.2 Basic Exploration/Exploitation Strategies

Two widely used methods for balancing exploration/exploitation are ε -greedy and Softmax [1]. With ε -greedy, at each time step, the agent selects a random action with a fixed probability, $0 \leq \varepsilon \leq 1$, instead of selecting greedily one of the learned optimal actions with respect to the Q -function:

$$\pi(s) = \begin{cases} \text{random action from } \mathcal{A}(s) & \text{if } \xi < \varepsilon \\ \operatorname{argmax}_{a \in \mathcal{A}(s)} Q(s, a) & \text{otherwise,} \end{cases} \quad (4)$$

where $0 \leq \xi \leq 1$ is a uniform random number drawn at each time step. In contrast, Softmax utilizes action-selection probabilities which are determined by ranking the value-function estimates using a Boltzmann distribution:

$$\pi(a|s) = Pr\{a_t = a | s_t = s\} = \frac{e^{\frac{Q(s,a)}{\tau}}}{\sum_b e^{\frac{Q(s,b)}{\tau}}} , \quad (5)$$

where τ is a positive parameter called temperature. High temperatures cause all actions to be nearly equiprobable, whereas low temperatures cause greedy action selections.

In practice, both methods have advantages and disadvantages as described in [1]. In the literature, both policies have been reported as methods for describing the action-selection process in the human brain, where Softmax seems to be the better fit of both [14].

2.3 Value-Difference Based Exploration

In order to control the agent's action-selection policy, the basic idea of "Value-Difference Based Exploration" (VDBE) is to extend the ε -greedy method by introducing a state-dependent exploration probability, $\varepsilon(s)$, instead of hand-tuning a global parameter [8]. The desired behavior is to have the agent being more explorative in situations when the knowledge about the environment is uncertain, e.g. at the beginning of the learning process, which is indicated by fluctuating values during learning. On the other hand, the amount of exploration should be reduced as far as the agent's knowledge becomes certain, which is indicated by very small or no value differences. Such an adaptive behavior is obtained by computing after each learning step a state-dependent exploration probability, $\varepsilon(s)$, according to the difference in a Boltzmann distribution of the value before and after learning:

$$f(s, a, \sigma) = \left| \frac{e^{\frac{Q_t(s, a)}{\sigma}}}{e^{\frac{Q_t(s, a)}{\sigma}} + e^{\frac{Q_{t+1}(s, a)}{\sigma}}} - \frac{e^{\frac{Q_{t+1}(s, a)}{\sigma}}}{e^{\frac{Q_t(s, a)}{\sigma}} + e^{\frac{Q_{t+1}(s, a)}{\sigma}}} \right| = \frac{1 - e^{\frac{-|Q_{t+1}(s, a) - Q_t(s, a)|}{\sigma}}}{1 + e^{\frac{-|Q_{t+1}(s, a) - Q_t(s, a)|}{\sigma}}} = \frac{1 - e^{\frac{-|\alpha \cdot \Delta|}{\sigma}}}{1 + e^{\frac{-|\alpha \cdot \Delta|}{\sigma}}} \quad (6)$$

$$\varepsilon_{t+1}(s) = \delta \cdot f(s_t, a_t, \sigma) + (1 - \delta) \cdot \varepsilon_t(s), \quad (7)$$

where σ is a positive constant called *inverse sensitivity* and $\delta \in [0, 1]$ a parameter determining the influence of the selected action on the state-dependent exploration probability. A reasonable setting for δ is the inverse of the number of actions in the current state, $\delta(s) = \frac{1}{|\mathcal{A}(s)|}$, since all actions should contribute equally to $\varepsilon(s)$, and which always led to good results in our experiments. At the beginning of the learning process, all exploration probabilities are initialized arbitrary, e.g. $\varepsilon_{t=0}(s) = 1$ for all states. The parameter σ influences $\varepsilon(s)$ in a way that low values cause full exploration at small value changes. On the other hand, high values of σ cause a high level of exploration only at large value changes. Finally, the exploration probability approaches zero as far as the Q -function converges which results to pure greedy action selections.

3 VDBE-Softmax

Although VDBE has successfully been applied in solving bandit problems with stationary reward distributions [8], one drawback of VDBE (in particular of ε -greedy) is that exploration actions are chosen uniformly distributed among all

possible actions in the current state. Such exploration behavior can lead to bad performance when many actions in the current state yield to relatively high negative reward, even if this knowledge is present through already learned Q values. Furthermore, Q -function oscillations cause a non-zero level of $\varepsilon(s)$, e.g. caused by stochastic rewards or by function approximators for the Q -function. In turn, this causes excessive selections of bad actions in cases when only a few actions lead to positive reward.

A way of relaxing the above drawback is by combining an ε -greedy policy with Softmax (Equation 5) as proposed by Wiering as the *Max-Boltzmann Exploration* method (MBE) [10]. MBE behaves the same as ε -greedy except that exploration actions are selected according to the Softmax rule:

$$\pi(s) = \begin{cases} \text{Softmax action according to Equation 5} & \text{if } \xi < \varepsilon \\ \operatorname{argmax}_{a \in \mathcal{A}(s)} Q(s, a) & \text{otherwise,} \end{cases} \quad (8)$$

where ξ is a uniform random number from the interval $[0, 1]$. Although *Max-Boltzmann Exploration* requires two parameters to be set (τ and ε), advantages of both methods are combined into one method [10].

The idea of combining both methods is now used for extending VDBE to the so-called VDBE-Softmax method. In contrast to MBE, VDBE-Softmax adapts the state-dependent exploration rate $\varepsilon(s)$ according to VDBE but selects random actions according to Softmax in case of $\xi < \varepsilon(s)$. Furthermore, in order to ease the search for reasonable parameters for Softmax, we propose using a normalization of the Q values into the interval $[V_{\text{normMin}}, V_{\text{normMax}}]$, e.g. $[-1, 1]$, and having the temperature parameter of Softmax set constantly to the value of $\tau = 1$. With this, a mean independency of the distribution of Q values in state s is achieved that enables the selection of τ more intuitively. In our experiments, such an approach turned out to be sufficient for suppressing the selection of actions yielding to highly negative reward in case of $\xi < \varepsilon(s)$. Finally, Algorithm 1 depicts the interaction between Q -learning and VDBE-Softmax, where SARSA is combined analogously when replacing lines 13-15 of Algorithm 1 according to Equation 2.

4 Experiments in the Cliff-Walking Task

The proposed method has been evaluated on the cliff-walking task presented by Sutton and Barto [1]. As shown in Figure 1, the goal of the agent is to learn a path from the start state, S, to the goal state, G. For each step, the agent receives a reward of $r = -1$ except for falling off the cliff which is rewarded by $r = -100$, and where the agent is instantly sent back to S.

In the cliff-walking task, Sutton and Barto demonstrated the different learning behaviors of *on-* and *off-policy* methods when using stochastic policies [1]. As a result, the agent learns the safe path when using Sarsa, but the optimal path when using Q -learning. The optimal (shortest) path, however, is a bad choice

¹ In particular, Sutton and Barto used ε -greedy having $\varepsilon = 0.1$.

Algorithm 1. *Q*-LEARNING WITH VDBE-SOFTMAX

```

1: Initialize  $Q(s, a)$  arbitrarily, e.g.  $Q(s, a) = 0$  for all  $s, a$ 
2: Initialize  $\varepsilon(s)$  arbitrarily, e.g.  $\varepsilon(s) = 1$  for all  $s$ 

3: for each episode do
4:   Initialize start state  $s$ 
5:   repeat
6:      $\xi \leftarrow \text{rand}(0..1)$ 
7:     if  $\xi < \varepsilon(s)$  then
8:        $a \leftarrow \text{SOFTMAX}(\mathcal{A}(s))$ 
9:     else
10:       $a \leftarrow \text{argmax}_{b \in \mathcal{A}(s)} Q(s, b)$ 
11:    end if
12:    take action  $a$ , observe reward  $r$  and successor state  $s'$ 
13:     $b^* \leftarrow \text{argmax}_{b \in \mathcal{A}(s')} Q(s', b)$ 
14:     $\Delta \leftarrow r + \gamma Q(s', b^*) - Q(s, a)$ 
15:     $Q(s, a) \leftarrow Q(s, a) + \alpha \Delta$ 
16:     $\varepsilon(s) \leftarrow \delta \cdot \frac{1 - e^{-|\alpha \cdot \Delta|}}{1 + e^{-|\alpha \cdot \Delta|}} + (1 - \delta) \cdot \varepsilon(s)$ 
17:     $s \leftarrow s'$ 
18:   until  $s$  is terminal state
19: end for

```

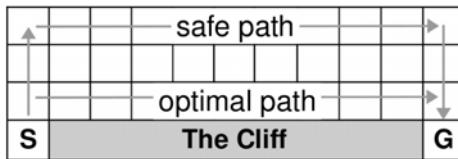


Fig. 1. The cliff-walking task as presented by Sutton and Barto [\[1\]](#)

in this example since the agent will travel right along the edge of the cliff and which occasionally results in falling off.

4.1 Experiment Setup

The cliff-walking experiment has been setup as follows and having the results averaged over 1000 experiments each having 400 episodes. An episode begins in the start state, S, and terminates when: (1) the agent has walked a maximum of 100 steps, or (2) the agent arrived at the goal state, G. Throughout the experiment, the step-size parameter α has been constantly set to the value of $\alpha = 0.8$.

Since the learning problem is an episodic task, no discounting ($\gamma = 1$) has been used. In this experimental setting, Q -learning and Sarsa have been investigated with ε -greedy, Softmax and VDBE policies using constant parameter settings, and using a tabular approximation of the value function. In experiments with VDBE and VDBE-Softmax, all exploration probabilities have been initialized with $\varepsilon(s) = 1$, as well all δ 's been configured with $\delta(s) = \frac{1}{|\mathcal{A}(s)|}$. For VDBE-Softmax, the normalization interval has been set to $[0, 1]$ using $\tau = 1$ for the Softmax method. At the beginning of each experiment, all state-action values have been optimistically initialized with $Q_{t=0}(s, a) = 0$, thus causing additional exploration in the first phase of learning.

4.2 Results

The experimental results of the cliff-walking study are shown in Figure 2. It is observable that all four exploration methods perform optimal when having (almost) a greedy exploration parameter configured, i.e. $\varepsilon = 0$ for ε -greedy, $\tau = 0.04$ for Softmax, or $\sigma = 100$ for VDBE and VDBE-Softmax. In case of stochastic policies, it can be observed that Sarsa outperforms Q -learning when using ε -greedy or Softmax policies, which confirms the results of Sutton and Barto. Interestingly, the performance of Q -learning in conjunction with ε -greedy is sometimes even better in the first episodes compared to the converged performance in the last episodes, e.g. when ε is set to 0.5. The effect of unlearning such an apparently better behavior is caused by greedy action selections in the first phase of learning, and also led back to the insecurity of the value-function estimates. Due to this, the agent walks more often away from the cliff during the first episodes, but travels right along the edge once the value function converges to the true values. In terms of learning speed, no remarkable changes other than the speed of learning were observable for different settings of α when using ε -greedy or Softmax.

In contrast, a different behavior is observable for VDBE as shown in Figure 2(c). Due to the nature of pursuing to greedy, VDBE in conjunction with Q -learning always converged to the optimal results under any settings of σ . On the contrary, VDBE in combination with Sarsa shows to converge to the optimal results only for high inverse sensitivities ($\sigma \gtrsim 10$), but diverges for values of $\sigma < 10$. The reason for this behavior is that Sarsa has no convergence guarantee, and oscillations in the value function are caused when the policy is stochastic. Furthermore, these oscillations cause VDBE to increase the exploration probability, thus causing the agent to explore its environment constantly. In evaluations for other values of α , the σ parameter could be more reduced the more α is reduced at the same time. The advantage of additionally combining Softmax to the VDBE-Softmax method is shown in Figure 2(d). In contrast to VDBE, the results of Sarsa in conjunction with VDBE-Softmax also converge to near optimal results independently of σ , which only had influence on the speed of convergence.

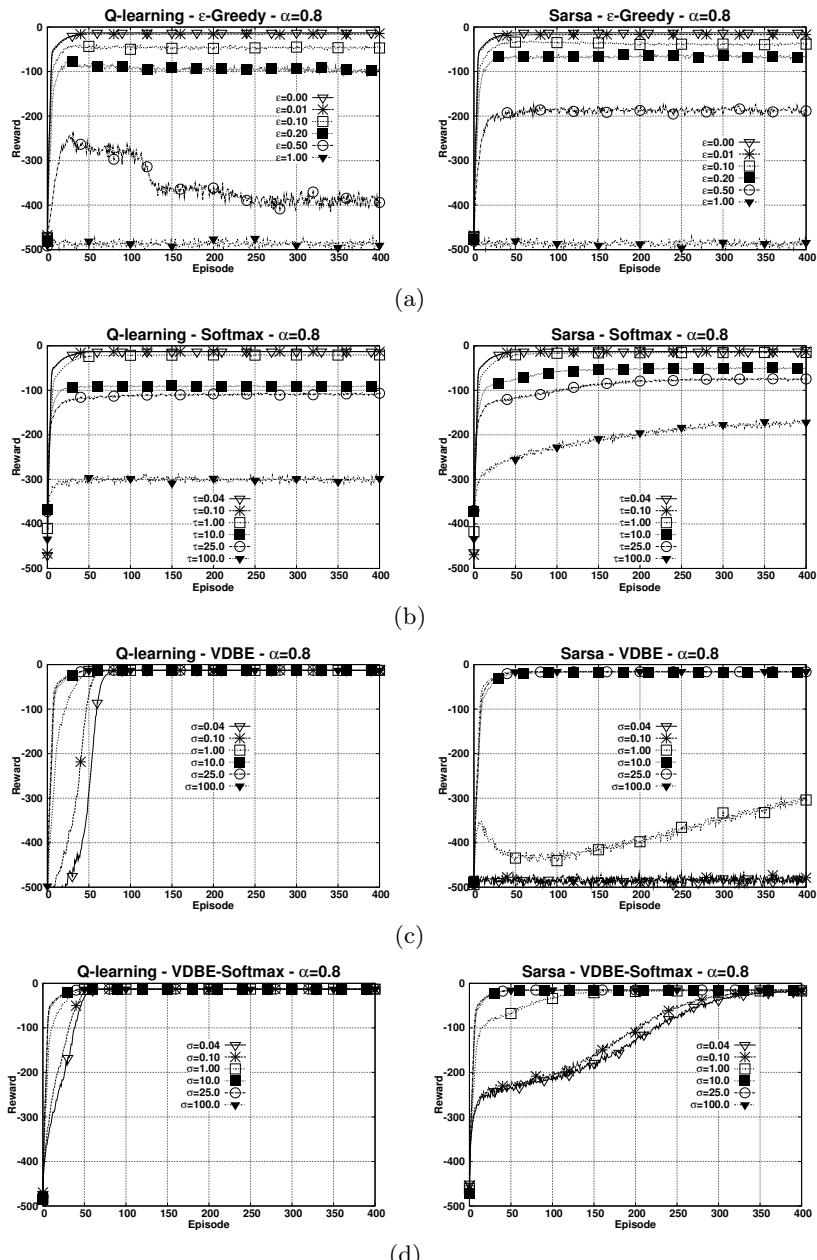


Fig. 2. Comparison of the cumulative reward per episode on the cliff-walking task using Sarsa and Q -learning in conjunction with: (a) ε -greedy, (b) Softmax, (c) VDBE and (d) VDBE-Softmax. Results are averaged over 1000 experiments.

5 Experiments in the Bandit-World Task

The second experiment has been conducted in an extension of the multi-armed bandit problem proposed here as the “*Bandit-World*” problem. In addition to the original multi-armed bandit problem [9], the environment in the bandit world consists of multiple states and diverse bandits (in this example states \mathbf{B}_1 and \mathbf{B}_2). The reward distributions of bandit states are unequal and the agent has to decide whether it sticks with the first bandit it reaches or whether it travels around in hope to find another (maybe better) bandit. Traveling around is expensive since each transition to another state is rewarded negatively by the value of $r = -1$. On the contrary, the reward for choosing a bandit lever in states \mathbf{B}_1 and \mathbf{B}_2 is drawn randomly according to a normal distribution $\mathcal{N}(Q^*(\mathbf{B}, a_{\text{lever}}), 1)$.

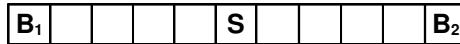


Fig. 3. The bandit-world task. \mathbf{S} indicates the start state; \mathbf{B}_1 and \mathbf{B}_2 indicate two different bandits.

5.1 Experiment Setup

The bandit world has been evaluated as follows. Since the environment is an episodic multi-state MDP having the rewards be a mixture of both deterministic and stochastic scalars, we evaluated low values for the step-size parameter α . Each bandit consists of three levers with mean values $Q^*(\mathbf{B}_1) = \{-1.0, 0.0, 1.0\}$ and $Q^*(\mathbf{B}_2) = \{2.0, 3.0, 4.0\}$. In bandit states, a fourth possible action is allowed (a_{left} or a_{right}) that leaves the bandit and which is rewarded by $r = -1$ (which also applies for every other transition in non-bandit states). Results are averaged over 500 experiments each running over 400 episodes. An episode begins in the start state \mathbf{S} and terminates after $T = 100$ steps. In this experimental setting, Q -learning and Sarsa have been investigated with ε -greedy, Softmax and VDBE policies using constant parameter settings, and using a tabular approximation of the value function. In experiments with VDBE and VDBE-Softmax, all exploration probabilities have been initialized with $\varepsilon(s) = 1$, as well all δ 's been configured with $\delta(s) = \frac{1}{|\mathcal{A}(s)|}$. All state-action values have been optimistically initialized with $Q_{t=0}(s, a) = 0$, thus causing additional exploration in the first phase of learning. Furthermore, the Softmax method used by VDBE-Softmax has been configured with temperature $\tau = 1$ and normalization boundaries $[-1, 1]$. Finally, we evaluated discounting in the bandit-world task with $\gamma = 0.9$.

5.2 Results

Figure 4 shows the results of the bandit-world experiments. For ε -greedy and Softmax policies, almost no performance difference is observable when using Q -learning and Sarsa. For both policies, the worst-case reward/episode is about -50 in case the action-selection policy is pure random ($\varepsilon = 1.0$ and $\tau \geq 100.0$).

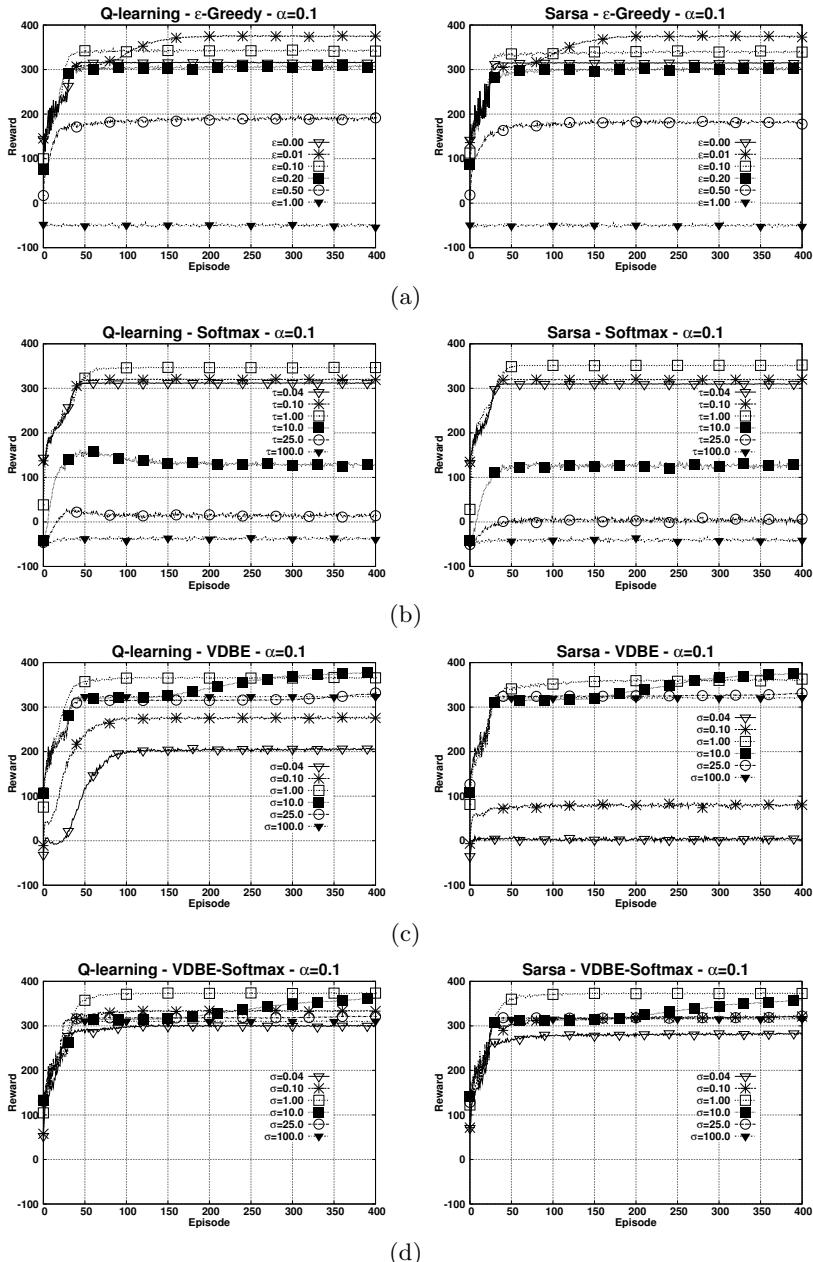


Fig. 4. Comparison of the cumulative reward per episode on the bandit-world task using Sarsa and Q -learning in conjunction with: (a) ϵ -greedy, (b) Softmax, (c) VDBE and (d) VDBE-Softmax. Results are averaged over 500 experiments.

In contrast, VDBE in combination with Q -learning converged to a worst-case reward/episode of about 200, and to 300 for VDBE-Softmax respectively. In combination with Sarsa, VDBE shows (again) to diverge when using low values for σ , since Sarsa causes value-function oscillations in case of stochastic policies. In turn, VDBE converged to near optimal results for settings of $\sigma \geq 1$. Finally, the results show that the worst-case reward/episode is much higher for VDBE-Softmax compared to the other three methods. Interestingly, VDBE, VDBE-Softmax and Softmax policies turned out to maximize the cumulative reward when setting the exploration parameter (σ or τ) to the value of 1.

6 Discussions and Conclusions

This paper showed that VDBE can successfully be applied to balance exploration/exploitation also in multi-state MDPs, which answers the first open question mentioned above. The obtained results lead to the conclusion that VDBE in conjunction with Q -learning is able to outperform other basic exploration methods, since the information is not only based on current information of the value function, but also biased on the progress of learning the function. The results highlight the importance of using learning algorithms that are proven to converge. As a counterexample, the experiment with Sarsa in conjunction with VDBE revealed that oscillations in the value function can lead to a constant level of exploration, thus to full exploration in the worst-case. Such behavior can sometimes successfully be handled by: (1) a fine-tuning of the inverse sensitivity σ , (2) a fine-tuning of the step-size parameter α or (3) by using the proposed VDBE-Softmax method. Finally, this fact answers the second open question because convergence proofs exist for Q -learning (*off-policy* control) rather than for Sarsa (*on-policy* control).

Although results were optimal using a pure greedy policy in the cliff-walking task, this setting does not apply for every learning problem as well, and most often a bit of exploration improves learning performance as shown in the bandit-world example. In fact, a pure greedy policy is most often sub-optimal, and less examples such as the cliff-walking problem exist that show the contrary. Only in the limit, the policy should converge to greedy as far as enough information about the environment has been sampled, and which has also been shown in a preceding study of VDBE on the multi-armed bandit problem [8]. Finally, the results also show that extending VDBE with the Softmax method converged much more reliable to near optimal results under a wide range of parameter configurations.

To sum up, the presented results suggest that balancing exploration and exploitation on basis of fluctuations in the value function is a reasonable technique for supporting the decision-making process in reinforcement learning.

Acknowledgements. The authors like to thank F. Schwenker¹, W. Ertel², P. Ertle² and R. Cubek² for valuable comments and discussions.

This work has been conducted within the Collaborative Center for Applied Research on Service Robotics (ZAFH Service Robotics). The authors gratefully acknowledge the research grants of the state Baden-Württemberg and the European Union.

References

- [1] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
- [2] Watkins, C.: Learning from Delayed Rewards. PhD thesis, University of Cambridge, Cambridge, England (1989)
- [3] Thrun, S.B.: Efficient exploration in reinforcement learning. Technical Report CMU-CS-92-102, Carnegie Mellon University, Pittsburgh, PA, USA (1992)
- [4] Kaelbling, L.P.: Learning in embedded systems. MIT Press, Cambridge (1993)
- [5] Auer, P.: Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* 3, 397–422 (2002)
- [6] Vermorel, J., Mohri, M.: Multi-armed bandit algorithms and empirical evaluation. In: Gama, J., Camacho, R., Brazdil, P., Jorge, A., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 437–448. Springer, Heidelberg (2005)
- [7] Heidrich-Meisner, V.: Interview with Richard S. Sutton. In: *Künstliche Intelligenz*, vol. 3, pp. 41–43 (2009)
- [8] Tokic, M.: Adaptive ε -greedy exploration in reinforcement learning based on value differences. In: Dillmann, R., Beyerer, J., Hanebeck, U.D., Schultz, T. (eds.) KI 2010. LNCS, vol. 6359, pp. 203–210. Springer, Heidelberg (2010)
- [9] Robbins, H.: Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society* 58, 527–535 (1952)
- [10] Wiering, M.: Explorations in Efficient Reinforcement Learning. PhD thesis, University of Amsterdam, Amsterdam (1999)
- [11] Rummery, G.A., Niranjan, M.: On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG/TR 166, Cambridge University (1994)
- [12] George, A.P., Powell, W.B.: Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming. *Machine Learning* 65(1), 167–198 (2006)
- [13] Watkins, C., Dayan, P.: Technical note: Q-learning. *Machine Learning* 8(3), 279–292 (1992)
- [14] Daw, N.D., O'Doherty, J.P., Dayan, P., Seymour, B., Dolan, R.J.: Cortical substrates for exploratory decisions in humans. *Nature* 441, 876–879 (2006)

Calculating Meeting Points for Multi User Pedestrian Navigation Systems

Bjoern Zenker and Alexander Muench

University of Erlangen-Nuernberg,

Erlangen, Germany

bjoern.zenker@cs.fau.de

Abstract. Most pedestrian navigation systems are intended for single users only. But pedestrians often prefer going out with other people, meeting friends and covering distances together. Thus we built a navigation system which allows calculating routes for multiple people who want to meet departing at different locations. In this paper we present, how satisfying meeting points can be found. We discuss two approaches, one based on the Steiner Tree Problem in Networks and one based on the Euclidian Steiner Problem which neglects the street network. Both approaches are evaluated and a user study demonstrates the applicability of our solution.

1 Introduction

People often go out with other people, meet friends and prefer covering distances together. According to [1] 70% of all pedestrians travel in a group. [2] found that 71.07% of pedestrian groups were groups consisting of two individuals and 28.93% were groups consisting of two to seven individuals. This results in an average group size of 2.41 individuals. However most pedestrian navigation systems are intended for single users only.

For closing this gap we built a mobile pedestrian navigation system for groups (PNS4G) of users who want to meet. Imagine two first-year fellow students living in two different student dormitories spread across a city. They agree to go to the cinema. On their way they want to talk to each other about their current lectures. But they also have to finish their homework first so they do not want to have too much detour. Thus a compromise between detour and the time walking together is needed. Where should they meet? Figure 1 shows four different possibilities for the individuals p and q with the common destination g . Current navigation systems cannot help users to solve this problem.

The focus of this paper is on finding satisfying meeting points for individuals who depart at different locations and who have a common goal. We will present two approaches to find satisfying meeting points and corresponding routes for the users. These routes together compose a meeting tree. We will cover this problem for two and more individuals. Figure 2 shows a possible meeting tree for three individuals.

First, we will present the state of the art and introduce our multi user pedestrian navigation system in section 2. In section 3 we first present a practical and theoretical formulation of the problem of finding good meeting points. We then propose two methods for

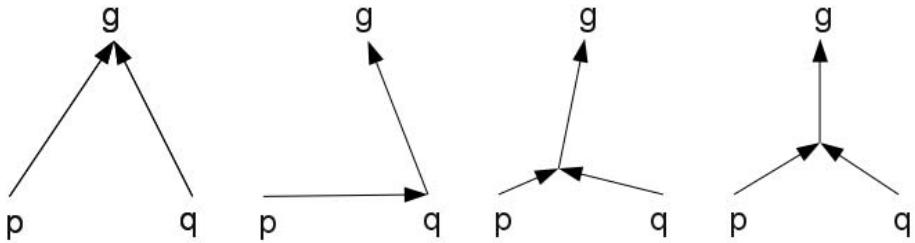


Fig. 1. From left to right: Users p and q meet at the destination g , at q 's position, at some intermediate place, at the Torricelli point

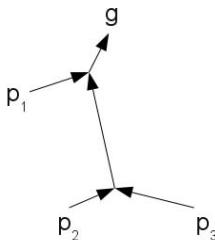


Fig. 2. Meeting tree of three individuals p_1, p_2, p_3 meeting at intermediate points

finding such meeting points in section 4. The first method is based on the Steiner Tree Problem in Networks. As the running time of the used algorithm is too high we research on a second method. This method is based on a relaxation of our problem, namely neglecting the underlying street network. Hence we can employ Euclidian Steiner Trees to model our problem and therefore utilize faster algorithms. Finally, we compare both methods and show the applicability in a user study in section 5. We conclude by giving an outlook for further research and development in section 6.

2 Towards Multi User Pedestrian Navigation

2.1 Pedestrian Navigation

Current pedestrian navigation systems help pedestrians to find their way to their goals by giving turn-by-turn instructions. Today all to the author known pedestrian navigation systems are for single users only. Examples of such systems are Google Maps Navigation, ovi maps, MobileNavigator, PECITAS [3], P-Tour [4], RouteCheckr [5], COMPASS [6] and many more. Besides that, most commonly used routing algorithms for pedestrian navigation in street networks like Dijkstra and A* as well as algorithms for public transport networks like [7] and [8] are single source only. Thus they cannot be used for calculating routes for more individuals or rather meeting trees. But as we have seen, most individuals go out in groups.

To address this, we created the multi user pedestrian navigation system GroupROSE, an extension of ROSE [9]. Every user is running a client software (currently in J2ME) on his mobile phone. One user can invite other users to a location, e.g. a certain cinema. After the users have accepted the invitation, their GPS positions are sent to the server. There, routes for all users are calculated with appropriate meeting points. These routes are displayed on the mobile phones and allow turn-by-turn navigation for each user. Figure 3 shows the client of a user who is going to meet shortly another user. Then, they will continue their journey to their destination conjoint. An overview map of the routes of two users in the city of Berlin can be seen in Figure 4.



Fig. 3. Screenshot: User (cross hairs) will meet his friend (user icon) soon at the meeting point (red cross)

2.2 Steiner Tree Problem

From a theoretical point of view the Steiner Tree Problem resembles the problem of finding meeting points and corresponding routes. Hence we will give a short introduction to this problem, namely to “Find the shortest network spanning a set of given points...” [10]. One can find two similar versions of the Steiner Tree Problem in literature:

Steiner (Tree) Problem in Networks (SPN). Given an undirected network $N = (V, E, c)$ with vertices V , edges E and cost function $c : E \rightarrow \mathbb{R}^+$, and a terminal set $T \subseteq V$ find the subnetwork S of N such that all nodes $t \in T$ are connected and that the total costs $\sum_{x \in E_S} c(x)$ are a minimum. S is called Steiner Minimum Tree (SMT). Vertices $S \in V_S \setminus T$ are called steiner points. SPN is NP-complete [11]. An overview of several exact and approximative algorithms as well as a introduction to SPN is for example given by [10] and [12].

Euclidian Steiner (Tree) Problem (ESP). Given a set T of n points in the Euclidian plane, find a tree S connecting all $t \in T$ minimizing the length of S . Note that this might introduce new points at which edges of the tree meet. A minimal tree for 3

terminals shows the rightmost graph in Figure 1. The three edges meet at the Torricelli point. A prominent exact algorithm was given by [13], heuristics e.g. by [14] and [15]. *Geosteiner* [16] is an implementation of an exact algorithm with special pruning techniques to rule out unfeasible SMTs. Detailed information about this approach can be found in [17].

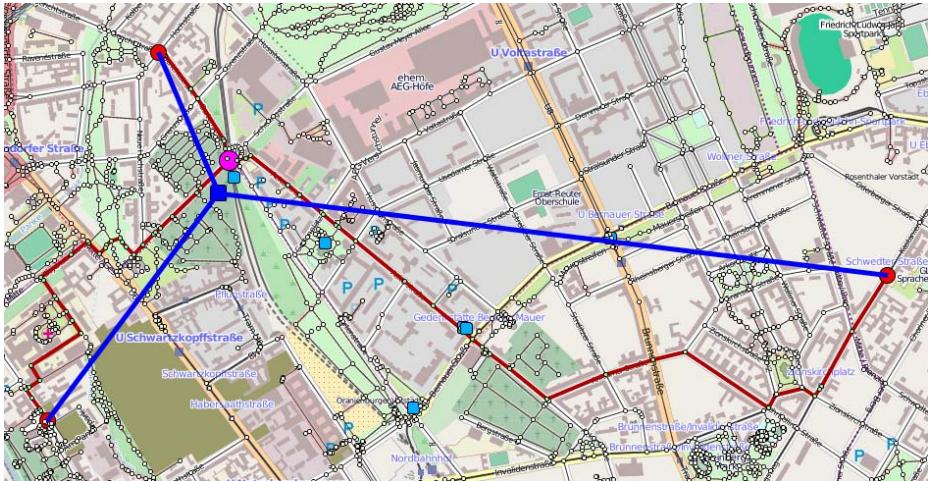


Fig. 4. Comparison between Toricelli point (and lines of sights) and Steiner point (and routes in the street network)

3 Meeting Problem

To the best of our knowledge there is no literature about meeting behaviour and especially meeting point finding of individuals. Thus we will give our own definition of the problem.

3.1 Practical Problem Formulation

Given is a map of a city and a set of individuals starting positions and a goal position in that city. For each individual a route has to be found from it's starting position to the goal position. All these routes together compose a meeting tree (MT). Now, find a MT such that these requirements are optimized:

- the distances travelled together are maximized
- the detour for doing so is minimized.

These requirement correspond to the every day meeting behavoir of people as observed by common sense.

From a social psychological point of view we can formulate a different requirement: Find a MT such that the costs for all individuals are minimized.

By assigning costs for detour and negative costs to distances travelled conjoint we transform the common sense requirements to the social psychological requirements. These requirements also conform to rational choice theory (see e.g. [18]), which thinks of individuals as if they would choose actions to maximize personal advantage.

In this paper we assume that positions are given as pairs of latitude and longitude and that all positions are in a city. The latter assumption means, that individuals can walk from one position to another by following roads.

Next, we will give a more precise formulation of this problem.

3.2 Formal Definition: Meeting Tree Problem (MTP)

We start with some definitions. A path $r = \{v_1, v_2, \dots\}$ is an ordered set of vertices. r_p is the path of individual p . An edge $e_w = (v_w, v_{w+1})$ is a pair of consecutive vertices in a path. The union of all individual's paths $\{r_1, r_2, \dots, r_n\} \in MT$ is the meeting tree. Each vertex has a position. Positions are given as pairs of latitude and longitude. The length of an edge $e = (e_1, e_2)$ will be measured using the great-circle distance $d(e) := \|e_1, e_2\|$.

The cost of a path which is covered by one individual is $c_{single}(r) := \sum_{e \in r} d(e)$. To consider the fact that people prefer to walk together we set costs for paths who are covered conjoint by m individuals to $c_{conjoint}(r) := \frac{c_{single}(r)}{s}$. This means that the costs of a path covered by more individuals is weighted by a factor s . In this paper we will always set $s = m$, which means, that the *conjoint* costs of a path only depend on the distance and not on the number of individuals travelling on the path. The costs of the MT are calculated by $c(MT) := \sum_{r \in MT} c_{conjoint}(r)$.

Now we can formulate the problem: Given is a set of n users with starting positions $\{p_1, p_2, \dots, p_n\} \in P$ and a goal position g . Find the routes $\{r_1, r_2, \dots, r_n\} \in R$ such that $c(MT)$ is minimized.

In the case of $s = m$ the cost function of the MTP equals the cost function of the Steiner Tree Problem. Under the assumption that it does not matter whether we exchange starting positions and goal position, hence writing $T = P \cup g$, finding meeting points in the MTP can be reduced to a Steiner Tree Problem! We are currently preparing a study to research whether this assumption produces a simplification that holds.

In the next section we will present two methods to solve this problem. Section 5.2 shows that our theory achieves good results when compared to meeting points from users.

4 Two Approaches

4.1 Solving MTP in the Street Network

At the beginning of our research we interpreted MTP as an instance of a SPN. Thus, we used an extended Dreyfus-Wagner algorithm as described in [12] to calculate the SMT in the network of the streets. The algorithm first computes the transitive hull of shortest

paths for all pairs of nodes. This equals all SMTs for all pairs of nodes. In further steps these results are used to calculate SMTs for subsets of 3, 4, 5, . . . nodes. As the first step is the same for all possible sets of terminals, this step can be precompiled.

On clippings of maps (from OpenStreetMap) of six different cities we calculated SMTs for three terminal nodes respectively. Note that three terminals equal a MTP with two individuals who want to go to one destination. All terminal nodes were randomly picked and in walking distance to each other. We measured the time t_h of the first step from the Dreyfus-Wagner algorithm to calculate the shortest path transitive hull and the time t_r for the following steps. The results for some cities are shown in table 1. You can clearly see the exponential increase in t_h and t_r . We also measured the times needed for problem instances with more terminals: for each additional terminal the time t_r doubled. In clippings with a practical amount of streets calculating a MT took over 30 seconds. As we wanted to construct a system with a response time smaller than two seconds we explored a second method.

Table 1. Time used by Dreyfus-Wagner algorithm

| city | nodes | shortest paths | t_h | t_r |
|---------|-------|----------------|---------|--------|
| Hamburg | 3 787 | 7 168 791 | 225.26s | 86.49s |
| Berlin | 3 216 | 5 169 720 | 114.69s | 56.77s |
| Madrid | 2 496 | 3 113 760 | 65.87s | 33.28s |

4.2 Solving MTP Geometrically

This method works on a relaxation of the problem. [9] observed that “Pedestrian navigation [...] is not confined to a network of streets, but includes all passable areas, such as walkways, squares, and open areas, within or outside buildings.”. In other words, pedestrian movement can be seen as largely independent of the street network. Inspired by their observation we neglect the actual structure of the street network in a first step. Now our problem resembles the ESP. Such, we can use e.g. the Melzak algorithm to estimate meeting points on a geometric basis only. Afterwards routes to these meeting points are calculated in the street network. The course of action is detailed in the following paragraphs. Our evaluation in section 5.2 affirms that this assumption is reasonable.

First Step: Estimate Theoretical Meeting Points. For calculating meeting points in the ESP we relied on the program GeoSteiner (see section 2.2). The runtime of GeoSteiner for our limited set of terminals is negligible short. We call meeting points obtained by solving an ESP theoretical meeting points (TMPs).

Second Step: Find MPOIs. TMPs calculated in the previous step can be situated in unaccessible places like buildings or lakes or unintuitive places like “42 meters west of house number 14”. Thus we move these points to better locations nearby, which we call practical meeting points (PMPs). A PMP can be in front of what we call a meeting point of interest (MPOI). MPOIs can be e.g. restaurants, bars, bus stops, subway stations, some points-of-interest (POIs), public open places or big crossroads.

We used OpenStreetMap as source for finding MPOIs. Figure 5 shows various MPOIs in downtown Nuremberg.

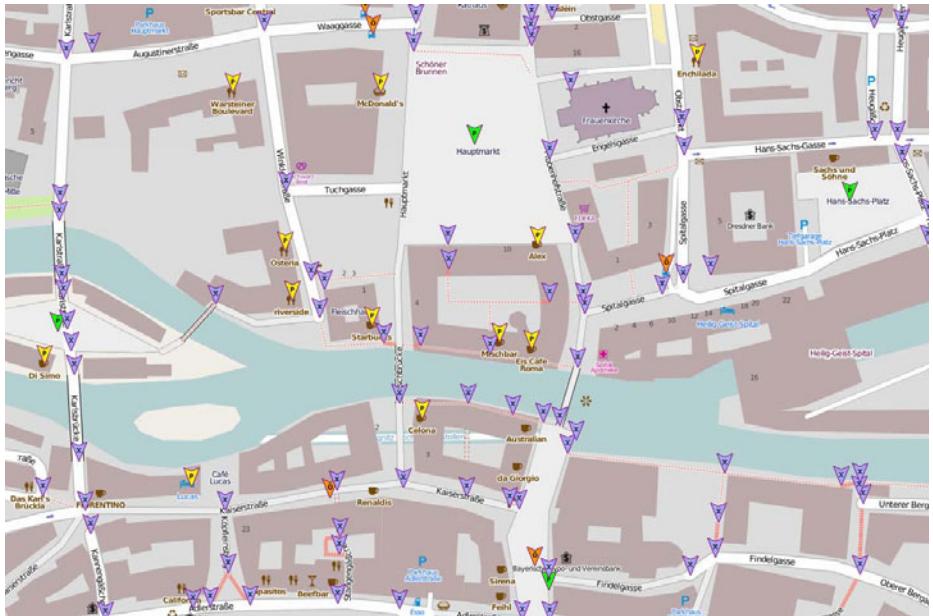


Fig. 5. MPOIs in downtown Nuremberg in front of restaurants (yellow) and bus stops (orange), crossroads (blue) and public open spaces (green)

Third Step: Times and Routes. Now, we calculate walking routes in the street network for the users. Each route reaches from the current user's position through none or some PMP to the goal.

Next, times at which users have to be at specific locations are calculated. Therefore a tree-traversal starting at the destination location labels all vertices in the meeting tree with the time, at which users have to leave when they want to arrive in time at the next vertex. The destination node is initialized with the time at which users want to arrive there, an additional time buffer can be inserted at meeting points.

Using the above tree-traversal, also the individual routes of all users can be extracted from the MT. An example for the result of these three steps is shown in Figure 6.

Note that in this figure e.g. the user from the top left corner has to walk a small stretch of way twice. This is due to our heuristic approach. But one could easily move in an additional step meeting points causing these indirections such, that walking distances back and forth are minimized.

5 Evaluation

We conducted three studies to show the suitability of our theory on meeting points and meeting trees.

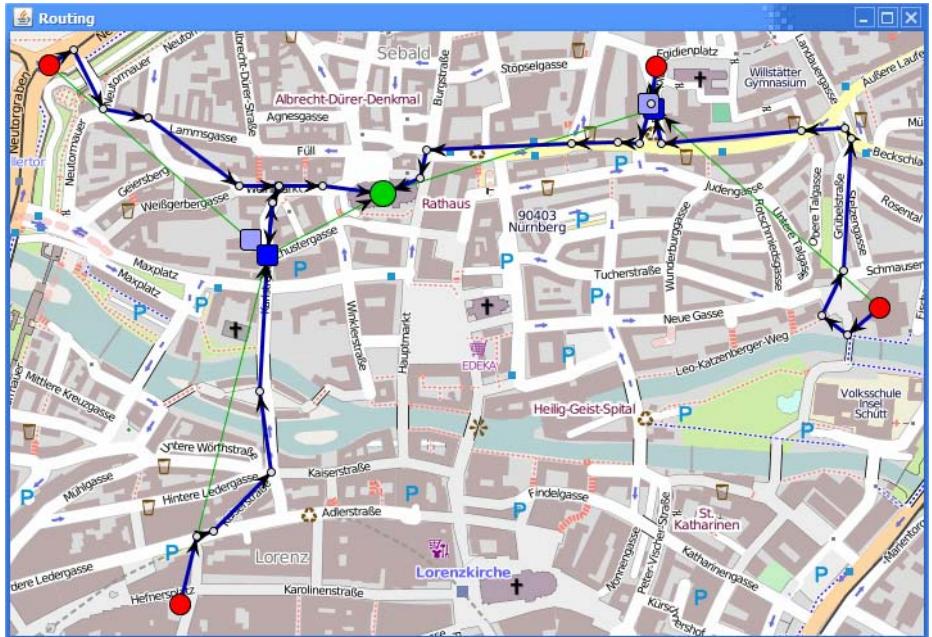


Fig. 6. TMPs (light blue) PMPs (dark blue) and Routes (arrows) of four users (red) to the destination (green)

5.1 Comparing ESP and SPN

To check whether the relaxation, neglecting the street network, yields suitable results, we compared the lengths l_G of meeting trees calculated geometrically with lengths l_N of meeting trees calculated in the street network.

For various numbers of terminals we calculated meeting tree lengths for MTPs in five cities. Average lengths for both methods are shown in table 2

Table 2. Influence from ESP and SPN based approaches on meeting tree length

| n | \bar{l}_N | \bar{l}_G | $ \bar{l}_G - \bar{l}_N = \Delta$ | $\sigma(\Delta)$ | $\Delta : l_N$ |
|-----|-------------|-------------|------------------------------------|------------------|----------------|
| 4 | 3 003m | 3 310m | 308m | 237m | 9.9% |
| 5 | 3 267m | 3 585m | 317m | 192m | 10.3% |
| 6 | 3 657m | 4 212m | 554m | 480m | 15.6% |
| 7 | 3 224m | 4 863m | 640m | 571m | 14.6% |

In the case of four terminals the length of geometric meeting trees are by 9.9% longer than network meeting trees. In our study this means an overall detour of 308 meters for all three individuals in the average. This results in even smaller detours for each individual. As [20] finds “[d]etours of up to about 25% seem to be acceptable to pedestrians.”, detours between 9.9% and 15.6% (12.5% in average) in contrast to the optimal route look very acceptable.

5.2 Meeting Points for Two Persons

In a study we asked 6 participants to mark their preferred meeting points in three different situations in five cities. All situations displayed positions of two individuals and their common destination on a map. The participants had to mark the location which they thought is the best meeting point.

The distance between the Torricelli point and the peoples choices was in the average 287 meters with a standard deviation of 183 meters. As the average meeting tree length in the graph in this study is 2558 meters, the average maximum detour for both individuals is 11.2%. This result also looks acceptable.

5.3 Subjective Evaluation

Additionally we conducted a subjective evaluation of our approach. We presented six scenarios to seven participants. Each scenario consisted of a map with positions of several individuals and a position of their destination. The participants had to draw a meeting tree for each scenario. Afterwards, the probands were presented the automatically created meeting trees from our system using the geometrical approach. Now, they had to answer a questionnaire on a discrete five point Likert scale (0: full reject, 5: full accept). For “Do you think the automatically created meeting tree is a good compromise?” the rated in average 4.31, whether they find the routes practically 4.0, whether the quality of the automatically created meeting tree is equal to their meeting tree was answered with 2.83. We summarize that they rated the automatically created routes positively relating to compromise and practicability. Obviously they still prefer their own routes, as we considered only a very limited set of aspects when calculating MTs.

6 Conclusion

Modeling the meeting behaviour of individuals is a new territory. We proposed a practical and theoretical formulation of the MTP. To solve MTP we presented and evaluated two methods which lead back to the Steiner tree problem.

The runtime of the method based on SPN is (at least when using the exact Dreyfus-Wagner algorithm) too high for our needs. For the future it would be interesting to evaluate the behaviour of approximative algorithms like [14] on the MTP.

For achieving better runtime we invented another method working on ESP which neglects the street network. Heuristics of GeoSteiner guarantee short runtime. Compared to the optimal solution this method results in an average of 12.5% overall detour, which is according to [20] acceptable for pedestrians. Further, our study showed that theoretical meeting points obtained by solving ESP are in average only 287 meters away of meeting points people would choose.

As PNS4G is a new area of research there are still many open questions to answer and many aspects of meeting behaviour to be researched. One problem raised in this paper is, how to set the parameter s , which weights the costs of route parts travelled conjoint. We think that s reflects the public spirit of the individuals who are meeting. Currently we are conducting studies to estimate this parameter. Besides that we focus on integrating support for considering means of public transportation in multi user routing.

Acknowledgements. This work was partially funded by Bundesministerium für Wirtschaft und Technologie (BMWi) in the project ROSE (16IN0689).

References

1. Moussaïd, M., Perozo, N., Garnier, S., Helbing, D., Theraulaz, G.: The walking behaviour of pedestrian social groups and its impact on crowd dynamics. *PLoS ONE* 5(4), e10047 (2010)
2. James, J.: A preliminary study of the size determinant in small group interaction. *American Sociological Review* 16(4), 474–477 (1951)
3. Tumas, G., Ricci, F.: Personalized mobile city transport advisory system. In: ENTER Conference 2009 (2009)
4. Maruyama, A., Shibata, N., Murata, Y., Yasumoto, K.: P-tour: A personal navigation system for tourism. In: Proc. of 11th World Congress on ITS, pp. 18–21 (2004)
5. Voelkel, T., Weber, G.: Routecheckr: personalized multicriteria routing for mobility impaired pedestrians. In: Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility, pp. 185–192 (2008)
6. van Setten, M., Pokraev, S., Koolwaaij, J.: Context-aware recommendations in the mobile tourist application compass. In: De Bra, P.M.E., Nejdl, W. (eds.) AH 2004. LNCS, vol. 3137, pp. 235–244. Springer, Heidelberg (2004)
7. Huang, R.: A schedule-based pathfinding algorithm for transit networks using pattern first search. *Geoinformatica* 11, 269–285 (2007)
8. Ding, D., Yu, J.X., Qin, L.: Finding time-dependent shortest paths over large graphs. In: EDBT Proceedings, pp. 697–706 (2008)
9. Zenker, B., Ludwig, B.: Rose - an intelligent mobile assistant - discovering preferred events and finding comfortable transportation links. In: ICAART, vol. (1), pp. 365–370 (2010)
10. Winter, P.: Steiner problem in networks: a survey. *Networks* 17(2), 129–167 (1987)
11. Karp, R.: Reducibility Among Combinatorial Problems. In: Complexity of Computer Computations: Proceedings, pp. 85 (1972)
12. Proemel, H., Steger, A.: The Steiner tree problem: a tour through graphs, algorithms, and complexity. Friedrick Vieweg & Son (2002)
13. Melzak, Z.: On the problem of Steiner. *Canad. Math. Bull* 4(2), 143–148 (1961)
14. Smith, J., Lee, D., Liebman, J.: An O ($n \log n$) heuristic for Steiner minimal tree problems on the Euclidean metric. *Networks* 11(1), 23–39 (1981)
15. Chang, S.: The generation of minimal trees with a Steiner topology. *Journal of the ACM (JACM)* 19(4), 699–711 (1972)
16. Warme, D., Winter, P., Zachariasen, M.: Exact algorithms for plane Steiner tree problems: A computational study. *Advances in Steiner Trees*, 81–116 (2000)
17. Winter, P.: An algorithm for the Steiner problem in the Euclidean plane. *Networks* 15(3), 323–345 (1985)
18. Becker, G.: The economic approach to human behavior. University of Chicago Press, Chicago (1976)
19. Wuersch, M., Caduff, D.: Refined route instructions using topological stages of closeness. In: Li, K.-J., Vangenot, C. (eds.) W2GIS 2005. LNCS, vol. 3833, pp. 31–41. Springer, Heidelberg (2005)
20. Helbing, D., Molnar, P., Farkas, I., Bolay, K.: Self-organizing pedestrian movement. *Environment and Planning B* 28(3), 361–384 (2001)

Algorithmic Debugging to Support Cognitive Diagnosis in Tutoring Systems

Claus Zinn

University of Konstanz, Department of Computer and Information Science,
Box D188, 78457 Konstanz, Germany
`claus.zinn@uni-konstanz.de`

Abstract. Cognitive modelling in intelligent tutoring systems aims at identifying a learner's skills and knowledge from his answers to tutor questions and other observed behaviour. In this paper, we propose an innovative variant of Shapiro's algorithmic debugging technique whose application can be used to pin-point learners' erroneous behaviour in terms of an irreducible disagreement to the execution trace of an expert model. Our variant has two major benefits: in contrast to traditional approaches, it does not rely on an explicit encoding on mal-rules, and second, it induces a natural teacher-learner dialogue with no need for the prior scripting of individual turns or higher-level dialogue planning.

1 Introduction

The interpretation and diagnosis of student answers is one of the central issues to be addressed when building intelligent tutoring systems (ITSs). Depending on the approach followed, it relies on a variety of knowledge sources such as domain models (modelling the expertise for the machine tutor in a given domain), task models (supporting students' problem solving process), error models (anticipating the many kinds of typical learner errors), and individual student models (capturing given learners' knowledge, strengths and weaknesses).

The first tutoring systems modelled learners solely in terms of expert skills or the lack thereof. The *overlay approach* only requires an adequate representation and operationalisation of expertise in terms of factual knowledge units and procedural skills. It assumes that all differences between learner and expert behaviour can be reduced to the learner's lack of skill. The studies of Brown & Burton and others suggest however that student errors cannot be described in terms of absent expert knowledge only [2]. They argue that the representation of expert knowledge must be complemented by a *bug library* to account for typical or high-frequent student errors in a given domain, usually in terms of buggy variants of expert skills. An erroneous student answer can then be reproduced by finding a combination of expert and buggy skills that yields the same answer. The resulting deep-structure model pin-points a student's misconception and supports the generation of appropriate corrective or remedial feedback.

The quality of cognitive diagnosis depends on the right granularity of (expert and buggy) skill decomposition, which in turn profits from an in-depth analysis

of a large and representative number of student protocols. Clearly, cognitive diagnosis that uses bug libraries can only recognise the errors it knows about, and usually the amount of buggy knowledge easily surpasses the amount of expert knowledge. The (De)Buggy programs [23], *e.g.*, relied on a bug library of 120 primitive and compound bugs to model errors in multi-column subtraction. Given the computational complexity of the approach, the systems performed diagnosis off-line, following a complex process of eliminating error hypotheses.

Model tracing tutors tackle the complexity issue by inviting learners to provide their answers in a piecemeal fashion. It is thus no longer necessary to reproduce a student's line of reasoning from question to (final) answer; only the student's next step towards a solution is analysed, and immediate feedback is given. While tutoring systems such as the Lisp Tutor [74] and the Algebra Tutor [6] have been highly successful, they are also expensive to build. A time-consuming cognitive task analysis now goes hand in hand with user interface design that encourages or enforces students to deliver their solution step by step.

In this paper, we report a method for the automated identification of errors that only requires the student's *full* answer to a given problem and a logic program encoding the expert's problem solving. The method relies on an innovative use of *algorithmic debugging* to identify learner errors by the analysis of correct (*sic*) Prolog-based procedures, given the answers of an oracle – a role being played by the student. Compared to previous approaches to cognitive diagnosis, the method does not rely on bug libraries and has low computational complexity. It supports the analysis of learners committing multiple bugs by attacking bugs one after another, and it is immune to learners giving inconsistent replies. Moreover, the execution of the method supports the generation of tutorial interactions without requiring dialogue planning or scripting. In addition, we can give a mechanisation of the oracle, which relieves the student from answering any tutor question at all. Errors can thus be identified without dialogue intervention.

2 Background

In this section, we give a brief account on multi-column subtraction, typical errors in this domain, and Shapiro's original algorithmic debugging method.

2.1 Multi-column Subtraction

Fig. 1 gives an implementation of multi-column subtraction in Prolog. Sums are processed column by column, from right to left. The predicate `subtract/2` implements the recursion, and `process_column/2` gets a partial sum, processes its right-most column and takes care of borrowing (`increment/2`) and payback (`increment/2`) actions. A column is represented as 3-element term (M , S , R) representing minuend, subtrahend and result cell. The program code implements the *equal additions method*, also known as *Austrian method*. When the subtrahend S is greater than the minuend M , then M is increased by 10 (borrowing) before the difference between M and S is taken. To compensate, the S in the column left to the current one is then increased by one (payback).

```

subtract(Sum, Sum)      :- finished(Sum).
subtract(Sum, NewSum) :- 
    process_column(Sum, Sum1),
    shift_left(Sum1, Sum2, ProcessedColumn),
    subtract(Sum2, SumFinal),
    append(SumFinal, [ProcessedColumn], NewSum).

process_column(Sum, NewSum) :-
    butlast(Sum, LastColumn),    allbutlast(Sum, RestSum),
    subtrahend(LastColumn, Sub), minuend(LastColumn, Min),
    Sub > Min,
    add_ten_to_minuend(LastColumn, LastColumn1),
    take_difference(LastColumn1, LastColumn2),
    butlast(RestSum, LastColumnRestSum), allbutlast(RestSum, RestSum1),
    increment(LastColumnRestSum, LastColumnRestSum1),
    append(RestSum1, [LastColumnRestSum1, LastColumn2], NewSum).

process_column(Sum, NewSum) :-
    butlast(Sum, LastColumn),    allbutlast(Sum, RestSum),
    subtrahend(LastColumn, Sub), minuend(LastColumn, Min),
    Sub <= Min,
    take_difference(LastColumn, LastColumn1),
    append(RestSum, [LastColumn1], NewSum).

shift_left( SumList, RestSumList, Item ) :- 
    allbutlast(SumList, RestSumList), butlast(SumList, Item).

add_ten_to_minuend( (M,S,R), (M10,S, R) ) :- irreducible, M10 is M+10.
increment(          (M,S,R), (M, S1,R) ) :- irreducible, S1 is S+1.
take_difference(   (M,S,_R), (M, S, R1)) :- irreducible, R1 is M-S.

minuend( (M,_S,_R), M). subtrahend( (_M,S,_R), S). finished( [] ).
```

Fig. 1. Multi-column subtraction

2.2 Error Analysis in Multi-column Subtraction

Some student errors may be caused by a simple oversight (usually, students are able to correct such errors as soon as they see them), but others are *systematic errors* (those keep re-occurring again and again). It is the systematic errors that we aim at diagnosing as they indicate a student's wrong understanding about some subject matter. The small sample of errors given in Fig. 2 can be classified as *errors of omission* (forget to do something), see (b,c); *errors of commission* (doing the task incorrectly), see (a,e); and *sequence errors* (doing the task not in the right order), see (d). Rather than having pre-compiled explicit representations of buggy program fragments (following the bug library approach) to detect such errors, we would like to use the method of algorithmic debugging on the expert program to help identifying them.

| | | |
|--|--|--|
| $ \begin{array}{r} 5 & 2 & 4 \\ - 2 & 9 & 8 \\ \hline = 3 & 7 & 4 \end{array} $ | $ \begin{array}{r} 3 & 8 & 2 \\ - 3 & 5 \\ \hline = 4 & 7 \end{array} $ | $ \begin{array}{r} 3 & 2 \\ - 1 & 7 \\ \hline = 2 & 5 \end{array} $ |
| (a) Always subtracting the smaller from the larger number | (b) Not finishing the task | (c) Forgot to payback |
| $ \begin{array}{r} 1 & 1 & 2 & 3 \\ - 4 & 9 & 0 \\ \hline = 1 & 7 & 2 & 2 \end{array} $ | $ \begin{array}{r} 5 & 2 & 3 & 4 \\ - 5 & 6 & 7 \\ \hline = 2 & 7 & 7 & 7 \end{array} $ | |
| (d) Perform algorithm from left to right | (e) Accumulating all paybacks to highest place value | |

Fig. 2. A small selection of errors in subtraction

2.3 Shapiro's Algorithmic Debugging

Shapiro's algorithmic debugging technique for logic programming prescribes a systematic manner to identify bugs in programs. In the top-down variant, the program is traversed from the goal clause downwards. At each step during the traversal of the program's AND/OR tree, the programmer is taking the role of the *oracle*, and answers whether the currently processed goal holds or not. If the oracle and the buggy program agree on a goal, then algorithmic debugging passes to the next goal on the goal stack. If the oracle and the buggy program disagree on the result of a goal, then this goal is inspected further. Eventually an *irreducible agreement* will be encountered, hence locating the program's clause where the buggy behaviour is originating from.

Shapiro's algorithmic debugging method extends, thus, a simple meta-interpreter for logic programs. During the meta-interpretation of the program, it generates oracle questions. Given the programmer's answer to a Prolog clause, it is either silently executed to quickly recur algorithmic debugging on the remaining clauses, or investigated further to identify the source of the bug.

In his thesis, Shapiro gives several variants or extensions to account for various types of erroneous program code, including procedures terminating with incorrect output and non-terminating procedures. He also gives algorithms for debugging a program top-down or bottom-up, and also for minimising the number of queries the oracle (*i.e.*, the programmer) needs to answer.

3 Algorithmic Debugging in Tutoring

Shapiro devised algorithmic debugging to systematically identify bugs in incorrect programs. Our Prolog code for multi-column subtraction in Fig. II, however, presents the expert model, that is, a presumably correct program. Given that cognitive modelling seeks to reconstruct students' erroneous procedures by an analysis of their problem-solving behaviour, it is hard to see – at least at first sight – how algorithmic debugging might be applicable in this context. There is a

simple but almost magical trick, however. We can turn Shapiro’s algorithm on its head: instead of having the oracle (the programmer) specifying how the assumed incorrect program should behave, we take the expert program to take the role of the buggy program, and the role of the oracle is filled by a student’s potentially erroneous answers. An irreducible disagreement between program behaviour and given answer then helps indicating a student’s potential misconception.

Our algorithm, especially adapted for tutoring, is given in Fig. 3. Our algorithm traverses a given program in a top-down manner. There are four cases. If a goal is a conjunction of goals, then the algorithm is called for each conjunct. If a goal is a simple goal, then we distinguish goals that can be discussed (`on_discussion_table/1`) from those that cannot and should not be discussed. In the latter case, we check whether the current goal is a built-in predicate (in which case it is called), or whether it is a user-defined goal (in which case, its body is subjected to algorithmic debugging). It is the second clause where the main part of algorithmic debugging takes place. Given a goal that can be discussed, we use a fail-safe approach to evaluate whether it succeeds or not; we then ask the oracle whether it agrees or disagrees with the fact that the goal succeeded (with some of its arguments potentially instantiated) or failed. When there is agreement on a goal, we regard the goal (and its subgoals) as processed, and continue with other goals on the stack resulting from recursion. If there is disagreement on a goal, we check whether we have identified an irreducible agreement (in which case we terminate the algorithm with the goal in question), or not (in which case we inspect the goal’s body).

Note. Programs that model expert problem solving will often have some rather technical steps that should not be subjected to questioning. In our subtraction routine, these are, *e.g.*, the clauses `butlast/2` and `allbutlast/2`; while they are necessary to implement the recursion to process partial sums from right to left, they might be hard to grasp for students and perceived as too low-level. The clause `on_discussion_table_p(Goal)` will fail on these technical clauses.

4 Example

We reconsider two of the five examples of typical subtraction errors. The GUI-based *exercise sheet* that students will interact with is presented further below.

4.1 Single Error Tracking

The result in Fig. 2(c) constitutes a simple error and can be described as a single omission in the program code; the learner forgets to honor the payback operation, following the borrowing that happened in the first (right-most) column.

$$\begin{array}{r}
 & 3 & 2 \\
 - & 1 & 7 \\
 \hline
 = & 2 & 5
 \end{array}$$

```

algorithmic_debugging( Goal1, Goal2 ) :- !,
    algorithmic_debugging( Goal1 ),
    algorithmic_debugging( Goal2 ).

algorithmic_debugging( Goal ) :-
    on_discussion_table_p( Goal ), !,
    copy_term( Goal, CopyGoal ), eval_goal( Goal, SucceededP ),
    ask_oracle( Goal, SucceededP, oracleAnswer ),
    (
        oracleAnswer = no
    ->
        (
            clause( CopyGoal, Clause ),
            (
                ( Clause == true ; Clause = ( irreducible, _Subgoals ) )
            ->
                throw( Goal )
            ;
                algorithmic_debugging( Clause )
            )
        )
    ;
        true
    ).

algorithmic_debugging( Goal ) :-
    predicate_property( Goal, built_in ), !,
    call(Goal).

algorithmic_debugging( Goal ) :-
    clause( Goal, Clause ),
    algorithmic_debugging( Clause ).

get_diagnosis( Goal, Diagnosis ) :-
    catch( algorithmic_debugging( Goal ),
        Diagnosis,
        ( Diagnosis = true
        -> format('`~w`n', [ 'correct solution' ])
        ; format('`~w` ~w`n', [ 'irreducible disagreement:', Diagnosis ]) ) ).
```

Fig. 3. Algorithmic debugging for tutoring (top-down)

Given our implementation of algorithmic debugging, the following dialogue (see clause `ask_oracle/3`) between system and learner will identify the error:

```
get_diagnosis(subtract([(3,1,S1),(2,7,S2)],[(3,1,2),(12,7,5)],Diagnosis).

do you agree that the following goal holds:
    subtract([ (3,1,_G226), (2,7,_G235) ], [ (3,2,1), (12,7,5) ])
|: no.

do you agree that the following goal holds:
    process_column([(3,1,_G475), (2,7,_G484)],[(3,2,_G475), (12,7,5)])
|: no.

do you agree that the following goal holds:
    add_ten_to_minuend((2,7,_G652), (12,7,_G652))
|: yes.

do you agree that the following goal holds:
    take_difference((12,7,_G652), (12,7,5))
|: yes.

do you agree that the following goal holds:
    increment((3,1,_G643), (3,2,_G643))
|: no.
irreducible disagreement: increment((3,1,_G643), (3,2,_G643))
```

When the student corrects the subtrahend in the left-most column, and subsequently updates the corresponding difference, we obtain the correct solution. Re-running algorithmic debugging will have both sides agree on the top clause.

4.2 Tracking Multiple Errors

The method also works well with student answers containing multiple errors. In this case, errors are attacked and repaired one by one. After each correction, the algorithm is re-run. Having the method memoizing the student answers to prior questions will avoid re-asking some – but not all – for them in subsequent runs.

Reconsider the example given in Fig. 2(a).

$$\begin{array}{r} & 5 & 2 & 4 \\ - & 2 & 9 & 8 \\ = & 3 & 7 & 4 \end{array}$$

First Run.

```
do you agree that the following goal holds:
    subtract([ (5,2,_G226), (2,9,_G235), (4,8,_G244) ],
              [ (5,3,2), (12,10,2), (14,8,6) ])
|: no.
```

```

do you agree that the following goal holds:
    process_column([(5,2,_G523), (2,9,_G532), (4,8,_G541)],
                  [(5,2,_G523), (2,10,_G532), (14,8,6)])
|: no.

do you agree that the following goal holds:
    add_ten_to_minuend((4,8,_G808), (14,8,_G808))
|: no.
irreducible disagreement: add_ten_to_minuend((4,8,_G808), (14,8,_G808))

```

Let us assume that the student corrects himself by only adding ten to the minuend in the right-most column. We re-run algorithmic debugging.

Second Run. The second run will produce the following interaction:

```

do you agree that the following goal holds:
    subtract([(5,2,_G226), (2,9,_G235), (4,8,_G244)],
              [(5,3,2), (12,10,2), (14,8,6)])
|: no.

do you agree that the following goal holds:
    process_column([(5,2,_G523), (2,9,_G532), (4,8,_G541)],
                  [(5,2,_G523), (2,10,_G532), (14,8,6)])
|: no.

do you agree that the following goal holds:
    add_ten_to_minuend((4,8,_G808), (14,8,_G808))
|: yes.

do you agree that the following goal holds:
    take_difference((14,8,_G808), (14,8,6))
|: no.
irreducible disagreement: take_difference((14,8,_G808), (14,8,6))

```

With the student having corrected the difference sum in this column, we start the third run.

Third Run. When we omit Oracle questions already asked (except the one that indicated the irreducible disagreement from the last run), we get the following questions:

```

do you agree that the following goal holds:
    take_difference((14,8,_G808), (14,8,6))
|: yes.

do you agree that the following goal holds:
    increment((2,9,_G799), (2,10,_G799))
|: no.
irreducible disagreement: increment((2,9,_G799), (2,10,_G799))

```

Once the student realizes the increment error, we start anew. Either the student learned from the past three runs so that he corrects the other remaining errors

in the exercise sheet, or algorithmic debugging will yield similar results for the processing of the middle column.

The practicality of our approach has to address two issues: (i) how to map Prolog queries into questions that learners can easily understand; and (ii) how to optimise the question-answering process induced by algorithmic debugging.

5 Practical Use in Tutoring

5.1 Graphical User Interface – Exercise Sheet

We have implemented the following browser-based graphical user interface for students to tackle multi-column subtraction tasks (see Fig. 4). In addition to rows for minuend, subtrahend and result cells, it also consists of explicit representations for borrow and payback cells. When students click on a “B” (“P”) cell, it automatically swaps its values to “10” (“1”). Clicking on a result cell “S” brings up a number pane with digits from 0-9. The GUI’s representation of the subtraction task extends the one used in the Prolog program `subtract/2`, but the B_i can be easily combined with the cells for minuends (and the P_i with the subtrahends). Moreover, to indicate to learners which GUI cells requires their attention, an adaptation to the program given in Fig. 1 was necessary, namely, the introduction of a column counter that is being increased at each recursive call to `subtract/3`. During algorithmic debugging, we can thus enhance tutorial interaction by highlighting the corresponding cells or columns in the GUI; erroneous cells that correspond to irreducible disagreements are marked in red.

| Problem Statement | | | | Click on the active buttons to solve. |
|-------------------|---|----|----|---------------------------------------|
| Borrow Row | B | 10 | 10 | 10 |
| Minuends | 1 | 2 | 3 | 4 |
| Subtrahends | 0 | 5 | 6 | 7 |
| Payback Row | 1 | 1 | 1 | P |
| Result Row | 0 | 6 | 6 | 7 |
| Submit | | | | |

Fig. 4. GUI interaction (fully-functional prototype)

5.2 Implementing the Oracle

It may not be necessary nor pedagogically effective to forward all questions induced by algorithmic debugging to students. Moreover, some learners will hardly be patient enough to go through many iterations of system-learner interactions once they have already given an answer to a given problem. To address this issue, we have built an oracle that can answer questions about the subtraction

task: it takes a learner's *full* answer from the exercise sheet, and extracts from it those parts required to answer a given question. Fig. 5 depicts a code fragment to handle cases for the top clause `subtract/2` and the clause `increment/2`.

```

oracle( Step, YesNo ) :-
    get_student_answer_from_gui( Answer ), !,
    oracle_helper( Step, Answer, YesNo ).

oracle_helper( subtract(Col, _Input, Output), Answer, YesNo ) :-
    length(Answer, NumberColumns),
    Col = NumberColumns, !,
    (
        subsumes_term(Output, Answer) -> YesNo = yes ; YesNo = no
    ).

oracle_helper( subtract( Col, Input, Output ), Answer, YesNo ) :-
    allbutlast(Answer, AnswerRed),
    oracle_helper( subtract(Col, Input, Output), AnswerRed, YesNo ).

oracle_helper(increment(Col,_Input,(M2, S2, R2)),Answer,YesNo ) :-
    length(Answer, NumberColumns),
    Col = NumberColumns, !,
    butlast( Answer, ( _MS, SS, _RS ) ),
    ( S2 == SS -> YesNo = yes ; YesNo = no ).

oracle_helper( increment(Col,InputCol,OutputCol),Answer,YesNo ) :-
    allbutlast(Answer, AnswerRed),
    oracle_helper(increment(Col,InputCol,OutputCol),AnswerRed,YesNo).

?- get_diagnosis([(3,1,S1),(2,7,S2)], [(3,1,2), (12,7,5)], Diagnosis).
==> irreducible disagreement: increment(2, (3,1,_G3659), (3,2,_G3659))

```

Fig. 5. Implementing the Oracle (fragment for subtraction task)

The clause `oracle/2` is called with the current step executed by the expert model and outputs `yes` or `no`, depending on whether the learner agrees or disagrees with the expert step. The new argument `Col` marks the column currently processed. It is compared to the length of `Answer`, initially capturing the full sum given by the learner. When they are equal, we extract from `Answer` the relevant parts and check them for correctness with respect to the expert answer; otherwise, we recurse with the learner's answer reduced by the one column no longer of interest.

6 Discussion and Related Work

In Fig. 2 we have given a small selection of typical errors in subtraction. Our use of algorithmic debugging alone will surely fail to identify the full nature of

learners' erroneous behaviour, but algorithmic debugging in combination with other techniques will get closer to this (see Future Work).

Our adaptation and use of algorithmic debugging in tutoring is similar in spirit to model tracing. While our discussion assumed the learner to give a full answer to a given subtraction task, we can instrument our algorithm to provide next-step help in case where learners only submitted a partial sum. Note also that our approach does not require any representation of buggy skills. The interaction induced by algorithmic debugging compares learner actions to those of the expert model only – admittedly at the cost of providing less effective remedial feedback once a disagreement has been identified. On the positive side, our expert model is *just* an executable Prolog program. While it encodes skills that we want the learner to acquire, it contains no (other) tutoring-specific information. The power and simplicity of our approach is due to the meta-programming perspective.

There is only little research in the ITS community that builds upon logic programming techniques. Thirty years ago, Self described student modelling in terms of an induction problem, where student observations can be seen as the I/O behaviour of an unknown program that needs to be induced from such behaviour [8]. An induction logic programming approach to cognitive diagnosis has been taken by [5]. Here, expert knowledge is represented as a set of Prolog clauses, and Shapiro's Model Inference System (MIS) [9] is used to synthesise the student model from expert knowledge and student answers. Once the student model, a set of Prolog clauses, is constructed, Shapiro's Program Diagnosis System (PDS), based upon algorithmic debugging, is used to identify students' misconceptions, that is, the bugs in the MIS-constructed Prolog program.

In [1], Beller & Hoppe use a fail-safe meta-interpreter to identify student error. A Prolog program, modelling the expert knowledge for doing subtraction, is executed by instantiating its output parameter with the student answer. While standard Prolog interpretation would fail, a fail-safe meta-interpreter can recover from execution failure, and can also return an execution trace. Beller & Hoppe formulate *error patterns* which are then matched against the execution trace, and where each successful match is indicating a plausible student bug.

7 Conclusion and Future Work

There is good evidence that a sound methodology of cognitive diagnosis in intelligent tutoring can be realised in the framework of logic programming. Its declarative aspect, its view that program equals data, the substantial work in areas such as meta-level interpreters, partial evaluation, reasoning about programs, algorithmic debugging and inductive logic programming shows that there is ample potential to harness such techniques to support cognitive diagnosis in the context of intelligent tutoring systems. In this paper, we reported on our work to use algorithmic debugging to advance cognitive diagnosis in this direction.

In the future, we aim at systematically studying, adapting and combining various logic programming methods to effectively perform cognitive diagnosis in educational settings. We are currently working on a combination of algorithmic

debugging and program transformation. Once algorithmic debugging identifies an irreducible disagreement between expert behaviour and the learner, we induce code perturbations (*i.e.* bugs) into the expert program to make the disagreement disappear. After a series of applications of algorithmic debugging and code perturbations, we obtain a buggy procedure which models the learner's erroneous behaviour. In a related strand, we would like to adapt and use inductive program synthesis techniques; those, for instance, will be necessary to cover program transformations where novel elements need to be constructed. Once an arsenal of such techniques is developed, we seek to characterise the types of errors that can be diagnosed with each method or combination of methods, and to determine the effectiveness of remedial feedback based on such diagnosis.

Acknowledgements. The idea of turning Shapiro's method for algorithmic debugging on its head to support the diagnosis of student errors originated from Alan Smaill and Alan Bundy (both University of Edinburgh); personal communication (Note 1396).

Many thanks to the reviewers whose comments helped improve the paper.

References

1. Beller, S., Hoppe, U.: Deductive error reconstruction and classification in a logic programming framework. In: Brna, P., Ohlsson, S., Pain, H. (eds.) Proc. of the World Conference on Artificial Intelligence in Education, pp. 433–440 (1993)
2. Brown, J.S., Burton, R.R.: Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science* 2, 155–192 (1978)
3. Burton, R.R.: Debuggy: Diagnosis of errors in basic mathematical skills. In: Sherman, D., Brown, J.S. (eds.) Intelligent Tutoring Systems. Academic Press, London (1982)
4. Corbett, A.T., Anderson, J.R., Patterson, E.J.: Problem compilation and tutoring flexibility in the lisp tutor. In: International Conference of Intelligent Tutoring Systems, Montreal (1988)
5. Kawai, K., Mizoguchi, R., Kakusho, O., Toyoda, J.: A framework for ICAI systems based on inductive inference and logic programming. *New Generation Computing* 5, 115–129 (1987)
6. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A.: Intelligent tutoring goes to school in the big city. *Journal of Artificial Intelligence in Education* 8(1), 30–43 (1997)
7. Reiser, B.J., Anderson, J.R., Farrell, R.G.: Dynamic student modelling in an intelligent tutor for lisp programming. In: IJCAI 1985: Proceedings of the 9th International Joint Conference on Artificial Intelligence, pp. 8–14. Morgan Kaufmann Publishers Inc., San Francisco (1985)
8. Self, J.: Student models and artificial intelligence. *Computers & Education* 3, 309–312 (1979)
9. Shapiro, E.Y.: Algorithmic Program Debugging. ACM Distinguished Dissertations. MIT Press (1983); Thesis (Ph.D.) – Yale University (1982)

Author Index

- Abolmaali, Nasreddin 111
Adolphs, Peter 38, 50
Adrian, Benjamin 134
Ahmadi, Babak 122
Airola, Antti 87
Albrecht, Sven 99

Beetz, Michael 144
Beierle, Christoph 63, 157
Benz, Anton 38
Bertomeu Castelló, Núria 38
Biedert, Ralf 134
Biundo, Susanne 216
Böhme, Hans-Joachim 111

Cesta, Amedeo 233
Cheng, Xiwen 38
Christensen, Andrea 75, 330

Dengel, Andreas 134, 260
Duemcke, Martina 325

Eisenbach, Markus 277
Endres, Dominik 75, 228, 330

Finthammer, Marc 63
Finzi, Alberto 233
Fratini, Simone 233

Giese, Martin A. 75, 228, 330
Gieseke, Fabian 87, 169
Gross, Horst-Michael 277
Günther, Martin 99
Gust, Helmar 289

Haase, Robert 111
Hadiji, Fabian 122
Hees, Jörn 134
Hein, Albert 139
Hertzberg, Joachim 99
Herzog, Otthein 325
Hoffmann, Jörg 1

Jain, Dominik 144, 191
Janning, Ruth 157

Kern-Isberner, Gabriele 63
Kersting, Kristian 122
Kirste, Thomas 139
Klein, Andreas 255
Klüwer, Tina 38
Kolarow, Alexander 277
Kolesnik, Marina 228
Kramer, Oliver 87, 169, 179
Krifka, Manfred 38
Krumnack, Ulf 289
Kühnberger, Kai-Uwe 289

Lässig, Jörg 179
Li, Hong 50

Maier, Paul 191
Matuszak, Michał 204
Miękisz, Jacek 204
Minaei, Behrouz 246
Muensch, Alexander 347
Müller, Felix 216

Oberhoff, Daniel 228
Omlor, Lars 75
Orlandini, Andrea 233

Pahikkala, Tapani 87
Palm, Günther 335
Parvin, Hamid 246
Parvin, Sajad 246

Ragni, Marco 255
Reif, Matthias 260
Reithinger, Norbert 272
Roller, Roland 272
Roth-Berghofer, Thomas 134

Sachenbacher, Martin 191
Satzger, Benjamin 179
Scheffler, Tatjana 272
Schenk, Konrad 277
Schmidt, Martin 289
Schon, Claudia 301
Schreiber, Tomasz 204
Schuldt, Arne 313

- Shafait, Faisal 260
Steels, Luc 14
Stommel, Martin 325
Strelakova, Alexandra 38

Taubert, Nick 330
Thielscher, Michael 26
Thimm, Matthias 63
Tokic, Michel 335

Uszkoreit, Hans 38, 50
- von Gleissenthall, Klaus 144

Wiemann, Thomas 99

Xu, Feiyu 38, 50

Zenker, Bjoern 347
Zinn, Claus 357
Zips, Daniel 111