THIRD BIENNIAL REPORT

March 1995 – April 1997

Contents

Part I

Overview

Research Programme

The institute is devoted to basic research in computer science, and in particular to the study of complex computer systems.

Complexity in computer systems arises for various reasons: A problem can be complex due to huge masses of data that have to be handled, sometimes in real time. For this sort of problem efficient algorithms and data structures as well as parallel processing are of great importance. Parallel algorithms are often designed for theoretical machines which abstract from the actual communication between their processors in one way or another. The problem of how we should build such machines is still unsolved.

Or complexity can mean logical complexity as we find it in large software systems, with many layers of abstraction, where applications from different problem domains interact with each other in often unpredictable ways. Here we need to apply methods based on mathematical logic in order to structure, reason about, and develop more systematically, such large systems.

Today's computer systems are very complex in the way that they consist of a large number of hardware components, both discrete and analogue, which operate concurrently, which are physically distributed, often in a non-local manner, and which are embedded into a natural environment that is governed by physical laws. We want to understand better the nature of such systems, how to develop them such that they behave as wanted and to ensure that they are, by the way they are constructed, insensitive to faults in their components.

Computer systems are more and more used to realize and simulate parts of the real or an imaginary world. Such simulations have to deal with all of the forms of complexity we have mentioned above.

Structure

The institute is planned to consist eventually of five research groups, to fit this research programme. Since it opened in Dec. 1990 two research units have been installed. The "Algorithms and Complexity" group is headed by K. Mehlhorn, and the "Programming Logics" group is headed by H. Ganzinger. In addition to these two, a research group at the University of Potsdam, headed by M. Gössel, was associated with the institute until the end of February 1997. This group has investigated fault tolerant computing.

Potential areas that have been identified for the remaining three units include concurrent and distributed systems, embedded systems, computer networks, graphics, robotics, computer vision, and neural computing.

At present 18 research associates, 28 doctorate students and 17 postdocs are affiliated with the institute. The scientific staff is complemented by an administrative unit with 15 members (including secretaries), by a computing support unit (5 members of staff) and by our library (2 members of staff). The computing support unit currently operates a network of approximately 200 workstations.

Grants

The institute is involved in a number of projects related to research grants awarded by the European Union, by the German Science Foundation (DFG) and by the German Ministry for Education and Research (BMBF); for the descriptions of these grants see sections II.15 and III.14. Funding of these projects in 1996 was about 566.000 DM.

Results

In the two parts to follow we describe in detail, for the two research groups, the research programmes and results obtained in the period March 1995 through April 1997. We have continued to be very successful in our research, as documented by our many scientific publications, including about 240 articles in journals, books or proceedings of major international conferences.

Many of the institute's results are, in addition, available to the public through computer programs such as the LEDA library of efficient algorithms, and the SPASS theorem prover for first-order logic.

Teaching activities

The institute makes an effort to offer a variety of courses to computer science students of the Universität des Saarlandes. Courses taught during the period of this report are listed in Sections II.12 and III.12. In the same period 8 doctoral dissertations and 1 habilitation have been successfully completed.

Professional activities

Members of the institute have been involved in the organization of 11 workshops and conferences. In 30 cases we have been invited to join the program committee of major international conferences, not counting program committee memberships for national and international workshops. Finally, we serve on the editorial board of 13 scientific journals.

Professorships

The following members of the institute have accepted (tenured) professorships or equivalent research positions in 1995/96:

Sunil Arya; Assist.Prof., Hong Kong Univ. of Science and Technology
David Basin: Full Prof. (C4), University of Freiburg, Germany
Ramesh Hariharan: Assist.Prof., Indian Institute of Science, Bangalore, India
Hans Jürgen Ohlbach: Lecturer, Imperial College, London, England
Michiel Smid: Assoc.Prof (C3), University of Magdeburg, Germany

Part II

The Algorithms and Complexity Group

1 Personnel

Director:
Prof. Dr. Kurt Mehlhorn

Research Associates:
Dr. Susanne Albers
Dr. Christoph Burnikel
Dr. Shiva Chaudhuri
Dr. Rudolf Fleischer
Dr. Naveen Garg
Priv.-Doz. Dr. Torben Hagerup
Dr. Hans-Peter Lenhof
Dr. Petra Mutzel
Dr. Christine Rüb
Dr. Stefan Schirra
Dr. Jop F. Sibeyn
Dr. Michiel Smid (until January 1996; Jan 1996 - Nov 1996 King's College, London; now Assoc. Prof. at the University of Magdeburg, Germany)
Dr. Jesper Larsson Träff (since March 1995)
Dr. Christian Uhrig (until January 1996; now at LEDA Software GmbH)
Dr. Christos Zaroliagis

Postdocs:
Dr. Srinivasa Arikati (until February 1996; now in Cadence Design Systems, San Jose, CA, USA)
Dr. Sunil Arya (until August 1995; now Assist.Prof. at The Hong Kong University of Science and Technology)
Dr. Phil Bradford (until September 1996)
Dr. Gerth Stølting Brodal (since February 1997; previously at the University of Aarhus, Denmark)
Dr. Vasilis Capoyleas (until June 1996)
Dr. Ka Wong Chong (since September 1996; previously at the University of Hong Kong)
Dr. Leszek Gąsieniec (since November 1995; previously at Université de Québec à Hull, Canada)
Dr. Dimitrios Gunopulos (September 1995 until August 1996; now at IBM Almaden Research Center, San Jose, CA, USA)
Dr. Prosenjit Gupta (October 1995 until September 1996; now in Bell Laboratories, Murray Hill, New Jersey, USA)
Dr. Ramesh Hariharan (until December 1995; now Assist.Prof. at the Indian Institute of Science, Bangalore, India)
Dr. Pierre Kelsen (until August 1996)
Dr. Sanjeev Mahajan (until February 1997)
Dr. Marina Papatriantafilou (since October 1996)
Dr. Edgar Ramos (since October 1996; previously at Rutgers University, Piscataway, New Jersey, USA)
Dr. Sanjeev Saluja (until October 1996; now in LSI Logic Corporation, Milpitas, California, USA)
Dr. Philippas Tsigas
Dr. Yaoguang Wang (since June 1996; previously at the University of Waterloo, Canada)

PhD students:
Hannah Bast
Andreas Crauser (since July 1996)
Ulrich Finkler
Jordan Gergov
Christoph Hundack (since December 1995)
Oliver Kohlbacher (since April 1996)
Ulrich Meyer (since February 1997)
Peter Müller (since April 1995)
Volker Priebe
Knut Reinert
Thomas Schilz
Erik Schwarzenecker (until September 1995)
Michael Seel (since July 1995)
Christian Thiel (until January 1996)
Thomas Ziegler (since February 1997)

Secretaries:
Andrea Eßer (until September 1996)
Ingrid Finkler
Evelyn Haak (since July 1995)
Martina Horn

2 Executive Summary

The "Algorithms and Complexity" group

- investigates the inherent complexity of computational problems,
- designs and analyses efficient algorithms for fundamental combinatorial and geometric problems (both for sequential and parallel machines),
- develops software libraries to bridge the gap between algorithms research and the use of algorithms, and
- solves algorithmic problems in specific domains, such as computational biology and graph visualization.

We have managed, quite successfully, to intertwine theory with practice; in fact, frequently the same researchers are involved in both. Our theoretical research has led to numerous publications in reputed conferences and journals (see Section 16) and the practical work has resulted in widely used commercial-quality software. We are also very active in teaching.

We have maintained our activities in computational geometry, parallel and distributed algorithms, data structures and graph algorithms, computational complexity, and the construction of software libraries. We have further increased our strength in combinatorial optimization and we have built up groups in graph drawing and in computational biology and chemistry. We now briefly survey our results in each of the areas and highlight our most important results. We also list the researchers, postdocs, and PhD-students that are currently (May 1st, 1997) working in the area and the fraction of their time that is devoted to the area. In case of outside funding this is also indicated[1]. We use different organizational schemes for the different areas: tightly coordinated groups in all areas that involve significant amounts of implementation and loosely connected sets of researchers in the other areas. In the former case we will also name the coordinator(s) of the area.

Complexity and Lower Bounds: This is the smallest of our research themes[2]. We have worked on circuit complexity, the randomized complexity of maintaining the minimum, the complexity of acyclic logic programs (in a cooperation with AG2), and the complexity of searching in monotone matrices.

The most important result is the invention of the method of so-called deterministic restrictions by Shiva Chaudhuri to prove lower bounds for circuits. It allows to separate circuits of size n from circuits of size n^2. Let AC^0 denote the set of functions computable by uniform polynomial size circuits of constant depth and let LC^0 denote the set of functions computable by uniform *linear* size circuits of constant depth. While LC^0 appears to be a severely restricted class, it turns out to contain surprisingly complex functions. Deterministic restrictions allow to show that LC^0 is properly contained in AC^0.

Research associates: Chaudhuri (1/2), Fleischer (1/4, DFG-Habilitationsstipendium)
Postdoc: Brodal (1/4, Carlsberg-foundation)

---

[1] ALCOM-IT and CGAL are projects financed by the European Union, SFB stands for Sonderforschungbereich "VLSI und Parallelität", DFG stands for Deutsche Forschungsgemeinschaft (German Research Foundation), BMBF stands for Ministry of Education and Research, and Graduiertenkolleg stands for the special PhD program financed by the DFG.

[2] We have attempted to enlarge it by attracting a postdoc; our attempts were unsuccessful so far.

Combinatorial Algorithms: Combinatorial algorithms continues to be an active area of research in our group spanning a wide range of topics. Over the past two years we have worked on sorting, graph and network algorithms, probabilistic analysis of algorithms, pattern matching, and online algorithms. Among the many results obtained we want to emphasize two.

Torben Hagerup and co-workers considered the problem of sorting sequentially n arbitrary integers. They developed a deterministic algorithm that runs in $O(n \log\log n)$ time on a RAM with standard arithmetic and bitwise Boolean operations and thus made considerable progress towards settling the question whether sorting is possible in linear time.

Susanne Albers developed an improved algorithm for Graham's m-machine scheduling problem, which is one of the most basic scheduling problems. The main contribution of her paper is not only the improvement in the competitive ratios, but the new scheduling strategy underlying her solution. At any time, the new algorithm maintains a set S_1 of $\lfloor \frac{m}{2} \rfloor$ machines with a low load and a set S_2 of $\lceil \frac{m}{2} \rceil$ machines with a high load. Every job is either scheduled on the least loaded machine in S_1 or on the least loaded machine in S_2. The decision, which of the two machines to choose, depends on the *ratio* of the load on machines in S_1 to the load on machines in S_2.

Research Associates: Albers (1), Chaudhuri (1/2), Fleischer (3/4, DFG-Habilitationsstipendium), Hagerup (3/4), Mehlhorn (1/4), Zaroliagis (1/2)
Postdocs: Brodal (3/4, Carlsberg-Foundation), Gąsieniec (1/2)
PhD-students: Bast (1/2, Graduiertenkolleg), Crauser (1), Meyer (1/2, Graduiertenkolleg), Priebe (1)

Parallel and Distributed Computing: Our work in parallel and distributed computing over the past two years has been extensive, and many members of the group have contributed to these areas. Our work falls roughly into the following categories: PRAM algorithms for fundamental graph and network problems like shortest paths, minimum spanning tree and network flow; parallel data structures, in particular parallel priority queues, and general techniques, designed with the PRAM as a basis, but with applicability also in other settings; alternatives to and variations of the PRAM as a model of parallel computation; algorithms for basic problems, like routing, sorting and list-ranking, for networks of interconnected computers; general-purpose, model-independent techniques for parallel computation (dynamic load balancing and scheduling); and distributed computing[3] (interprocess communication and coordination, synchronization, resource management).

We give details on the work on scheduling. The following scheduling problem arises frequently in parallel computing. Given are n independent tasks whose execution times are only approximately known and which can be scheduled in any order on a set of p processors. The tasks are scheduled in batches. The execution time of a batch of size m is modeled as the sum of a fixed overhead and m independent random variables with a common fixed probability distribution. Torben Hagerup studied the performance of most previously suggested strategies together with a new *Bold* strategy in a common experimental framework. His experiments, carried out for a spectrum of parameter settings as representative as possible, revealed major weaknesses of each of the previously known strategies for certain parameter patterns, so that their general and unquestioned use does not appear advisable. The new Bold strategy, on the other hand, performed well throughout, and in fact no other strategy ever achieved a significantly lower (estimate of the) expected finishing time. In a subsequent paper Hannah Bast was able to give a partial theoretical underpinning to these experimental findings. In particular, she introduced a simple *Balancing* strategy that provably always achieves minimal expected finishing time (up to a small constant factor), except when the

---

[3] The work on distributed computing will not be continued as the researchers in the area have accepted positions elsewhere.

number of tasks is small, in which case the trivial strategies do sufficiently well. As she was able to keep her analysis on a very abstract and general level, many of the previous strategies became analyzable as well.

Research Associates: Hagerup (1/4), Sibeyn (1, SFB), Träff (1/2, SFB), Rüb (1/2), Zaroliagis (1/2)
Postdocs: Chong (1), Gąsieniec (1/2), Papatriantafilou (1), Tsigas (1)
PhD-students: Bast (1/2, Graduiertenkolleg), Gergov (1/2), Meyer (1/2, Graduiertenkolleg)

Computational Geometry: We continued our work both on rather theoretical core problems in geometry and rather applied problems motivated by actual problems in application areas like robotics, geographical information systems, and computer aided manufacturing.

We discuss Edgar Ramos' work on geometric divide-and-conquer algorithms. Ramos (together with Mike Goodrich from John Hopkins) found time and work optimal deterministic parallel algorithms for computing near-optimal size ϵ-nets and ϵ-approximations, the basic ingredient for most geometric divide-and-conquer algorithms. He also showed how to use these sampling techniques to obtain output-sensitive algorithms for 1-dimensional lower envelopes and 2-dimensional Voronoi diagrams.

Research Associates: Burnikel (1/2, CGAL), Mehlhorn (1/4), Schirra (1/2, CGAL)
Postdocs: Ramos (1)
PhD-students: Seel (1/2)

Combinatorial Optimization and Applications: The activity of our group in combinatorial optimization has increased considerably during the last two years. Our research mainly focuses on approximation algorithms and on practical problem solving using branch-and-cut algorithms, including the necessary theoretical polyhedral investigations. We are working on applications in Computational Biology and Physics and in Automatic Graph Drawing. We report about the former in the next paragraph.

In approximation algorithms our most important results are improved approximation algorithms for maximum planar subgraphs (Uli Finkler et al), for minimizing weighted completion time for single machine scheduling with release dates (Yaoguang Wang), and for computing sparse cuts in undirected graphs (Naveen Garg).

We also investigated the planar subgraph problem from a polyhedral point of view (Petra Mutzel). This led to a branch-and-cut algorithm for maximum planar subgraphs.

In graph drawing we have begun a major implementation effort that is directed by Petra Mutzel. The goal is, in cooperation with colleagues in Halle, Cologne and Passau, to develop the so-called AGD (Algorithms for Graph Drawing) Library.

Research Associates: Garg (1, ALCOM-IT), Mutzel (3/4)
Postdocs: Wang (1)
PhD-students: Gergov (1/2), Finkler (1/2, Graduiertenkolleg), Hundack (1, Graduiertenkolleg), Ziegler (1, Siemens), NN (1, BMBF)

Computational Biology, Chemistry and Physics: Our activity in this area has considerably expanded over the last two years. Two years ago only Hans-Peter Lenhof has worked in the area, we now have close to five persons working in the area. The work is directed by Hans-Peter Lenhof.

We work on four topics: multiple sequence alignment, the computation of ground states for Ising spin glasses, protein docking, and molecular dynamics simulations for synthetic polymers. On the first two topics there is a strong interaction with our researchers in combinatorial optimization (as the developed algorithms use integer linear programming formulations and the branch-and-cut paradigm) and on the last topic there is interaction with our researchers in parallel algorithms (as the simulation algorithm is parallel).

We give more details on the Protein Docking Problem. The goal of protein docking research is the development and implementation of algorithms that enable the user to predict reactions between proteins and to compute the three-dimensional structure of the resulting protein complexes. Protein docking algorithms can be used, for example, to screen protein databases for possible inhibitors of a given virus enzyme. The main result of our protein docking research was the development and implementation of new contact measures for judging possible docking conformations. We have tested the contact measures with "real world" docking examples. The quality of the docking results is very good. To be more precise, there is only one other docking algorithm that yields results of similar quality; that algorithm, however, needs more running time.

Research Associates: Lenhof (1), Mutzel (1/4), Rüb (1/2)
PhD-students: Kohlbacher (1), Müller (1, DFG), Reinert (1)

Software Libraries and Supporting Theory: We have continued our work on the LEDA platform of combinatorial and geometric computing (this work is coordinated by Stefan Schirra and Kurt Mehlhorn) and we started the PAD project of building a library of PRAM algorithms. In the PAD-project we aim at showing that many non-trivial algorithms can be made efficient in a practical setting where both memory size and number of processors are limited and we investigate the language features required for expressing complex parallel algorithms. *All work on LEDA is joint work with Stefan Näher's group in Halle.* LEDA is now used at about 1500 academic sites. A commercial version of LEDA is marketed by the spin-off company LEDA Software GmbH. The license holders are mainly in the telecommunication industry, the graphics industry, and the automotive industry. In the past two years we have considerably expanded the functionality of LEDA, in particular, in the area of computational geometry. On the theoretical side we advanced in three respects: We have developed program checkers for many geometric and combinatorial problems, we have started an investigation of running time prediction of real programs, and we have extended the range of efficient exact arithmetic to a subset of the algebraic numbers.

The latter advance by Fleischer et al is probably the most important. It allows us to implement geometric algorithms such as Voronoi diagrams of line segments and parametric search that nobody else can implement.

Research Associates: Burnikel (1/2, CGAL), Mehlhorn (1/2), Schirra (1/2, CGAL), Träff (1/2, SFB)
PhD-students: Finkler (1/2, Graduiertenkolleg), Seel (1/2, ALCOM-IT)

Michiel Smid completed his Habilitation in 1995, he is now associate professor of Computer Science at the University of Magdeburg. The Habilitation's procedure of Shiva Chaudhuri is ongoing. Three graduate students (C. Burnikel, E. Schwarzenecker, and C. Thiel) completed their PhD work and twelve PhD students are currently working in the group. Section 13 surveys their topics and expected completion dates.

Most of the researchers in the group are on two-year postdoc contracts or three-year graduate student scholarships and hence there is considerable fluctuation. We run an intensive seminar and lecture program to spur interaction within the group. We have two 35 minute "noon seminars" per week, which every group member has to attend, and we run two to four week "mini-courses" on "hot" topics, see Section 12 for more details. We run an intensive visitor program; more than 80 guests visited our group for stays up to three months, cf. Section 10 for details.

The group contributes to the master's program in Computer Science at the Universität des Saarlandes. We have offered more than thirty courses and seminars in the past two years, cf. Section 12 for details.

The group is involved in several national and international research projects. Section 15 gives details. We find it particularly pleasing that several of our research associates have won own research grants from the DFG.

Our scientific work is discussed in more detail in Sections 3 through 9. To be easily distinguishable, references regarding work done by members of the group (during the period considered in this report) are marked with a • .

3 Complexity and Lower Bounds

Although the activity of our group in this area is smaller compared to the rest, a number of interesting results have been obtained. The problem of proving *any* superlinear lower bound for *any* problem in NP on *any* reasonable model of computation (i.e., one that is comparable to a Turing machine) remains open. Two such models are Boolean circuits and the Cell Probe model of Yao. A major part of our work deals with proving lower bounds in these two models.

One of the few techniques known for proving lower bounds on Boolean circuit size is *random restrictions.* While several proofs using random restrictions have appeared in the literature, it is still not well understood why the technique works. While attempting to understand this technique, we developed a new technique, called deterministic restrictions, which in certain instances performs better than random restrictions. This technique has immediate applications, described in Section 3.1. Perhaps even more significant is the possibility that this technique can be used to give an alternative proof of the Switching Lemma of Håstad. Work on this question is in progress.

In the cell probe model, it is likely that sorting has a superlinear lower bound. However, we are far from being able to prove this. A related problem for which it is probably easier to prove a lower bound is the problem of maintaining a *priority queue.* However, even in the comparison model, the only lower bound known for maintaining a priority queue was via a reduction to sorting. We have developed a new proof for the comparison model (Section 3.2), which directly proves much stronger lower bounds for maintaining a priority queue than those obtained by a reduction to sorting. This helps to understand the nature of the problem and is a step towards proving a lower bound in the Cell Probe model.

The rest of our work deals with complexity problems that come from other areas. In joint work with a member of AG2, the parallel complexity of some problems in logic programming was studied. We have shown that computing the stable model of an acyclic logic program is P-complete (Section 3.3).

Applications often give rise to matrices with special properties, which allow efficient computations on these matrices. *Monotone* and *totally monotone* matrices are two such classes of matrices. In Section 3.4, a nontrivial lower bound on the number of comparisons and the number of accesses required to compute the row minima of a totally monotone matrix is described. An efficient parallel algorithm for this problem is also reported.

3.1 Deterministic Restrictions in Circuit Complexity

Investigators: Shiva Chaudhuri

One of the most successful techniques for proving lower bounds on the size of circuits is the method of *random restrictions* [3, 7, 4]. In [1], we have developed a new technique, called *deterministic restrictions*, that yields lower bounds in some situations when random restrictions fail.

Using the random restriction technique, it can be shown that a circuit with S gates of depth d can be made to output a constant by setting the values of $n - n/(O(\log S))^d$ inputs. Thus, any polynomial size circuit can be made to output a constant by setting $n - n/polylog(n)$ inputs. This yields a superpolynomial size lower bound for a constant depth circuit computing any function that cannot be made constant by setting $n - n/polylog(n)$ input bits. However, it does not say anything about functions which *can* be made constant by setting this many bits. In [1], we prove that a circuit with S gates and depth d can be made to output a constant by setting $5S^{1-\epsilon(d)}$ input values, where $\epsilon(d) = 1/4^d$. Thus, for appropriate constants α, β, a circuit of constant depth with $O(n^{1+\alpha})$ gates can be made to output a constant by setting $O(n^{1-\beta})$ input values. This yields a superlinear lower bound for any constant depth circuit computing a function that cannot be made constant by setting $O(n^{1-\beta})$ inputs. In other words, this allows us to prove a superlinear lower bound for a much larger class of functions.

Håstad showed that PARITY cannot be computed by a circuit of polynomial size and depth d for any fixed d [4]. Since any Boolean function can be computed by a depth 2 circuit of exponential size, the result can be interpreted as saying that exponential size circuits are strictly more powerful than polynomial, for constant depth. We would also like to see such a separation between polynomial sizes of different degrees, i.e., between n^k and n^{k+1}. However, nothing is known about this. A special case of this question is to separate circuits of size n from circuits of size n^2. AC^0 is the set of functions computable by uniform polynomial size circuits of constant depth. LC^0 is the set of functions computable by uniform *linear* size circuits of constant depth [5, 6]. While LC^0 appears to be a severely restricted class, it turns out to contain surprisingly complex functions. An open question posed in [5, 6] is whether LC^0 is properly contained in AC^0. Using the deterministic restrictions technique, we answer this question in the affirmative [1].

The threshold function T_r^n assumes the value 1 iff at least r of its input bits have value 1. The complexity of computing threshold functions has been widely studied. However, for small values of r, there is a large gap between the upper and lower bounds. In particular, the lower bounds do not even depend on n. In [1], we show that any circuit computing T_r^n, for small r, must have $\Omega(r^2(\log n)/\log r)$ gates. The lower bound holds for circuits without any restriction on depth, and considerably improves the previous bounds.

The lower bounds for threshold functions are based on a combinatorial lemma of independent interest. A family of sets is k-cover-free if no set is contained in the union of k others. Suppose we are given a k-cover-free system of n subsets of a ground set X. What is the minimum possible cardinality of X? Erdős, Frankl and Füredi [2] showed that $|X| \geq \Omega(k \log n)$. In [1], we improve this bound and show that $|X| \geq \Omega(k^2 \log n/\log k)$.

References

- [1] S. Chaudhuri and J. Radhakrishnan. Deterministic restrictions in circuit complexity. In *Proc. 28th ACM Symp. on Theory of Computing (STOC)*, pages 30–37, 1996.

[2] P. Erdős, P. Frankl, and Z. Füredi. Families of finite sets in which no set is covered by the union of r others. *Israel Journal of Mathematics*, 51:79–89, 1985.

[3] M. Furst, J. Saxe, and M. Sipser. Parity, circuits and the polynomial time hierarchy. *Mathematical Systems Theory*, 17:13–27, 1984.

[4] J. Håstad. Almost optimal lower bounds for small depth circuits. In *Proc. 18th STOC*, pages 6–20, 1986.

[5] I. Newman, P. Ragde, and A. Wigderson. Perfect hashing, graph entropy and circuit complexity. In *Proc. 5th Ann. Conf. on Structure in Complexity Theory*, pages 91–99, 1990.

[6] P. Ragde and A. Wigderson. Linear-size constant-depth polylog-threshold circuits. *Information Processing Letters*, 39:143–146, 1991.

[7] A.C. Yao. Separating the polynomial -time hierarchy by oracles. In *Proc. of 26th FOCS*, pages 1–10, 1895.

3.2 The Randomized Complexity of Maintaining the Minimum

Investigators: Gerth Stølting Brodal and Shiva Chaudhuri

In [3] we consider the complexity of maintaining a set S of elements from a totally ordered universe under the operations: INSERT(x) which inserts the element x into S, DELETE(x) which removes from S the element x provided it is known where x is stored, and FINDMIN which returns the minimum element in S without removing it.

We refer to this problem as the INSERT-DELETE-FINDMIN problem. We denote the size of S by n. The analysis is done in the comparison model, i.e., the time required by the algorithm is the number of comparisons it makes. The input is a sequence of operations, given to the algorithm in an online manner, that is, the algorithm must process the current operation before it receives the next operation in the sequence.

Worst case asymptotic time bounds for some existing data structures supporting the above operations are: Doubly linked lists that support both INSERT and DELETE in worst case constant time and FINDMIN in worst case $O(n)$ time; and search trees and priority queues that support INSERT and DELETE in worst case $O(\log n)$ time and FINDMIN in constant time [4]. These examples suggest a tradeoff between the worst case times of the two update operations INSERT, DELETE and the query operation FINDMIN. In [3] we prove the following lower bound on this tradeoff: any randomized algorithm with expected amortized update time at most t requires expected time at least $(n/e2^t) - 1$ for FINDMIN. Thus, if the update operations have expected amortized constant cost, FINDMIN requires linear expected time. On the other hand if FINDMIN has constant expected time, then one of the update operations requires logarithmic expected amortized time. This shows that the mentioned data structures are optimal in the sense of the tradeoff, and they cannot be improved even by considering amortized cost and allowing randomization.

The above lower bound shows that it is hard to maintain the minimum. Is it any easier to maintain the rank of *some* element, not necessarily the minimum? We consider a weaker problem called INSERT-DELETE-FINDANY, which is defined exactly as the previous problem, except that FINDMIN is replaced by the weaker operation FINDANY that returns an element in S and its rank. FINDANY is not constrained to return the same element each time it is invoked or to return the element with the same rank. The only condition is that the rank returned should be the rank of the element returned. We give a randomized algorithm for the INSERT-DELETE-FINDANY problem with constant expected time per operation. Thus, this problem is strictly easier than INSERT-DELETE-FINDMIN, when randomization is allowed. However, we show that for deterministic algorithms, the two problems are essentially equally hard. We show that any deterministic algorithm with amortized update time at most t requires at least $n/2^{4t+3} - 1$ comparisons for some FINDANY operation. This lower bound is proved using an explicit adversary argument, similar to the one used by Borodin, Guibas, Lynch and Yao [2]. The adversary strategy is simple, yet surprisingly powerful. The same strategy may be used to obtain the well known $\Omega(n \log n)$ lower bound for sorting. An explicit adversary for sorting has previously been given by Atallah and Kosaraju [1].

The previous results show that maintaining any kind of rank information online is hard. However, if the sequence of instructions to be processed is known in advance, then one can do better. We give a deterministic algorithm for the offline INSERT-DELETE-FINDMIN problem which has an amortized cost per operation of at most three comparisons.

Our proofs use various averaging arguments which are used to derive general combinatorial properties of trees.

References

[1] M.J. Atallah and S. Rao Kosaraju. An adversary-based lower bound for sorting. *Information Processing Letters*, 13:55–57, 1981.

[2] A. Borodin, L. Guibas, N. Lynch, and A.C. Yao. Efficient searching using partial ordering. *Information Processing Letters*, 12:71–75, 1981.

• [3] G.S. Brodal, S. Chaudhuri, and J. Radhakrishnan. The randomized complexity of maintaining the minimum. *Nordic Journal of Computing*, 3(4):337–351, 1996. Special issue on SWAT'96.

[4] K. Mehlhorn and A. Tsakalidis. Data structures. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*. MIT Press/Elsevier, 1990.

3.3 The Parallel Complexity of Acyclic Logic Programs

Investigators: Shiva Chaudhuri and Christos Zaroliagis

The semantics of a general logic program (roughly the problem of what a general logic program should entail) has been an active research area over the last years within logic programming. A propositional (general) logic program is a set of rules of the form $A \leftarrow L_1, \ldots, L_n$, where A is an atom (called the *head* of the rule) and $L_1, \ldots, L_n$ are literals (called the *body* of the rule). A literal is a formula of the form B (called *positive literal*) or $\neg B$ (called *negative literal*), where B is an atom. A logic program is called *definite or positive*, if every literal in the body of its rules is positive. On the other hand, if every literal in the body of its rules is negative, then the logic program is called *negative*.

Due to the differences in the way negation is interpreted, several different semantics for general logic programs exist [3] (e.g., stable models, partial stable models, minimal models and the well-founded model semantics). While these semantics in general draw different inferences from a general logic program, there are two cases where these inferences coincide. The first is the case of definite logic programs, where the semantics are defined by the stable (or equivalently the minimal) model of the definite program. The second is the case of *acyclic* general logic programs, where their semantics are defined by their stable model. A general logic program P is acyclic if a certain directed graph G_P associated with it does not contain cycles.

In [2], we investigate the parallel complexity of computing the stable model of an acyclic general logic program. Our restriction to acyclic programs was a necessary one. The problem of deciding whether a general logic program has a stable model is NP-complete, even in the case of negative logic programs. On the other hand, although for every definite logic program there is a polynomial time algorithm that computes its stable (or minimal) model, the problem has been proved to be P-complete. Furthermore, though severely restricted, acyclic general logic programs are of particular practical importance [1].

In [2], we show that: (i) computing the stable model of a negative acyclic logic program is P-complete; (ii) computing the stable (or minimal) model of a definite acyclic logic program is also P-complete. The first result is based on an interesting property between a graph-theoretic concept (the kernel of a graph) and the stable model of an acyclic logic program.

References

[1] K. Apt and M. Bezem. Acyclic programs. *New Generation Computing*, 9:335–363, 1991.

• [2] S. Chaudhuri, Y. Dimopoulos, and C. Zaroliagis. On the parallel complexity of acyclic logic programs. *Parallel Processing Letters*, 6(2):223–230, 1996.

[3] J. Minker. An overview of nonmonotonic reasoning and logic programming. *Journal of Logic Programming*, 17, 1993.

3.4 The Complexity of Searching in Totally Monotone Matrices

Investigators: Phil Bradford, Rudolf Fleischer, Knut Reinert, and Michiel Smid

Let M be a $n \times n$ matrix with entries from a totally ordered set. The *row minima (maxima) problem* is to find for each row $i \in \{1, \ldots, n\}$ the index $\mathsf{mc}(i)$ of the leftmost column which contains a minimal (maximal) element of row i. M is *monotone* if $\mathsf{mc}(i) \leq \mathsf{mc}(j)$ for all $1 \leq i < j \leq n$. M is *totally monotone* if each of its 2×2 submatrices is monotone. Row minima problems on (totally) monotone matrices are well motivated due to their large host of applications like for example the prediction of RNA secondary structure [10], the all-farthest-neighbor problem [1, 8], the computation of maximal enclosed or minimal circumscribed k-gons [1], or the matrix-chain-ordering problem [6].

For arbitrary $n \times n$ matrices, the sequential time complexity of the row minima problem is $\Theta(n^2)$. For monotone matrices, Aggarwal *et al.* [1] gave an $O(n \log n)$ time algorithm and showed that this is asymptotically optimal. For totally monotone matrices, they gave an asymptotically optimal $O(n)$ time algorithm, widely known as the SMAWK algorithm. An exact analysis shows that the SMAWK algorithm takes at most $5n - 2\log n - 6$ comparisons [9] (an upper bound of $5n$ was proven in [10]). This illustrates that it is not necessary to generate all n^2 entries of a (totally) monotone matrix M, but rather it is sufficient to compute selected entries of M when they are needed. This is of particular importance if the generation of each matrix element cannot be done in constant time (for a discussion of this sort of phenomena see [8]).

In [7] we have addressed the question of exactly how many matrix elements we must generate to solve the row minima problem on a totally monotone matrix. We showed that any algorithm needs at least $3n - 7$ comparisons to solve the problem. We also gave a $4n - 12$ matrix access lower bound which means that any algorithm that solves the row minima problem on a totally monotone matrix must generate at least $4n - 12$ matrix elements.

Because of the importance of the row minima problem on totally monotone matrices as a subroutine for other problems, there has been a great effort to design time and work efficient parallel algorithms. This has turned out to be a difficult problem (the optimal SMAWK algorithm is inherently sequential). Only optimal randomized algorithms were known [11]. In [5] we gave a parallel deterministic algorithm with slightly better work bounds than previous algorithms (at the cost of slightly worse time bounds). The table below summarizes previous and our new results. For the CREW and EREW PRAM, the bottleneck of our algorithm is an integer sorting step, where the best known algorithms are due to Albers and Hagerup [3]. Faster algorithms would immediately improve our time bounds.

PRAM Model	Work	Time	Source
CRCW	$O(n\sqrt{\lg n})$	$O(\lg n \lg\lg n)$	[5]
CREW	$O(n\sqrt{\lg n})$	$O(\lg n (\lg\lg n)^2)$	[5]
EREW	$O(n \lg n / \lg\lg n)$	$O(\lg^2 n / \lg\lg n)$	[2, 11]
	$O(n\sqrt{\lg n \lg\lg n})$	$O(\lg n \sqrt{\lg n \lg\lg n})$	[5]
	$O(n \lg n)$	$O(\lg n)$	[4]

References

[1] A. Aggarwal, M.M. Klawe, S. Moran, P. Shor, and R. Wilber. Geometric applications of a matrix-searching algorithm. *Algorithmica*, 2:195–208, 1987.

[2] A. Aggarwal and J. Park. Notes on searching multidimensional monotone arrays. In *Proceedings of the 29st Symposium on the Foundations of Computer Science (FOCS'88)*, pages 497–512, 1988.

[3] S. Albers and T. Hagerup. Improved parallel integer sorting without concurrent writing. In *Proceedings of the 3rd ACM-SIAM Symposium on Discrete Algorithms (SODA'92)*, pages 463–472, 1992.

[4] M.J. Atallah and S.Rao Kosaraju. An efficient parallel algorithm for the row minima of a totally monotone matrix. *Journal of Algorithms*, 13:394–413, 1992.

• [5] P.G. Bradford, R. Fleischer, and M. Smid. More efficient parallel totally monotone matrix searching. *Journal of Algorithms*, to appear.

[6] P.G. Bradford, G.J.E. Rawlins, and G.E. Shannon. Efficient matrix chain ordering in polylog time with linear processors. In *Proceedings of the 8th IEEE International Parallel Processing Symposium (IPPS'94)*, pages 234–241, 1994.

• [7] P.G. Bradford and K. Reinert. Lower bounds for row minima searching. In *Proceedings of the 23rd International Colloqium on Automata, Languages and Programming (ICALP'96), Springer Lecture Notes in Computer Science 1099*, pages 454–465, 1996.

[8] J. Hershberger and S. Suri. Matrix searching with the shortest path metric. In *Proceedings of the 25th ACM Symposium on the Theory of Computation (STOC'93)*, pages 485–494, 1993.

[9] L.L. Larmore, 1995. Private communication.

[10] L.L. Larmore and B. Schieber. Online dynamic programming with applications to the prediction of RNA secondary structure. *Journal of Algorithms*, 12:490–515, 1991.

[11] R. Raman and U. Vishkin. Optimal randomized parallel algorithms for computing the row minima of a totally monotone matrix. In *Proceedings of the 5th ACM-SIAM Symposium on Discrete Algorithms (SODA'94)*, pages 613–621, 1994.

4 Combinatorial Algorithms

Combinatorial algorithms continue to be an active area of research in our group spanning a wide range of topics. Over the past two years we have concentrated on the following:

- Sorting
- Graph and network algorithms
- Probabilistic analysis of algorithms
- Pattern matching
- Online algorithms

Our work here is mostly of a theoretical nature. However, some of the results have already coupled with experimental work, and on-going efforts aim at bringing most of the other results into a more practical setting.

Within *Sorting* (Section 4.1), we considered the problem of sorting sequentially n arbitrary integers. We developed a deterministic algorithm that runs in $O(n \log\log n)$ time on a RAM with standard arithmetic and bitwise Boolean operations and thus made considerable progress towards settling the question whether sorting is possible in linear time. We have also worked on the constant factor analysis (so called meticulous analysis) of mergesort programs, and on the problem of matching nuts and bolts.

Within *Graph and network algorithms* (Section 4.2), we have studied a number of interesting problems which mainly involve the (efficient) computation of various quantities defined by input graphs and networks. In particular, we studied minimum cut problems and gave new optimal or near-optimal algorithms for the all-pairs minimum cut problem in directed sparse networks. An important consequence of the techniques used is a better algorithm for computing an s-t max-flow on a planar network. We studied fast data transmission in communication networks and gave new upper and lower bounds for sparse and general networks. We have investigated the problem of finding the maximum agreement subtree of two given trees, a problem with applications in computational biology, and have given an optimal algorithm for its solution.

Another line of research focused on graphs of bounded treewidth, which allow efficient solutions for many computational problems that are hard on general graphs. We extended the formalism of monadic second-order logic, which can express many efficiently solvable problems on graphs of small treewidth, from static to dynamic settings. Further, we conducted a comprehensive study on a graph parameter (thickness) which arises, for instance, in VLSI design and proved some new results for some minor-excluded classes of graphs.

Within *Probabilistic analysis of algorithms* (Section 4.3), we studied the average-case complexity of shortest-path problems on complete directed graphs; in particular, we investigated a reasonable probability distribution for graphs with arbitrary (as opposed to non-negative) edge lengths. Regarding derandomization of randomized algorithms, we investigated the fourth moment of a probability space proposed by Luby and applied our results to analyze random edge cuts in weighted graphs.

Within *Pattern matching* (Section 4.4), we examined a large number of problems in stringology and developed, for instance, several algorithms for string matching on compressed text. Moreover, we presented the first linear time algorithm for approximate dictionary matching and investigated time-space trade-offs.

Within *Online algorithms* (Section 4.5), we continued and extended our work on self-organizing data structures and distributed data management. On-going research mostly concentrates on scheduling, bin packing and robot exploration problems. We developed an improved algorithm for Graham's m-machine scheduling problem, which is one of the most basic scheduling problems. Our algorithm does not only give a better worst-case performance than all previous solutions, but it is also based on a new scheduling strategy. Moreover, we presented a sub-exponential algorithm for a robot exploration problem posed by Deng and Papadimitriou that has further applications in computational learning theory. Further, we have started to build a library of online algorithms with emphasis on visualization and animation.

4.1 Sorting

Word Sorting

Investigator: Torben Hagerup

It has been known since the early days of computing that the complexity of sorting n items is $\Theta(n \log n)$. The usual lower bound, however, is valid only in the comparison-based setting, i.e., it assumes that the correct order of the input items can be deduced from the outcomes of the comparisons executed between pairs of items. This assumption can be a significant restriction, as evidenced by usual bucket sorting, which manages to sort n integers in the range $0 \mathinner{.\,.} n-1$ in $O(n)$ time without ever comparing two input items. Still, bucket sorting applies only to sorting small integers, so that it makes only a very small dent in our $\Theta(n \log n)$ view of the world.

The fact that bucket sorting can sort only integers may not be as restrictive as it first seems, since one can argue that "God made the integers, and nothing else has ever made it into a computer word." The requirement that the integers must be small, linear or (in light of radix sorting) polynomial in the input size n, on the other hand, is a true restriction, so that bucket sorting does not capture the essence of sorting. It seems natural to ask about the complexity of sorting larger integers. How large? Well, suppose that we work on a binary computer with a word length of w bits. Then the integers that we will want to sort are those that can be represented on the machine, i.e., those of (at most) w bits. If w is large, sorting problems are thus hard but, on the other hand, our primitives are also powerful, since we assume that standard RAM operations like addition of w-bit arguments take constant time.

In [2], we attempted to throw light on the following fundamental question, motivated by the discussion above: What is the complexity of sorting n w-bit integers on a w-bit RAM? The parameters n and w are independent of each other, except that we require $w \geq \log n$ to keep the problem meaningful. Note also that "no holds are barred"—an algorithm may compare two input keys but also, say, multiply them, if it feels so inclined. We refer to this problem as "word sorting." It is clear that the complexity of word sorting lies between $\Theta(n)$ (the obvious lower bound) and $\Theta(n \log n)$ (the obvious upper bound), and Fredman and Willard [4] startled the community by showing that $O\big(n\sqrt{\log n}\big)$ is possible. We demonstrated that two previously known results, put together in a suitable fashion, yield a simple algorithm for the word-sorting problem with a running time of $O(n \log\log n)$. (Incidentally, one of the two building blocks for the new algorithm was provided several years ago by members of our group [1].) A second result is a randomized word-sorting algorithm that works in $O(n)$ expected time under the assumption that the word size w

is sufficiently large relative to the input size n. A crucial ingredient in this algorithm is a new universal class of hash functions introduced in [3].

Our word-sorting algorithms, in their current formulation, are not practical, partly due to the small word sizes of present-day computers, compared to typical problem sizes. On the theoretical front, however, our results and methods have already had a significant impact. In one of the most interesting developments, Thorup [5, 6] considered the complexity of the single-source shortest-paths problem in the "word model" and designed algorithms that beat the $O(m + n \log n)$ bound of comparison-based algorithms (n = number of vertices, m = number of edges).

References

[1] S. Albers and T. Hagerup. Improved parallel integer sorting without concurrent writing. In *Proc. 3rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 463–472, 1992. Also *Inform. and Comput.*, to appear.

• [2] A. Andersson, T. Hagerup, S. Nilsson, and R. Raman. Sorting in linear time? In *Proc. 27th Annual ACM Symposium on Theory of Computing (STOC)*, pages 427–436, 1995.

• [3] M. Dietzfelbinger, T. Hagerup, J. Katajainen, and M. Penttonen. A reliable randomized algorithm for the closest-pair problem. *Journal of Algorithms*, to appear.

[4] M. L. Fredman and D. E. Willard. Surpassing the information theoretic bound with fusion trees. *J. Comput. System Sci.*, 47:424–436, 1993.

[5] M. Thorup. On RAM priority queues. In *Proc. 7th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 59–67, 1996.

[6] M. Thorup. Undirected single source shortest path in linear time. Technical Report DIKU–TR–97/4, Dept. of Computer Science, University of Copenhagen, Denmark, January 1997.

Meticulous Analysis of Mergesort Programs

Investigator: Jesper Larsson Träff

In contrast to the *asymptotical analysis* of algorithms the goal in the *meticulous analysis* is to determine also the constant factors, especially constants in the leading terms of the function expressing the running time. Meticulous analysis has previously been performed by (among many others) Knuth in his book on sorting [2].

The efficiency of various *mergesort programs* has been analysed under a simple unit-cost model, here termed the Pure C model. The model takes the costs of key comparisons, element moves and address calculations into account (but not processor caches, pipelining and other forms of processor parallelism). The goal has been to establish the best possible time-bounds relative to the model when sorting n integers. New implementations of two-way and four-way bottom-up mergesort have been given, the *worst-case* complexities of which are shown to be bounded by $5.5n \log_2 n + O(n)$ and $3.25n \log_2 n + O(n)$, respectively. In contrast, standard implementations of two-way mergesort in the Pure C model uses at least $7n \log_2 n + O(n)$ operations. The theoretical findings have been backed up by experiments which show the relevance of the model for analysing programs for internal-memory computations [1].

References

• [1] Jyrki Katajainen and Jesper Larsson Träff. A meticulous analysis of mergesort programs. In *Algorithms and Complexity. Third Italian Conference, CIAC'97*, volume 1203 of *LNCS*, pages 217–228, 1997.

[2] D.E. Knuth. *Searching and Sorting*, volume 3 of *The Art of Computer Programming.* Addison Wesley, 1973.

Matching Nuts and Bolts

Investigators: Phil Bradford and Rudolf Fleischer

In [5], Rawlins introduced the following problem (the *nuts-and-bolts-matching problem*) :

> Assume we are given a pile of n bolts of different sizes and a pile of n corresponding nuts. The problem is to find all pairs of corresponding nuts and bolts when we cannot compare nuts to nuts or bolts to bolts, i.e., we can only test whether the size of a given nut is smaller, equal or larger than the size of a given bolt.

Randomized Quicksort solves the problem in $O(n \log n)$ time, but it seems to be difficult to find good split elements for Quicksort deterministically in subquadratic time. The first subquadratic solution by Alon *et al.* [1] needs $\Theta(n \log^3 n)$ time for finding a good split element (and hence $\Theta(n \log^4 n)$ time for Quicksort). And since the algorithm is based on some strong expander graphs, there is a very large constant hidden in the O-notation.

In [2] we presented a simpler algorithm which finds a good split element in $O(n \log n)$ time. First, we connect the nuts and bolts with some expander graph of constant degree and do all the comparisons defined by the edges of this graph. Then we eliminate all the bolts which are only connected to larger (or smaller) nuts. Finally, we play a simple knockout tournament on the remaining bolts which guarantees that the winner of the tournament must be a good split element. Since each round of the tournament needs only linear time, we get in $O(n \log n)$ time a winner of the tournament which divides the pile of nuts (and bolts) into two piles, each of size $\frac{n}{100}$ at least. This gives a total running time for Quicksort of $O(n \log^2 n)$ (but also with quite a large constant).

Meanwhile, the existence of an $O(n \log n)$ time algorithm has been shown [3, 4], based on some properties of random graphs (so the algorithms can not be constructed explicitly).

References

[1] N. Alon, M. Blum, A. Fiat, S. Kannan, M. Naor, and R. Ostrovsky. Matching nuts and bolts. In *Proceedings of the 5th ACM-SIAM Symposium on Discrete Algorithms (SODA'94)*, pages 690–696, 1994.

• [2] N. Alon, P.G. Bradford, and R. Fleischer. Matching nuts and bolts faster. *Information Processing Letters*, 59:123–127, 1996.

• [3] P.G. Bradford. Matching nuts and bolts optimally. Technical Report MPI-I-95-1-025, Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany, September 1995.

[4] J. Komlós, Y. Ma, and E. Szemerédi. Matching nuts and bolts in $O(n \log n)$ time. In *Proceedings of the 7th ACM-SIAM Symposium on Discrete Algorithms (SODA'96)*, pages 232–241, 1996.

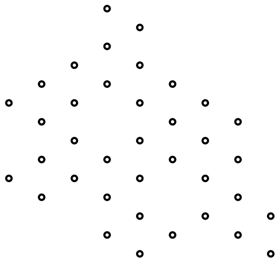
[5] G.J.E. Rawlins. *Compared to What ? An Introduction to the Analysis of Algorithms.* Computer Science Press, 1992.

4.2 Graph and Network Algorithms

Algorithms for Graphs of Small Treewidth

Investigator: Torben Hagerup

It was realized in the 1970s that even hard graph problems are often easy when restricted to trees. At an abstract level, the reason for this usually is that the tree structure allows an efficient bottom-up or dynamic-programming solution approach. Gradually the techniques were extended, with some effort, to other treelike graphs, such as series-parallel or outerplanar graphs, and by the late 1980s it had become clear that an adequate measure of being "treelike" was having small *treewidth*. The rather complicated and nonintuitive definition of treewidth will not be given here, but we offer the following example of a graph of treewidth 2.



Usually efficient algorithms for solving problems defined by graphs of small treewidth need to know precisely *how* an input graph is similar to a tree before they can take advantage of the tree structure (which may not be evident). In technical terms, the algorithms need a *tree decomposition* of the input graph. Constructing tree decompositions of graphs of bounded treewidth therefore is a key problem, and successively faster algorithms for this task culminated in a linear-time algorithm by Bodlaender [1]. Tree decompositions are as useful in a parallel setting as they are in a sequential one, for much the same reason, and in [2] we described an efficient parallel tree-decomposition algorithm. The parallelization required the development of several new techniques, one of which, based on derandomization, has applications to a number of other parallel graph algorithms.

Although Bodlaender's algorithm in principle closes the problem of computing tree decompositions sequentially, in practice his algorithm is probably too complicated to ever be implemented and likely to be affected by large constant factors in the running time (this is even more true of our parallel algorithm). We plan to try to develop more practical algorithms for (sequential) tree decomposition. A theoretical challenge is to reduce the running time of the parallel algorithm from $O\big((\log n)^2\big)$ to $O(\log n)$ ($n =$ number of vertices).

Approaching the same topic from a different angle, one may ask for characterizations of the computational problems that can be solved efficiently on graphs of bounded treewidth. No complete such characterization is known or likely to be forthcoming, but a formalism that allows the expression of a vast number of decision problems on graphs is that of *monadic second-order (MS) logic*. As an example, the formula

$$\begin{aligned} \mathit{Joins}(A,s,t) \;\;\equiv\;\; & \forall U((s \in U \wedge t \notin U) \Rightarrow \\ & \quad \exists e \exists u \exists v(\mathit{tail}(u,e) \wedge \mathit{head}(v,e) \wedge e \in A \wedge u \in U \wedge v \notin U)) \end{aligned}$$

expresses the fact that the subgraph spanned by an edge set A contains a path from s to t. MS logic is a powerful language that allows, e.g., the existence of a Hamiltonian path or of a 3-coloring to be easily asserted. On the other hand, a result due to Courcelle [3] shows that every graph property definable by an MS sentence can be decided in linear time on graphs of bounded treewidth. Later this was extended by various researchers from decision problems to counting, optimization, reliability, and other types of problems. In [4] we extended the framework still further from static problems, where the graph and the question to be answered are known from the outset, to dynamic problems, where, following a certain preprocessing, a sequence of updates and queries must be executed online. To show the variety of problems that can be handled automatically using our framework, we provide the following examples of possible queries, where s, t, u, v, and w are vertices specified in a query: (1) How many cuts are there separating s from t? (2) Is there a DFS tree rooted at s in which u is a descendant of v, but not of w? (3) Which is the shortest simple path from s to t? (where each edge is assumed to have an associated real length).

References

[1] H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25:1305–1317, 1996.

• [2] H. L. Bodlaender and T. Hagerup. Parallel algorithms with optimal speedup for bounded treewidth. In *Proc. 22nd International Colloquium on Automata, Languages and Programming (ICALP), Lecture Notes in Computer Science, Vol. 944*, pages 268–279. Springer-Verlag, 1995. Also *SIAM J. Comput.*, to appear.

[3] B. Courcelle. The monadic second-order logic of graphs. I. recognizable sets of finite graphs. *Inform. and Comput.*, 85:12–75, 1990.

• [4] T. Hagerup. Dynamic algorithms for graphs of bounded treewidth. In *Proc. 24th International Colloquium on Automata, Languages and Programming (ICALP), Lecture Notes in Computer Science.* Springer-Verlag, 1997. To appear.

Max-Flows and Min-Cuts in Sparse Networks

Investigators: Srinivasa Arikati, Shiva Chaudhuri, and Christos Zaroliagis

A central problem in network flows is that of computing the maximum possible amount of flow that can be sent from a specific source s to a specific sink t (called *s-t max-flow*) in a network with capacities on its edges. By the well-known max-flow min-cut theorem, the problem is equivalent to computing an *s-t cut* in the network (i.e., a partition of its vertices into two sets, one containing s and the other containing t) of minimum capacity (where the capacity of the cut is defined as the sum of the capacities of the edges going from the part containing s to the part containing t). The best algorithm for computing an s-t max-flow or an s-t min-cut in an n-vertex, m-edge network with real edge capacities is $O\big(\min\{n^3/\log n, nm\log n\}\big)$ [2, 6]. In the case of bounded treewidth and planar networks better algorithms were known: an $O(n)$ time algorithm for the former class [5] and an $O(n\log n)$ time algorithm for the latter class of networks [7].

In [1], we consider the all-pairs min-cut problem (APMC): for any pair of vertices s and t in the network, the task is to compute an s-t min-cut. The naive solution to the APMC problem is to compute $\Theta(n^2)$ max-flow problems, and this is the best known so far for the case of directed networks (i.e., whose underlying graph is directed). In the case of undirected networks it was shown in [4] that the APMC problem can be solved by computing $n-1$ suitably chosen max-flow problems.

In [1], we give new algorithms for the APMC problem in directed sparse networks. The starting point is a new optimal algorithm for the APMC problem for bounded treewidth networks that runs in $O(n^2)$ time. Actually, our algorithm does something more than this: it can find the value of an s-t min-cut in $O(\alpha(n))$ time, after an $O(n)$ preprocessing of the input network, where $O(\alpha(n))$ is the inverse-Ackermann function. This algorithm is used to develop efficient algorithms for sparse networks which are based on a decomposition of the original network into outerplanar networks (that have treewidth 2), called hammocks [3]; the number of hammocks obtained, $\tilde{\gamma}$, depends on the topological properties of the input sparse network and varies between 1 and $\Theta(n)$. In the case of planar directed networks, this leads to a solution of the s-t min-cut (or s-t max-flow) problem in $O(n + \tilde{\gamma} \log \tilde{\gamma})$ time, and to a solution of the APMC problem in time $O(n^2 + \tilde{\gamma}^3 \log \tilde{\gamma})$. Thus our algorithms are always competitive with the best previous ones and are significantly better when $\tilde{\gamma} = o(n)$. Analogous improvements are achieved for general sparse networks. All of our algorithms can output the corresponding s-t min-cut in time proportional to the number of edges in the cut.

Our methods use the construction of a small network which "mimics" the flow behavior of a large network. This idea was developed in [5], where it is shown that a network G with q terminals has a mimicking network of size 2^{2^q}. To obtain our algorithms for planar and sparse networks, we also prove a new result of independent interest: an outerplanar network has a mimicking network which is also outerplanar and which has size $q^2 2^{q+2}$.

Work in progress aims at providing better constructions for mimicking networks.

References

- [1] S. Arikati, S. Chaudhuri, and C. Zaroliagis. All-pairs min-cut in sparse networks. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS'95)*, pages 363–376. Lect. Notes in Comp. Sc. 1026, Springer-Verlag, 1995. Also Tech. Rep. MPI-I-96-1-007, 1996.

[2] J. Cheriyan, T. Hagerup, and K. Mehlhorn. Can a maximum flow be computed on $o(nm)$ time? In *Automata, Languages and Programming (ICALP'90)*, pages 235–248. Lect. Notes in Comp. Sc. 443, Springer-Verlag, 1990.

[3] G. Frederickson. Using cellular graph embeddings in solving all pairs shortest path problems. *Journal of Algorithms*, 19:45–85, 1995.

[4] R.E. Gomory and T.C. Hu. Multi-terminal network flows. *Journal of SIAM*, 9:551–570, 1961.

[5] T. Hagerup, J. Katajainen, N. Nishimura, and P. Ragde. Characterizations of k-terminal flow networks and computing network flows in partial k-trees. In *Proc. 6th ACM-SIAM Symp. on Discrete Algorithms (SODA'95)*, pages 641–649, 1995.

[6] D.D. Sleator and R.E. Tarjan. A data structure for dynamic trees. *Journal of Computer and System Sciences*, 26:362–391, 1983.

[7] K. Weihe. Maximum (s,t)-flows in planar networks in $O(|V| \log |V|)$ time. In *Proc. 35th Annual IEEE Symp. on Foundations of Comp. Sc. (FOCS'94)*, pages 178–189, 1994.

Fast Data Transmissions in Networks

Investigator: Christos Zaroliagis

Fast data transmissions between any two nodes in a communication network are of crucial importance. In this problem we are given a network $N = (V, E, c, l)$, where $G = (V, E)$ is a directed graph and each edge $e \in E$ is associated with a capacity $c(e) > 0$ and a lead or delay time $l(e) \geq 0$. The nodes represent transmitters/receivers without data memories and the edges represent communication channels. The capacity $c(e)$ of an edge e represents the amount of data that can be transmitted in a time unit through e. The delay $l(e)$ of an edge e represents the time required for the data to traverse e. Let p be a path in N. The capacity $c(p)$ of p is defined as the minimum capacity of the path edges in p and the lead time $l(p)$ of p is defined as the sum of the lead times of the path edges in p. The transmission time to send σ units of data from a node $s \in V$ to a node $t \in V$ using p, is $T(\sigma, p) = l(p) + \frac{\sigma}{c(p)}$.

The task, here, is to compute a partition $\sigma_1, \ldots, \sigma_j$ of the σ units of data and a set of paths $p_1, \ldots, p_j$, from s to t, not necessarily edge or node disjoint, in order to transmit the data and such that: (i) each σ_i, $1 \leq i \leq j$, is a positive integer; (ii) $\sum_{i=1}^{j} \sigma_i = \sigma$, where σ_i is the amount of data transmitted through path p_i; (iii) the total amount of data that pass through each edge $e \in E$ in a time unit does not exceed $c(e)$; and (iv) the transmission time $\max_{1 \leq i \leq j} T(\sigma_i, p_i)$ is minimized over all possible data partitions and groups of transmission paths.

In [2] we show, unfortunately, that the above problem – called the *minimum transmission time (MTT)* problem – is NP-complete which, in general, precludes a polynomial-time solution for it. Note that in the absence of delays, which is a special case of the problem (i.e., $l(e) = 0$, $\forall e \in E$), a polynomial time algorithm was already known [1].

An alternative approach to the MTT problem might be to partition the σ units of data into σ_i units, $1 \leq i \leq k$, for some positive integer k, and then transmit each σ_i sequentially along a single path which minimizes the transmission time. The proposed partitioning scheme would consist of k stages, and at each stage i one has to find the path that minimizes the transmission time for transmitting the σ_i units of data. We have also shown in [2] that there is no partition of data which, when routed in this manner, yields an overall transmission time less than the time needed for transmitting all σ units of data along a single path from s to t which minimizes the transmission time; such a path is called the *quickest path.* For this reason, we also investigate in [2] the quickest path problem. Note that the problem is a restricted version of the MTT one, where no partition of data is required and the transmission is done along a single path. The relation between the quickest path and the MTT problem is new, although the former problem had already been investigated [3].

In [2] we have given efficient algorithms for the quickest path problem in general and sparse networks which outperform all previous approaches (especially for the latter class of networks). Furthermore, we have examined the dynamic version of the quickest path problem, in which the lead time and the capacity of an edge may change, as well as the amount of data to be transmitted. To the best of our knowledge, there was no previous algorithms for the dynamic quickest path problem.

References

[1] A. Itai and M. Rodeh. Scheduling transmissions in a network. *Journal of Algorithms*, 6:409–429, 1985.

- [2] D. Kagaris, G. Pantziou, S. Tragoudas, and C. Zaroliagis. On the computation of fast data transmissions in networks with capacities and delays. In *Algorithms and Data Structures (WADS'95)*, pages 291–302. Lect. Notes in Comp. Sc. 955, Springer-Verlag, 1995. Also to appear in *Networks*, 1997.

[3] J.B. Rosen, S.Z. Sun, and G.L. Xue. Algorithms for the quickest path problem and the enumeration of quickest paths. *Computers and Operations Research*, 18:579–584, 1991.

The Graph Thickness Problem

Investigator: Petra Mutzel

The thickness $\theta(G)$ of a graph $G = (V, E)$ is the minimum number k such that G is the union of k planar subgraphs (here, by "union of k planar subgraphs" we mean that the edge-set E can be partitioned into k sets so that the graph induced by each set is planar). Therefore, the thickness is one measure of the degree of nonplanarity of a graph.

Clearly, $\theta(G) = 1$ if and only if G is planar. The thickness problem, asking for the thickness of a given graph G, is $\mathcal{NP}$-hard [3], so there is little hope to find a polynomial time algorithm for the thickness problem on general graphs. However, for some graph classes, the thickness can be determined in polynomial time. For example, the thickness is known for complete graphs and complete bipartite graphs [1]. In some cases, there are (often relatively poor) bounds on the thickness of a graph. In [4] we give a survey on known results about this problem.

The thickness problem has applications in VLSI-design. In electronic circuits, components are joined by means of conducting strips. These may not cross, since this would lead to undesirable signals. In this case, an insulated wire must be used. For that reason, circuits with a large number of crossings are decomposed into several layers without crossings, which are then pasted together. The goal is to use as few layers as possible. In this application it would be desirable to know the thickness of a hypergraph whose nodes are cells to be placed and whose hyperedges correspond to the nets connecting the cells. If the thickness problem could be solved for graphs, it would be a useful engineering tool in the layout of electronic circuits.

Using the classical decomposition theorem by Wagner for graphs without K_5-minors [6], we show that the thickness of these class of graphs is at most two; this leads to a linear time algorithm for determining the thickness of K_5-minor free graphs [2]. The decomposition theorem essentially states that any K_5-minor free graph is planar, or can be constructed recursively by 2-sums and Δ-sums of certain building blocks. Examples of such building blocks are planar graphs or the graph $K_{3,3}$.

Figure 4.1: The graph G_{12}

Moreover in [2], we have been able to extend this result to the graph class with no G_{12}-minors (see Figure 4.1) using a similar decomposition theorem by Truemper [5]. In particular, this graph class contains the classes of K_5- and $K_{3,3}$-minor free graphs (except for K_5 itself).

References

[1] L. Beineke and R. Wilson. *Selected topics in graph theory.* Academic Press, 1978.

• [2] M. Jünger, P. Mutzel, T. Odenthal, and M. Scharbrodt. The thickness of a minor-excluded class of graphs. Technical Report MPI-I-95-1-027, Max-Planck-Institut für Informatik, Saarbrücken, 1995. To appear in: Discrete Mathematics.

[3] A. Mansfield. Determining the thickness of graphs is NP-hard. *Math. Proc. Cambridge Philos. Soc.*, 9:9–23, 1983.

• [4] P. Mutzel, T. Odenthal, and M. Scharbrodt. The thickness of graphs: A survey. Technical Report MPI-I-95-1-009, Max-Planck-Institut für Informatik, Saarbrücken, 1996. To appear in: Graphs and Combinatorics.

[5] K. Truemper. *Matroid decomposition.* Academic Press, 1992.

[6] K. Wagner. Über eine Eigenschaft ebener Komplexe. *Math. Ann.*, 114:570–590, 1937.

Finding Agreement Subtrees

Investigator: Ramesh Hariharan

Given two binary trees with leaves drawn from the same set of items (e.g., species or languages, as in the case of evolutionary trees), how does one find the *maximum agreement subtree* of these two trees? The maximum agreement subtree is the largest subset of the leaf items so that the portions of the two trees restricted to these items are isomorphic.

The motivation behind this problem is the following. An evolutionary tree for a set of species/languages is a rooted tree whose leaves represent the species/languages and whose internal nodes represent ancestor information. There is no common standard for constructing evolutionary trees and different methods often give different answers. Current practice dictates that different methods be applied to the same data and the resulting trees be compared to arrive at a consensus. The Maximum Agreement Subtree is one method of arriving at such a consensus.

We obtained an optimal algorithm for this problem which takes $O(n \log n)$ time, where n is the size of each tree [1]. This improves the previous best bound of $O(n \log^3 n)$ due to Farach, Przytycka, and Thorup [2]. Our algorithm uses a recursive scheme coupled with a non-trivial analysis to account for the costs of the various recursive subproblems created.

References

• [1] R. Cole and R. Hariharan. An $O(n \log n)$ algorithm for the maximum agreement subtree problem for binary trees. In *Proceedings of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 323–332, 1996.

[2] M. Farach, T. Przytycka, and M. Thorup. The maximum agreement subtree problem for binary trees. In *Proceedings of 3rd Annual European Symposium on Algorithms, Springer Lecture Notes in Computer Science, Volume 979*, pages 381–393, 1995.

4.3 Probabilistic Analysis of Algorithms and Applications

Average-case Complexity of Shortest-Paths Problems

Investigators: Kurt Mehlhorn and Volker Priebe

A variety of combinatorial optimization problems can be modeled as *shortest-paths problems*. Given a (directed) n-vertex graph in which edges are assigned real edge lengths, these problems ask for *distances* between pairs of vertices, where the distance of vertex j from vertex i is defined as the infimum of the lengths of all directed paths from i to j. Two types of shortest-paths problems are usually considered. In the *single-source shortest-paths problem*, we are interested in the distances of all vertices from a given source vertex; in the *all-pairs shortest-paths problem*, we want to compute the distance between each pair of vertices.

There is a wealth of literature on the worst-case performance of algorithms for shortest-paths problems. However, worst-case analysis sometimes fails to bring out the advantages of algorithms that perform well in practice; average-case analysis has turned out to be more appropriate for these purposes. We therefore study the expected running time of shortest-paths algorithms, with instances of shortest-paths problems generated according to a probability distribution on the set of complete directed graphs with edge lengths. Two kinds of probability distributions have been considered in the literature. In the *uniform model*, the edge lengths are independent, identically distributed random variables. The *endpoint-independent model* is more general. A random instance generated according to the endpoint-independent model has the property that if the edges leaving a vertex are sorted according to their lengths, then the associated endpoints occur in random order. This includes even the case that the edge lengths themselves are arbitrarily fixed, and only the assignment of the edge lengths to the edges leaving a vertex is random.

The average running time of shortest-paths algorithms has mainly been studied for the case of non-negative edge lengths. In this case, if instances are generated according to the uniform model, the structure of these graphs (as related to shortest-paths problems) is well understood, and there are several algorithms [3, 4, 6] that solve the all-pairs shortest-paths problem in expected time $O(n^2 \log n)$. The algorithm by Moffat and Takaoka [8] can be shown to solve the all-pairs shortest-paths problem within the same bounds on the expected running time, even with respect to the endpoint-independent model. A concise analysis, however, turns out to be intricate, and, as far as we know, is given in [7] for the first time. Moreover, we prove that the algorithm is *reliable*, that is, finishes its computation within time $O(n^2 \log n)$ with high probability.

Concerning the case of *arbitrary* real edge lengths, it is a major problem to define a probability distribution that allows negative edge lengths but does not trivialize the problem. If, for example, edge lengths have a non-zero probability of being negative and instances are generated according to the uniform model, then, with high probability, the graph will contain a *negative cycle* (a directed cycle of negative length) which renders all distances $-\infty$. To the best of our knowledge, the first probability distribution for graphs with arbitrary real edge lengths was proposed by Kolliopoulos and Stein [5]. They study the endpoint-independent model and give an algorithm for the single-source shortest-paths problem with expected running time $O(n^2 \log n)$.

In [1], we study the average-case complexity of shortest-paths problems in the *vertex-potential model.* This is a family of probability distributions on complete directed graphs with arbitrary real edge lengths but without negative cycles. In the vertex-potential model, there is a random variable π_i for each vertex $i \in [n] = \{1, \ldots, n\}$ and a random variable $r_{i,j}$ for each edge (i, j), $i, j \in [n]$. The length $c_{i,j}$ of each edge (i, j) is defined by

$$c_{i,j} = r_{i,j} - \pi_i + \pi_j \quad .$$

Of course, only the $c_{i,j}$'s are revealed to our algorithms and the $r_{i,j}$'s and π_i's are hidden parameters of the model. The $r_{i,j}$'s, $i, j \in [n]$, are independent, identically distributed random variables with values in $[0, 1]$, and the π_i's, $i \in [n]$, are independent, identically distributed random variables with values in $[-1, 1]$ and mean 0. The assumption $r_{i,j} \geq 0$, $i, j \in [n]$, guarantees that instances generated according to the vertex-potential model contain no negative cycle. We show that the single-source shortest-paths problem can be solved in $O(n^2)$ expected time and that the all-pairs shortest-paths problem can be solved in $O(n^2 \log n)$ expected time. In both cases our algorithms are reliable.

How fast can shortest-paths problems be solved on average? We prove in [7] that for a large class of probability distributions, $\Omega(n \log n)$ time is necessary to solve the single-source shortest-paths problem. The design of an algorithm that solves the all-pairs shortest-paths problem in expected time $o(n^2 \log n)$ with respect to a reasonable probability distribution remains a challenge [2, Research Problem 22].

References

- [1] C. Cooper, A. Frieze, K. Mehlhorn, and V. Priebe. On the average-case complexity of shortest-paths problems in the vertex-potential model. To be presented at *1st International Symposium on Randomization and Approximation Techniques in Computer Science*, Bologna, July 11–12, 1997.

[2] A. Frieze and C. McDiarmid. Algorithmic theory of random graphs. *Random Structures & Algorithms*, 10:5–42, 1997.

[3] A. M. Frieze and G. R. Grimmett. The shortest-path problem for graphs with random arc-lengths. *Discrete Appl. Math.*, 10:57–77, 1985.

[4] D. R. Karger, D. Koller, and S. J. Phillips. Finding the hidden path: Time bounds for all-pairs shortest paths. *SIAM J. Comput.*, 22:1199–1217, 1993.

[5] S. G. Kolliopoulos and C. Stein. Finding real-valued single-source shortest paths in $o(n^3)$ expected time. In W. Cunningham, S. McCormick, and M. Queyranne, editors, *Integer Programming and Combinatorial Optimization (Proc. 5th International IPCO Conference 1996)*, volume 1084 of *Lecture Notes in Comput. Science*, pages 94–104, 1996.

[6] C. C. McGeoch. All-pairs shortest paths and the essential subgraph. *Algorithmica*, 13:426–441, 1995.

• [7] K. Mehlhorn and V. Priebe. On the all-pairs shortest-path algorithm of Moffat and Takaoka. *Random Structures & Algorithms*, 10:205–220, 1997.

[8] A. Moffat and T. Takaoka. An all pairs shortest path algorithm with expected time $O(n^2 \log n)$. *SIAM J. Comput.*, 16:1023–1031, 1987.

The Fourth Moment in Luby's Distribution

Investigator: Christos Zaroliagis

Probabilistic analysis of algorithms was proved extremely useful in the derandomization of randomized algorithms. These algorithms, contrary to probabilistic algorithms, do not make any assumption on the input distribution. A randomized algorithm performs random choices during its execution and its running time becomes a random variable. Analyzing such an algorithm involves understanding the distribution of this (and possibly other) random variable(s). This is crucial in the derandomization of randomized algorithms, where the typical approach used is the following: the random variables considered are defined over a smaller probability space, specially designed, containing only a polynomial number of sample points. In that space, the random variables are proved to be only k-wise independent (for constant k); and this is usually enough to replace the analysis of the randomized algorithm with fully independent random variables. In this way efficient deterministic (sequential and parallel) algorithms can be developed, because one can afford now an exhaustive search of this polynomial size sample space to find a good solution (whose existence is guaranteed by the probabilistic analysis).

Luby [3] designed such a small probability space whose elements are 3-wise independent; this space was proved extremely useful in the design of fast and efficient deterministic parallel (i.e., NC) algorithms for many problems. In most of the cases 3-wise independence is enough. However, in some other cases this is not sufficient [1]. The algorithms described in [1] can be derandomized successfully only because of the 4-wise independence property (which Luby's probability space does not have and thus cannot be used for the derandomization of these algorithms). But perhaps, it can be hoped that by relating the fourth moments under Luby's distribution and the fully independent distribution, one can use Luby's distribution in some other cases.

In [2] we examine the fourth moment in Luby's space and relate it precisely with the fourth moment under the fully independent distribution. As an application, we study the behavior of random edge cuts in a weighted graph. Based on Luby's probability space, it is easy to construct a linear sized space of edge cuts. We then prove that this smaller space has *bigger* variance compared with the variance in the fully independent space of all possible edge cuts taken equiprobably (which is exponential in size). Thus, the smaller space can be a good predictor of extreme values of random variables defined on the larger space, possibly leading to NC algorithms for better approximations to the maximum edge cut problem.

References

[1] B. Berger. The fourth moment method. In *Proc. 2nd ACM-SIAM Symp. on Discrete Algorithms (SODA'91)*, pages 373–383, 1991.

• [2] D. Dubhashi, G. Pantziou, P. Spirakis, and C. Zaroliagis. The fourth moment in Luby's distribution. *Theoretical Computer Science*, 148(1):133–140, 1995.

[3] M. Luby. Removing randomness in parallel computation without a processor penalty. In *Proc. 29th IEEE Symp. on Foundations of Comp. Sc. (FOCS'88)*, pages 162–174, 1988.

Randomization Techniques in Data Mining

Investigators: Dimitris Gunopulos, Sanjeev Saluja, and Heikki Mannila

Data mining is a relatively new field in computer science that is concerned with searching and analyzing very large data sets. Data mining can in many instances be viewed as the task of computing a representation of a theory of a model or a database. In [1] we present a randomized algorithm that can be used to compute the representation of the theory in terms of the most specific sentences of that theory. In addition to randomization, the algorithm uses a generalization of the concept of hypergraph traversals. We apply the general algorithm in two ways: for the problem of discovering maximal frequent sets in 0/1 data, and for computing minimal keys in relations. We present some empirical results on the performance of these methods in real data. We also show some complexity theoretic evidence of the hardness of these problems.

References

• [1] D. Gunopulos, H. Mannila, and S. Saluja. Discovering all most specific sentences by randomized algorithms. In *Proc. 6th Intern. Conf. on Database Theory (ICDT'97)*, pages 215–229. Lect. Notes in Comp. Sc. 1186, Srpinger-Verlag, 1997. Also Tech. Rep. MPI-I-96-1-023, 1996.

4.4 Pattern Matching

String Matching and Applications

Investigator: Leszek Gąsieniec

The string matching problem is another fundamental and well-studied problem within the theoretical computer science. Given two strings, a pattern p and a text t (of length m and n, respectively), the *string matching problem* is to find all occurrences of p in t. Many sequential and optimal parallel algorithms are already known for the problem; see e.g., the recent book [7]. The recent progress in the so called *stringology* caused an extension of the standard string matching to more complex objects as matrices, trees, dictionaries, and finally leading to the more general area called *pattern matching*. Simultaneously the researchers became more interested in various aspects of the matching problem, like dealing with large data (e.g. *compression*), partial matchings (e.g. *approximate matchings*), or optimization problems with a wide application to the other fields like data mining, linguistics, computational biology, and data security. However, still some important problems in the core string matching remain open.

One of the recent challenging problems is to deal with extremely large data which usually means dealing with their compressed form. The possibility to do searching and updating with the compressed files (without decompression) turned out to be quite hard when the traditional compression is in use. However, some attempt to the fully compressed string matching was presented in [12, 11]. We introduced two polynomial time string matching algorithms: deterministic and randomized, both working for the compressed (with Lempel-Ziv encoding [16]) pattern and text.

The string matching has also its natural applications in *data mining* – a new field in computer science which deals with searching and analyzing huge data. In [15], Mannila et al. introduced the problem of finding all frequent episodes in the event sequences. In [8] we consider its simplified version called an *episode matching*, i.e., we search for the short occurrences of a given episode p in the input sequence t. In contrast to the standard string matching, it is allowed that the episode p is a sparse subsequence of t which makes the entire problem much more complex. We present several on-line and off-line, deterministic, time-space efficient algorithms, all of which deliver the exact solutions. In addition, we present a superlinear deterministic algorithm which gives the approximate answers with any constant approximation ratio.

Another aspect of the pattern matching problem is considered in [10]. For a given text t, the *external inverse pattern matching* problem is to construct a pattern p which minimizes (maximizes) the sum of *hamming distances* between p and all text subwords of length m. It was shown in [2] that the inverse pattern matching is strongly related to other string problems like generalized string matching [1] and has the applications in information retrieval, data compression, computer security and molecular biology. In [10] we present the first optimal linear time algorithm for the *external inverse pattern matching* problem which improves significantly over the previously known quadratic time algorithm [2].

In many string applications it is very important to have a quick test if a given word belongs to a given set of words (called *dictionary*). The problem becomes more difficult if a query could contain errors and instead of the exact matches we are rather interested in the approximate matches. In [3] we present the first (optimal) linear time algorithm based on a linear size data structure (representing the dictionary) which solves the approximate dictionary matching problem when only one error in the query string is allowed.

One of the core string matching problems is the time-space trade-off. For example, there are known linear time string matching algorithms which use constant space [6, 9, 13]. All of them have an average linear time complexity. It is also known [14] that when the text is random one can expect that the algorithm has $O\left(\frac{n\log m}{m}\right)$ average time complexity. However, in this case all known algorithms (see e.g., [4]) use linear space. The relation between average time and space turned out to be an important, but still not answered, question. In [5] we do a small progress presenting an $o(n)$ time string matching algorithm which works with constant additional memory.

References

[1] K. Abrahamson. Generalized string matching. *SIAM Journal on Computing*, 16(6):1039–1051, 1987.

[2] A. Amir, A. Apostolico, and M. Lewenstein. Inverse pattern matching. *Journal of Algorithms*, to appear.

• [3] G. Brodal and L. Gąsieniec. Approximate dictionary queries. In *Proceedings of 7th Symposium on Combinatorial Pattern Matching (CPM'96)*, pages 65–74, June 1996.

[4] M. Crochemore, A. Czumaj, L. Gąsieniec, S. Jarominek, T. Lecroq, W. Plandowski, and W. Rytter. Speeding up two string matching algorithms. *Algorithmica*, 12:247–267, 1994.

• [5] M. Crochemore, L. Gąsieniec, and W. Rytter. Constant-space string-matching in sublinear average time. In *Proceedings of Compression and Complexity of SEQUENCES 1997*, June 1997. To appear.

[6] M. Crochemore and D. Perrin. Two-way string-matching. *J. Assoc. Comput. Mach.*, 38(3):651–675, 1991.

[7] M. Crochemore and W. Rytter. *Text Algorithms.* Oxford University Press, New York, 1994.

• [8] G. Das, R. Fleischer, L. Gąsieniec, D. Gunopulos, and J. Karkkainen. Episode matching. In *Proceedings 8th Symposium on Combinatorial Pattern Matching (CPM'97)*, July 1997. To appear.

[9] Z. Galil and J. Seiferas. Time-space optimal string matching. *J. Comput. System Sci.*, 26:280–294, 1983.

• [10] L. Gąsieniec, P. Indyk, and P. Krysta. External inverse pattern matching. In *Proceedings 8th Symposium on Combinatorial Pattern Matching (CPM'97)*, July 1997. To appear.

• [11] L. Gąsieniec, M. Karpiński, W. Plandowski, and W. Rytter. Efficient algorithms for Lempel-Ziv encoding. In *Proceedings 5th Scandinavian Workshop on Algorithm Theory (SWAT'96)*, pages 392–403, July 1996.

• [12] L. Gąsieniec, M. Karpiński, W. Plandowski, and W. Rytter. Randomized efficient algorithms for compressed strings: the finger-print approach. In *Proceedings 7th Symposium on Combinatorial Pattern Matching (CPM'96)*, pages 39–49, June 1996.

[13] L. Gąsieniec, W. Plandowski, and W. Rytter. Sequential sampling: a new approach to constant space pattern-matching. In *Proceedings of 6th Symposium on Combinatorial Pattern Matching (CPM'95)*, pages 78–89, June 1995.

[14] D.E. Knuth, J.H. Morris, and V.R. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6:322–350, 1977.

[15] H. Mannila, H. Toivonen, and A.I. Verkamo. Discovering frequent episodes in sequences. In *Proceedings of 1st International Conference on Knowledge Discovery and Data Mining (KDD'95)*, pages 210–215, 1995.

[16] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transaction on Inf. Theory*, IT 23(3):337–343, 1977.

Approximate Pattern Matching

Investigator: Srinivasa Arikati

In the recent years there has been a growing interest in finding efficient algorithms for multi-dimensional pattern matching problems (see e.g., [1, 6]). Consider the following optimization variant of the classical pattern matching problem: Given a text t and a pattern p over a finite alphabet Σ, find the maximum number of *non-overlapping* occurrences of the pattern p in t. We call this problem the *maximum pattern matching* problem and use MPM_2 to denote the maximum 2-dimensional pattern matching problem. Maximum pattern matching problems arise naturally in automated digital image processing and other areas as well [7, 4].

We studied the MPM_2 problem and several of its variants [2]. We observed that a known NP-completeness result on planar geometric packing implies the NP-completeness of the problem MPM_2. Then, we gave a simple and efficient approximation algorithm with a performance guarantee of 2 for the problem MPM_2. If the set of the so-called periods of the pattern is appropriately restricted, our simple approach yields maximum solutions. We then presented an involved approximation algorithm for MPM_2, based on good separation properties of the intersection graph of the pattern locations. This algorithm yields a solution of relative error $O\big(1/\sqrt{\log\log n}\big)$ for constant size patterns, and runs in $O(n \log n)$ time, on an input of size n. We also presented another sophisticated approximation algorithm for MPM_2 based on the *shifting strategy* [3, 5]. Specifically, when patterns are of fixed size, we obtained NC-approximation schemes for MPM_2.

References

[1] A. Amir, G. Benson, and M. Farach. An alphabet-independent approach to two-dimensional matching. In *Proceedings 24th ACM Symposium on Theory of Computing*, pages 59–68, 1992. Journal version to appear in *SIAM J. Computing.*

- [2] S.R. Arikati, A. Dessmark, A. Lingas, and M.V. Marathe. Approximation algorithms for maximum two-dimensional pattern matching. In *Proceedings Combinatorial Pattern Matching, Lecture Notes in Computer Science, Vol. 1075*, pages 348–360, 1996.

[3] B.S. Baker. Approximation algorithms for NP-complete problems on planar graphs. In *Proceedings 24th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 265–273, 1984. Journal version in *J. ACM,* Vol. 41, No. 1, 1994, pp. 153-180.

[4] M. Crochemore and W. Rytter. *Text Algorithms.* Oxford University Press, New York, 1994.

[5] D.S. Hochbaum and W. Maass. Fast approximation algorithms for a nonconvex covering problem. *Journal of Algorithms*, 8:305–323, 1987.

[6] S. R. Kosaraju. Faster algorithms for the construction of parameterized suffix trees. In *Proceedings 36th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 631–637, 1995.

[7] CANDID Project. Los Alamos National Laboratories, 1993.

4.5 Online Algorithms

Investigator: Susanne Albers

Loosely speaking, online computation is decision-making with incomplete information. Typically, an online algorithm receives a sequence of *requests* and must immediately react to each of these requests, not knowing future requests. An online algorithm A is *c-competitive* if, for every request sequence, A's solution is at most a factor of c away from an optimal solution for the sequence.

Our work on online algorithms covers a variety of different topics, including data structuring, scheduling, bin packing, exploration and learning, as well as distributed data management.

Scheduling

One of the most basic scheduling problems was introduced by Graham [22] in 1966. Consider a sequence of jobs that have to be scheduled on m identical parallel machines. The goal is to minimize the *makespan*, i.e., the completion time of the last job that finishes. Algorithms for this scheduling problem are used in multiprocessor scheduling. Moreover, the problem is important because it is the root of many problem variants where, for instance, preemption is allowed, precedence constraints exist among jobs, or machines run at different speeds. Of particular interest are *online algorithms* that schedule each job in the sequence without knowledge of any future jobs.

Graham developed the well-known *List* algorithm that schedules every incoming job on the least loaded machine. The *List* algorithm is $(2 - \frac{1}{m})$-competitive. In recent years, research has focused on finding algorithms that achieve a competitive ratio of c, $c < 2$, for all m. Bartal, Fiat, Karloff and Vohra [11] gave an algorithm that is 1.986-competitive. Karger, Phillips and Torng [25] generalized the algorithm and proved an upper bound of 1.945. The best lower bound known so far is due to Bartal, Karloff and Rabani [12] who showed that no deterministic scheduling algorithm can have a competitive ratio smaller than 1.837.

In [2] we have developed a new scheduling algorithm that is 1.923-competitive. Moreover, we have shown that no deterministic online scheduling algorithm can be better than 1.852-competitive. The main contribution of our paper is not the improvement in the competitive ratios, but the new scheduling strategy underlying our solution. At any time, our new algorithm maintains a set S_1 of $\lfloor \frac{m}{2} \rfloor$ machines with a low load and a set S_2 of $\lceil \frac{m}{2} \rceil$ machines with a high load. Every job is either scheduled on the least loaded machine in S_1 or on the least loaded machine in S_2. The decision, which of the two machines to choose, depends on the *ratio* of the load on machines in S_1 to the load on machines in S_2.

Robot Exploration

In [3] we presented a sub-exponential algorithm for an exploration problem posed by Deng and Papadimitriou [20]. A robot has to explore an unknown directed, strongly connected graph using as few edge traversals as possible. This problem has applications in computational learning theory, when a learner (robot) has to learn finite automata. Secondly, the problem arises in robot exploration, when a robot has to construct a complete map of an unknown environment using a short path. In many situations it is convenient to model the geometric environment by a graph because this allows to neglect geometric features of the environment and to concentrate on the combinatorial aspects of the exploration problem.

Deng and Papadimitriou [20] gave an algorithm that explores an m-edge graph using $d^{O(d)}m$ edge traversals. Here d is the *deficiency* of the graph, i.e., the minimum number of edges that have to be added to make the graph Eulerian. Deng and Papadimitriou also gave a graph such that every exploration algorithm needs at least $\Omega(d^2 m / \log d)$ edge traversals. Koutsoupias (reported in the revised version of [20]) improved the lower bound to $\Omega(d^2 m)$.

In [3] we first show that a number of natural exploration strategies, such as greedy, depth-first and breadth-first approaches, do not work well. They need $2^{\Omega(d)}m$ edge traversals in the worst case. Our main result is a new robot strategy that explores an arbitrary graph with deficiency d and traverses each edge at most $(d+1)^6 d^{2\log d}$ times. The algorithm does not need to know d in advance. We also demonstrate that our analysis is tight. There exists a graph that is explored by our algorithm using $d^{\Omega(\log d)}m$ edge traversals. Additionally, we show that various variants of the algorithm have the same lower bound.

Self-organizing Data Structures and Adaptive Data Compression

We have continued our work on the *list update problem*, which is a fundamental and extensively studied online problem. The problem is to maintain an unsorted linear linked list so as minimize the total access cost on a sequence of requests.

In [1] we have presented a randomized list update algorithm that is defined for any real number $p \in [0, 1]$. For $p = 0$, the algorithm is deterministic and can be formulated as follows.

> Algorithm TS(0): Insert the requested item, say x, in front of the first item in the list that has been requested at most once since the last request to x. If x has not been requested so far, leave the position of x unchanged.

In [8] we show that a combination of TS(0) and the BIT algorithm, due to Reingold, Westbrook and Sleator [29], achieves a competitive ratio of 1.6. This represents the best upper bound known for randomized list update. Our results developed in [1, 8] now appear in a text book written by Borodin and El-Yaniv. Substantial portions of the first two chapters in [14] deal with our algorithms and proofs.

In [7] we study the performance of TS(0) when request sequences are generated by probability distributions, i.e., by discrete memoryless sources. Our work is motivated by the goal of finding a universal algorithm that achieves an optimal competitive ratio and has an extremely good performance on memoryless sources. Previous research [30, 17] has shown that the Move-To-Front algorithm is such an algorithm. We show in [7] that TS(0) has an even better overall performance than Move-To-Front. We determine performance ratios of TS(0) against the optimal offline and the optimum static algorithm when requests are generated by discrete memoryless sources. Previous work on such sources compared online algorithms only to static algorithms. One practical motivation for our work is the use of the Move-to-front heuristic in various compression algorithms.

In general, data compression schemes based on list update algorithms can be developed in the following way: The idea is to convert a string S of symbols into a string I of integers. Whenever the symbol x has to be compressed, an encoder looks up the current position of x in a linear list of symbols it maintains, outputs this position and updates the list. A decoder that receives the string I can recover the original message by looking up in its own linear list, for each integer j it reads, the symbol that is currently stored at position j. The decoder also updates its list. Our theoretical results in [7] suggest that in many cases using TS(0) in place of Move-To-Front in schemes that use the latter should improve compression. Tests using implementations on a standard corpus of test documents demonstrate that TS(0) leads indeed to improved compression.

In [6] we have revisited the COUNTER algorithms for list update that were introduced by Reingold, Westbrook and Sleator [29]. These algorithms are interesting because of their simplicity: On each access, a COUNTER algorithm either moves the requested item to the front of the list or leaves it where it is. In [6] we use a mixture of two COUNTER algorithms to achieve a competitiveness of 12/7, which is less than the $\sqrt{3}$ ratio shown by Reingold *et al.* [29]. Furthermore, we demonstrate that it is impossible to prove a competitive ratio smaller than 12/7 for any mixture of COUNTER algorithms using the type of potential function argument that has been used so far. We also provide various new lower bounds on the competitiveness of COUNTER algorithms.

With Jeffery Westbrook we have written a survey [9] on self-organizing data structures that will appear in a book on online algorithms, edited by Fiat and Woeginger. This survey contains all major results published on self-organizing linear lists and binary search trees over the past 25 years.

Distributed Data Management

Page migration is an important online problem in distributed data management. The problem is to distribute a set of memory pages in a network of processors, each of which has its local memory, so that a sequence of memory accesses can be served with low total cost. More specifically, the goal is to minimize the cost incurred by communication (when a prcessor has to read/write a page that is not in its local memory) and possible re-allocation of pages.

In [4] we have studied the page migration problem under the assumption that the local memories are directly mapped, i.e., the processors use a hash function to located page in their memory. We have presented constant competitive algorithms for a number of important network topologies. Our work is among the first papers that investigated page migration in the model that the local memories of the processors have bounded capacity. Most previous work assumed infinite local capacity.

Bin Packing

The *one-dimensional bin packing problem* is a classical problem in computer science. The goal is to pack a sequence $a_1, \ldots, a_n \in (0, 1]$ of items into bins of unit capacity so as to minimize the number of bins used. First Fit is a well-known algorithm for online bin packing that was proposed in the early seventies. Its performance in the worst case and uniform average case has been settled for quite some time. Research on First Fit is now focused on analyzing *expected waste*, i.e. the difference between the number of bins used and the total size of the items packed so far, when item sizes are generated by discrete uniform distributions. A discrete uniform distribution, denoted by $U\{j,k\}, 1 \leq j \leq k$, is one where item sizes are chosen uniformly from the set $\{1/k, 2/k, \ldots, j/k\}$. The expected waste produced by First Fit has been determined by Coffman *et al.* [18] for distributions $U\{k,k\}$, $U\{k-1,k\}$, $k > 1$, and $U\{j,k\}$, when $k \geq j^2$. For the latter distributions, First Fit is *stable*, i.e., the expected waste remains bounded (that is, $O(1)$), even as the number of items n goes to infinity.

In [5] we prove that First Fit is stable under the input distribution $U\{k-2,k\}$, settling an open question from the recent survey by Coffman, Garey, and Johnson [19]. Our proof generalizes the multi-dimensional Markov chain analysis used by Kenyon, Rabani, and Sinclair to prove that Best Fit is also stable under this distribution [26]. Our proof is motivated by an analysis of Random Fit, a new simple packing algorithm related to First Fit, that is interesting in its own right.

Implementation and Visualization of Online Algorithms

To complement our theoretical studies of online algorithms, we also started to build a library of online algorithms with the main focus on visualization and animation. To this purpose, we designed the OnVis system (*On*line *Vis*ualisation system). The OnVis system is a distributed system consisting of a central server and other independent modules (request generators, algorithm modules, animation modules, etc.; this is similar to the Balsa system for algorithm animation [15]) which allows the visualization and animation of several algorithms at the same time to compare their behaviours. A prototype of the system has been implemented [28] together with algorithms for changing money [21] and algorithms for the Lost Cow Problem [10, 24]. Currently, algorithms for navigating in unfamiliar terrain [13], the k-server problem [16, 23, 27], and Splay Trees [31, 32] are implemented. We believe that the OnVis system can develop into a useful tool for demonstrating algorithms and general techniques, especially when teaching students. We are also working on a redesign of the central server to make it easier to use.

References

[1] S. Albers. Improved randomized on-line algorithms for the list update problem. In *Proceedings 6th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 412–419, 1995. To appear: *SIAM Journal on Computing.*

• [2] S. Albers. Better bounds for online scheduling. In *Proceedings 29th Annual ACM Symposium on Theory of Computing (STOC)*, 1997.

• [3] S. Albers and M. Henzinger. Exploring unknown environments. In *Proceedings 29th Annual ACM Symposium on Theory of Computing (STOC)*, 1997.

• [4] S. Albers and H. Koga. Page migration with limited local memory capacity. In *Proceedings 4th International Workshop on Algorithms and Data structures (WADS), Springer Lecture Notes in Computer Science, Volume 955*, pages 514–525, 1995.

• [5] S. Albers and M. Mitzenmacher. Average-case analyses of First Fit and Random Fit bin packing. Manuscript 1997.

• [6] S. Albers and M. Mitzenmacher. Revisiting the COUNTER algorithms for list update. Manuscript 1997. Submitted to *Information Processing Letters.*

- [7] S. Albers and M. Mitzenmacher. Average case analyses of list update algorithms, with applications to data compression. In *Proceedings 23rd International Colloquium on Automata, Languages and Programming (ICALP), Springer Lecture Notes in Computer Science, Volume 1099*, pages 147–158, 1996. To appear in *Algorithmica*.
- [8] S. Albers, B. von Stengel, and R. Werchner. A combined BIT and TIMESTAMP algorithm for the list update problem. *Information Processing Letters*, 56:135–139, 1995.
- [9] S. Albers and J. Westbrook. A survey of self-organizing data structures. Technical Report MPI-I-96-1-026, Max-Planck-Institut für Informatik, Saarbrücken, 1996. To appear in: Online Algorithms. Ed.: A. Fiat and G. Woeginger. Springer Verlag.

[10] R.A. Baeza-Yates, J.C. Culberson, and G.J.E. Rawlins. Searching in the plane. *Information and Computation*, 106:234–252, 1993.

[11] A. Bartal, A. Fiat, H. Karloff, and R. Vohra. New algorithms for an ancient scheduling problem. In *Proceedings 24th Annual ACM Symposium on Theory of Computing*, pages 51–58, 1992.

[12] Y. Bartal, H. Karloff, and Y. Rabani. A better lower bound for on-line scheduling. *Information Processing Letters*, 50:113–116, 1994.

[13] A. Blum, P. Raghavan, and B. Schieber. Navigating in unfamiliar geometric terrain. In *Proceedings of the 23rd ACM Symposium on the Theory of Computation (STOC'91)*, pages 494–504, 1991.

[14] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, 1997. To appear.

[15] M.H. Brown and R. Sedgewick. A system for algorithm animation. *Computer Graphics*, 18(3):177–186, 1984.

[16] M. Chrobak and L.L. Larmore. An optimal on-line algorithm for k-servers on trees. *SIAM Journal on Computing*, 20(1):144–148, 1991.

[17] F.R.K. Chung, D.J. Hajela, and P.D. Seymour. Self-organizing sequential search and Hilbert's inequality. In *Proceedings 17th Annual Symposium on the Theory of Computing*, pages 217–223, 1985.

[18] E.G. Coffman, C. Courcoubetis, M.R. Garey, D.S. Johnson, L.A. McGeoch, P.W. Shor, R.R. Weber, and M. Yannakakis. Fundamental discrepancies between average-case analyses under discrete and continuous distributions: A bin packing case study. In *Proceedings 23th Annual ACM Symposium on Theory of Computing*, pages 230–240, 1991.

[19] E.G. Coffman Jr., M.R. Garey, and D.S. Johnson. Approximation algorithms for bin packing: A survey. In *Approximation Algorithms for NP-hard Problems,* edited by D. Hochbaum, PWS, 1996.

[20] X. Deng and C.H. Papadimitriou. Exploring an unknown graph. In *Proceedings 31st IEEE Symposium on Foundations of Computer Science*, pages 356–361, 1990.

[21] R. El-Yaniv, A. Fiat, R. Karp, and G. Turpin. Competitive analysis of financial games. In *Proceedings of the 33rd Symposium on the Foundations of Computer Science (FOCS'92)*, pages 327–333, 1992.

[22] R.L. Graham. Bounds for certain multi-processing anomalies. *Bell System Technical Journal*, 45:1563–1581, 1966.

[23] E.F. Grove. The harmonic online k-server algorithm is competitive. In *Proceedings of the 23rd ACM Symposium on the Theory of Computation (STOC'91)*, pages 260–266, 1991.

[24] M.-Y. Kao, J.H. Reif, and S.R. Tate. Searching in an unknown environment : An optimal randomized algorithm for the Cow-Path problem. In *Proceedings of the 4th ACM-SIAM Symposium on Discrete Algorithms (SODA'93)*, pages 441–447, 1993.

[25] D.R. Karger, S.J. Phillips, and E. Torng. A better algorithm for an ancient scheduling problem. In *Proceedings 5th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 132–140, 1994.

[26] C. Kenyon, A. Sinclair, and Y. Rabani. Biased random walks, Lyapunov functions, and stochastic analysis of best fit bin packing. In *Proceedings 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 351–358, 1995.

[27] E. Koutsoupias and C.H. Papadimitriou. On the k-server conjecture. *Journal of the ACM*, 42(5):971–983, 1995.

• [28] K. Kursawe and M. Reinstädtler. *OnVis 1.1 : Ein System zur Visualisierung von Online-Algorithmen.* 66123 Saarbrücken, Germany, December 1995. Documentation of the prototype OnVis 1.1 (in German).

[29] N. Reingold, J. Westbrook, and D.D. Sleator. Randomized competitive algorithms for the list update problem. *Algorithmica*, 11:15–32, 1994.

[30] D.D. Sleator and R.E. Tarjan. Amortized efficiency of list update and paging rules. *Communication of the ACM*, 28:202–208, 1985.

[31] D.D. Sleator and R.E. Tarjan. Self-adjusting binary search trees. *Journal of the ACM*, 32(3):652–686, 1985.

[32] A. Subramanian. An explanation of splaying. *Journal of Algorithms*, 20:512–525, 1996.

5 Parallel and Distributed Computing

Our work in parallel and distributed computing over the past two years has been extensive, and many members of the group have contributed to these areas. Our work falls roughly into the following categories:

- PRAM algorithms for fundamental graph and network problems like shortest paths, minimum spanning tree and network flow (Section 5.1).
- Parallel data structures, in particular parallel priority queues, and general techniques, designed with the PRAM as a basis, but with applicability also in other settings (Section 5.2).
- Alternatives to and variations of the PRAM as a model of parallel computation (Section 5.3).
- Basic problems, like routing, sorting and list-ranking, for networks of interconnected computers (Section 5.4).
- General-purpose, model-independent techniques for parallel computation: dynamic load balancing and scheduling (Section 5.5).
- Distributed computing: interprocess communication and coordination, synchronization, resource management (Section 5.6).

The PRAM is a convenient model for the design and analysis of parallel algorithms, and algorithms and techniques developed with the PRAM as the target machine have often been influential in the design of algorithms for other, perhaps more "realistic" models of parallel computation. We continue to work within the PRAM framework, and results concerning specific problems and techniques are described in the following sections. A unifying theme for most of our research has been to obtain algorithms that are not necessarily fast(er), but more efficient in terms of the *total work* performed. PRAM algorithms will also be found elsewhere in this report as natural complements to sequential algorithms. Despite skepticism as to whether a (reasonable approximation of the) PRAM will eventually be built, much of the knowledge accumulated over the past decade is in principle ripe for practical utilization. With this in mind, we have undertaken a project to construct a library, called PAD, of *PRAM Algorithms and Data structures* (Section 9.9).

Parallel algorithms designed directly for specific architectures like meshes, tori, and hypercubes remain an active research area. We have carried out a considerable amount of theoretical and experimental work on algorithms for basic problems for such interconnection networks, in particular for the list-ranking problem, but also for routing problems and sorting. It is worth noting that both within our group and in the (theoretical) community at large, implementations, experiments, and practical evaluation have come to play a much larger role in recent years than what used to be the case.

Another important stream in our research is the study of parallelism and concurrency in general, independently of any concrete model architecture like the PRAM. This is manifest in work (a) on dynamic load balancing, a much-studied problem of relevance irrespective of the underlying parallel system, and (b) on general, distributed computing. In both of these areas we did both theoretical and experimental work, using appropriate simulation tools for the latter. For the load balancing problem we suggested and experimentally studied two new strategies and proposed a novel framework for analyzing the problem. In distributed computing we contributed new results concerning fundamental problems in interprocess communication and coordination, concurrent data objects, synchronization, global state detection, and resource management.

5.1 Parallel Graph and Network Algorithms

Shortest Paths

Investigators: Shiva Chaudhuri and Christos Zaroliagis

The shortest path problem is one of the most fundamental and ubiquitous problems in network optimization. Given an n-vertex digraph G with real edge weights, the shortest path problem asks for paths of minimum weight between vertices in G. In the single-source problem we seek such paths from a specific vertex to all other vertices and in the all-pairs shortest paths (APSP) problem we seek such paths between every pair.

For general digraphs the best parallel algorithm for the APSP problem takes $O(\log^2 n)$ time using $O(n^3)$ work on an EREW PRAM [5]. In the case of planar digraphs there is an $O(\log^4 n)$-time, $O(n^2)$-work EREW PRAM algorithm [3]. An APSP algorithm must output paths between $\Omega(n^2)$ vertex pairs and thus requires this much work and space.

In [1] we consider an alternative and more efficient approach (especially for the case of sparse digraphs); namely, to preprocess the digraph so that subsequently, *queries* can be efficiently answered. A query specifies two vertices and a *shortest path query* asks for a minimum weight path between them, while a *distance query* only asks for the weight of such a path. This approach has already led to better algorithms for finding APSP in outerplanar and planar digraphs [4].

In [1] we give parallel algorithms for the EREW PRAM model of computation that depend on the treewidth of the input graph. Informally, the treewidth is a parameter that shows how close is the structure of a graph to that of a tree. When the treewidth is bounded by a constant, our algorithms can answer distance queries in $O(\alpha(n))$ time using a single processor, after a preprocessing of $O(\log^2 n)$ time and $O(n)$ work, where $\alpha(n)$ is the inverse of Ackermann's function. The class of bounded treewidth graphs contains many natural classes of graphs including (among others) outerplanar graphs, series-parallel graphs, and graphs with bounded bandwidth and cutwidth. To the best of our knowledge, these are the first parallel algorithms which achieve these bounds for any class of graphs except trees. We also give a dynamic algorithm which, after a change in an edge weight, updates our data structures in $O(\log n)$ time using $O(n^\beta)$ work, for any constant $0 < \beta < 1$. Note that the time bottleneck in preprocessing is the computation of the so-called tree-decomposition of the input graph (a decomposition that reflects the tree-like structure of the graph; see also Section 4.2). If an explicit tree-decomposition of the graph is also provided with the input, then the preprocessing time is $O(\log n)$.

We further provide an additional result of independent interest: the single-source shortest path problem (or alternatively, the problem of finding a cycle of total negative weight) in a digraph of bounded treewidth can be solved in $O(\log^2 n)$ time using $O(n)$ work. (The time can be reduced to $O(\log n)$, if an explicit tree-decomposition is provided with the input.) All previous results for this problem required $\Omega(n \log n)$ work, even for the case of outerplanar digraphs.

All of our results are work-optimal w.r.t. the best sequential algorithms for the same problems [2].

References

- [1] S. Chaudhuri and C. Zaroliagis. Optimal parallel shortest paths in small treewidth digraphs. In *Algorithms (ESA'95)*, pages 31–45. Lect. Notes in Comp. Sc. 979, Springer-Verlag, 1995. To appear in *Theoretical Computer Science*, 1997, special issue on ESA'95.

[2] S. Chaudhuri and C. Zaroliagis. Shortest path queries in digraphs of small treewidth. In *Automata, Languages and Programming (ICALP'95)*, pages 244–255. Lect. Notes in Comp. Sc. 944, Springer-Verlag, 1995. Also to appear in *Algorithmica.*

[3] E. Cohen. Efficient parallel shortest-paths in digraphs with a separator decomposition. In *Proc. 5th ACM Symp. on Parallel Algorithms and Architectures (SPAA'93)*, pages 57–67, 1993.

[4] H. Djidjev, G. Pantziou, and C. Zaroliagis. On-line and dynamic algorithms for shortest path problems. In *Theoretical Aspects of Computer Science (STACS'95)*, pages 193–204. Lect. Notes in Comp. Sc. 900, Springer-Verlag, 1995.

[5] Y. Han, V. Pan, and J. Reif. Efficient parallel algorithms for computing all pair shortest paths in directed graphs. In *Proc. 4th ACM Symp. on Parallel Algorithms and Architectures (SPAA'92)*, pages 353–362, 1992.

Minimum Spanning Tree

Investigators: Ka Wong Chong and Christos Zaroliagis

Another fundamental and intensively studied problem in network optimization is the minimum spanning tree problem. Given a connected n-vertex, m-edge undirected graph G with real edge weights, the *minimum spanning tree* (MST) problem is to find a spanning tree of minimum total weight among all spanning trees of G. In sequential computation, almost linear-time algorithms are known for the problem. However, in parallel computing this is far from being the case.

In the parallel context, the MST problem is closely related to the connected components problem in the sense that almost all parallel algorithms for either of the problems use the *hook-and-contract* approach: initially each vertex represents a component by itself. Then, every component C hooks to another component C' by selecting an edge whose one endpoint is in C and the other in C'. After hooking, every (new) component is contracted into a single vertex. The hooking and contraction steps are repeated until there are no more edges connecting different components. While in the connected components algorithms the selection of the hooking edge can be arbitrary, in the MST problem this selection is critical: it has to be the edge with minimum weight. This particular difficulty usually increases the running time and/or the work of the MST algorithms. Further, it has been shown in [8] that the MST problem can be reduced to a connected components problem without an increase in the time; however, the number of processors used (and hence the work) is increased to $m^{1+\varepsilon}$, for some $\varepsilon > 0$.

The results in [2, 7] for the MST problem came therefore as a surprise since they match the corresponding connected components bounds in [2, 6]. Namely, in [2] a CREW PRAM algorithm was given for the MST problem that runs in $O(\log^2 n)$ time and performs $O(n^2)$ work. In [7], this was improved to $O(\log^{3/2} n)$ time and $O(m \log^{3/2} n)$ work on the EREW PRAM. The currently best connected components algorithm, on the EREW PRAM, runs in $O(\log n \log\log n)$ time and performs $O(m \log n \log\log n)$ work [4].

Even on the CRCW PRAM, there is still a gap in the work performed between the best connectivity algorithm and the best MST algorithms. The best connectivity algorithm [5] has an almost-optimal work complexity on the ARBITRARY CRCW PRAM, while the best MST algorithms use the PRIORITY CRCW PRAM model in order to achieve a total work of $O(m \log n)$ [1] or of $O(m \log\log\log n)$ [5].

Our group has worked towards providing better parallel algorithms for the MST problem. More precisely, in [3] we give an EREW PRAM algorithm that runs in $O(\log n \log\log n)$ time and performs $O(m \log n \log\log n)$ work, thus matching the bounds of the best connected components algorithm [4] for this model.

In [9], we give two simple and work-efficient parallel algorithms. Both algorithms perform $O(m \log n)$ work. The first algorithm runs in $O(\log^2 n)$ time on an EREW PRAM, while the second one runs in $O(\log n)$ time on a COMMON CRCW PRAM. The EREW PRAM algorithm is not faster than those in [7, 3], but it is more work-efficient since it avoids the contraction of components (a process that requires sorting). The COMMON CRCW PRAM algorithm is both faster and more work-efficient than the algorithms in [1, 5], since direct simulations of these algorithms on a COMMON CRCW PRAM would either increase the time and/or the total work performed by a logarithmic factor. It is worth mentioning that the EREW PRAM algorithm is very simple and therefore a good candidate for future implementation in the PAD library of PRAM algorithms (Section 9.9).

References

[1] B. Awerbuch and Y. Shiloach. New connectivity and MSF algorithms for shuffle-exchange network and PRAM. *IEEE Trans. on Computers*, 36(10):1258–1263, 1987.

[2] F. Y. Chin, J. Lam, and I.N. Chen. Efficient parallel algorithms for some graph problems. *Communications of the ACM*, 25:9:659–665, 1982.

• [3] K. W. Chong. Finding minimum spanning trees on the EREW PRAM. In *Proc. Int'l Computer Symp, Taiwan*, pages 7–14, 1996.

[4] K.W. Chong and T.W. Lam. Finding connected components in $O(\log n \log\log n)$ time on the EREW PRAM. *Journal of Algorithms*, 18:378–402, 1995.

[5] R. Cole and U. Vishkin. Approximate and exact parallel scheduling with applications to list, tree and graph problems. In *Proc. 27th IEEE Symp. on Foundations of Comp. Sc. (FOCS'86)*, pages 478–491, 1986.

[6] D.B. Johnson and P. Metaxas. Connected components in $O(\log^{3/2}|V|)$ parallel time for the CREW PRAM. In *Proc. 32nd IEEE Symp. on Foundations of Comp. Sc.*, pages 688–695, 1991.

[7] D.B. Johnson and P. Metaxas. A parallel algorithm for computing minimum spanning trees. *Journal of Algorithms*, 19:383–401, 1995.

[8] D.R. Karger. Approximating, verifying, and constructing minimum spanning trees. Unpublished manuscript, 1992.

• [9] C. Zaroliagis. Simple and work-efficient parallel algorithms for the minimum spanning tree problem. *Parallel Processing Letters*, 1997. In print.

Network Flow

Investigators: Jordan Gergov and Torben Hagerup

The *dynamic trees* and the *maximum distance discharge* (MDD) algorithms of Goldberg and Tarjan are among the best sequential algorithms for the fundamental *maximum flow* problem [6]. Given a network G with n vertices and m edges, their time complexities are $O(nm \log(n^2/m))$ and $O(n^2\sqrt{m})$, respectively [6, 3]. If the edge capacities of G are integral and bounded by U, the *excess scaling* maximum flow algorithm of Ahuja and Orlin achieves a running time of $O(nm + n^2 \log U)$ [1].

The best known parallel algorithms for the maximum flow problem are the algorithm of Shiloach and Vishkin and parallel implementations of the excess scaling and maximum distance discharge algorithm that can be executed in $O(n^2 \log(m/n) \log U)$ time using $O(nm \log(m/n) \log U)$ operations on the EREW PRAM, and in $O(n^2 \log n)$ time using $O(n^2\sqrt{m} \log n)$ operations on the CREW

PRAM [7, 8, 1, 5]. The algorithm of Shiloach and Vishkin needs $O(n^2 \log n)$ time and $O(n^3 \log n)$ operations on an EREW PRAM [7, 8].

All these algorithms deviate by a polylogarithmic factor from being optimal w.r.t. the work performed by the corresponding sequential algorithms. While this polylogarithmic deviation may appear insignificant compared to the remaining factor in the operation bound, it should be noted that the factor by which a parallel algorithm deviates from optimal speedup is precisely the minimum number of processors needed for a meaningful execution of the algorithm. If we run any of these parallel flow algorithms on a multicomputer with fewer processors, the corresponding sequential algorithm will be faster.

Our research aimed at the development of algorithmic techniques and tools that reduce this logarithmic deviation or eliminate it altogether [4]. The results obtained are fairly general and may find applications in other parallel network flow algorithms. In particular, they lead to the first parallel implementations with optimal speedup of the corresponding sequential algorithms. For the sake of concreteness, let us concentrate on the MDD algorithm and fix some PRAM model. Further, let $s(n)$, $t(n) \geq \log n$ and $q(n) > \log^{(\Theta(1))}(n)$ be resource bounds such that n positive integers bounded by n can be sorted in $t(n)$ time using $nq(n)$ operations and $s(n)$ space. Then, we obtain an $O(n^2 t(n))$ time parallel maximum flow algorithm that uses $O(n + m + s(n))$ space and performs $O(n^2 \sqrt{m} q(n) + nm \log n)$ operations [4]. As an example, let us take one of the best integer sorting results on the EREW PRAM [2]. Then, except for very dense networks, we get an EREW PRAM maximum flow algorithm that is slower than Goldberg's CREW PRAM algorithm by a factor of about $\Theta(\sqrt{\log n})$, but more efficient by a factor of the same order of magnitude. We show how to keep this efficiency for very dense networks as well at the price of an increase in the time by another factor of about $\Theta(\sqrt{\log n})$.

References

[1] R. K. Ahuja and J. B. Orlin. A fast and simple algorithm for the maximum flow problem. *Operations Research*, 37:939–954, 1989.

[2] S. Albers and T. Hagerup. Improved parallel integer sorting without concurrent writing. In *Proc. 3rd ACM-SIAM Symposium on Discrete Algorithms*, pages 463–472, January 1992.

[3] J. Cheriyan and S.N. Maheshwari. Analysis of preflow push algorithms for maximum network flow. *SIAM J. Comput.*, 18:1057–1086, 1989.

• [4] J. Dedorath, J. Gergov, and T. Hagerup. More efficient parallel flow algorithms. In *Proc. 6th International Symposium on Algorithms and Computation*, pages 234–243, December 1995.

[5] A.V. Goldberg. Processor-efficient implementation of a maximum flow algorithm. *Inform. Process. Lett.*, 38:179–185, 1988.

[6] A.V. Goldberg and R.E. Tarjan. A new approach to the maximum flow problem. *J. Assoc. Comput. Mach.*, 35:921–940, 1988.

[7] Y. Shiloach and U. Vishkin. An $O(n^2 \log n)$ parallel max-flow algorithm. *J. Algorithms*, 3:128–146, 1982.

[8] U. Vishkin. A parallel blocking flow algorithm for acyclic networks. *J. Algorithms*, 13:489–501, 1992.

Approximating Biconnectivity in Parallel

Investigator: Ka Wong Chong

Consider a biconnected undirected graph G. G may contain edges whose removal does not affect the biconnectivity property. This observation gives rise to the problem of finding a minimum biconnected subgraph of a G (a problem arising in the design of communication networks that can tolerate a site failure). It is known that the problem is NP-hard [5]. Nevertheless, simple approximation algorithms, achieving approximation factor of 2, for the problem have been known for a while. In the sequential context, Khuller and Vishkin [8] are the first to give a polynomial-time algorithm for finding a biconnected subgraph with an approximation factor of $\frac{5}{3}$. Later, Garg *et al.* [6] improved the factor to $\frac{3}{2}$.

In the parallel context, the work of Cheriyan *et al.* [1] and Kelsen *et al.* [7] implies an NC algorithm for finding a biconnected subgraph with an approximation factor of 2. The sequential algorithms in [8, 6] do not admit efficient parallel implementation due to their sequential nature. Recently, Chong and Lam [2, 3] designed an NC approximation algorithm which achieves a factor of $\frac{5}{3}+\epsilon$ for any constant $\epsilon > 0$. Finally, we have successfully devised an NC algorithm [4] that match the performance of the sequential counterpart. The algorithm improves the approximation factor of the minimum biconnected subgraph problem to $\frac{3}{2}+\epsilon$ for any $\epsilon > 0$. Our work also implies a new sequential approach which is different from the traditional depth-first-search tree based approach, but which can still achieve an approximation factor of $\frac{3}{2}$.

References

[1] J. Cheriyan, M. Y. Kao, and R. Thurimella. Scan-first search and sparse certificates: an improved parallel algorithm for k-vertex connectivity. *SIAM J. Computing*, 22:157–174, 1993.

[2] K. W. Chong and T. W. Lam. Approximating biconnectivity in parallel. In *Proc. 7th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 363–372, 1995.

[3] K. W. Chong and T. W. Lam. Improving biconnectivity approximation via local optimization. In *Proc. 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 26–35, 1996.

• [4] K. W. Chong and T. W. Lam. Towards more precise parallel biconnectivity approximation. In *Proc. 7th Annual International Symposium on Algorithms and Computation*, pages 223–232, 1996.

[5] M. R. Garey and D. B. Johnson. *Computers and Intractability: A guide to the theory of NP-completeness.* Freeman, San Francisco, 1978.

[6] N. Garg, V. S. Santosh, and A. Singla. Improved approximation algorithms for biconnected subgraphs via better lower bounding techniques. In *Proc. 4th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 103–111, 1993.

[7] P. Kelsen and V. Ramachandran. On finding minimal 2-connected subgraphs. *Journal of Algorithms*, pages 1–49, 1995.

[8] S. Khuller and U. Vishkin. Biconnectivity approximations and graph carvings. *Journal of ACM*, 41:2:214–235, 1994.

5.2 Data Structures and Techniques

Parallel Data Structures for the Single-Source Shortest Path Problem

Investigators: Gerth Stølting Brodal, Jesper Larsson Träff and Christos Zaroliagis

Due to its wide applicability, the single-source shortest path problem has been intensively studied in sequential computation (see e.g., [2]), where optimal or almost optimal algorithms exist especially for the important case of nonnegative edge weights. In the case of general networks, there is an $O(m + n \log n)$ time algorithm (Dijkstra's algorithm implemented using a Fibonacci heap). If in addition the network is planar, then the problem can be solved optimally in $O(n)$ time [11].

On the other hand, in parallel computing there is a certain lack of efficient algorithms: the best algorithm for general networks runs in $O(\log^2 n)$ time performing $O(n^3(\log\log n/\log n)^{1/3})$ work on an EREW PRAM [10], and in the case of planar networks the best algorithm runs in $O(\log^4 n)$ time performing $O(n^{3/2})$ work on an EREW PRAM [6]. One possible reason for this lack could be that most of the emphasis so far has been on obtaining very fast algorithms, i.e., algorithms running in polylogarithmic time. However, in most practical situations, where the number of available processors p is fixed and much smaller than the sizes of the problems at hand, the primary goal is to have a work-efficient rather than very fast parallel algorithm, since the running time will be dominated by the work divided by p.

The fastest work-efficient parallel algorithm for the single-source shortest path problem runs in $O(n\log n)$ time and performs $O(m + n\log n)$ work [9]. This algorithm is based on a simple parallelization of Dijkstra's algorithm in combination with an efficient priority queue data structure, called relaxed heaps. The algorithm performs n iterations and the parallel time spent in each iteration is determined by (processor local) priority queue operations of finding a vertex of minimum distance and deleting an arbitrary vertex, plus the time to find and broadcast a global minimum among the local minima. Either or both of the priority queue operations take $O(\log n)$ time, as does the parallel minimum computation; for the latter $\Omega(\log n)$ time is required, even on a CREW PRAM [7]. Hence, the approach with processor local priority queues does not seem to make it possible to improve the running time beyond $O(n\log n)$. An alternative approach would be to use a parallel global priority queue supporting some form of multi-decrease key operation, but unfortunately none of the known parallel priority queue implementations support such an operation. Consequently, a different idea is required to improve upon the running time.

In [4] a new parallel data structure is presented that speeds up the parallel implementation of Dijkstra's algorithm, by supporting the operations required at each iteration in $O(1)$ time. Using this data structure Dijkstra's algorithm can be implemented to run in $O(n)$ time and $O(m\log n)$ work on a CREW PRAM. The main idea of the new data structure is to perform a concurrent merging of the keys involved in a pipelined fashion. In [5] the data structure is reformulated as a parallel priority queue. As such, this is the first parallel priority queue to support a form of parallel multi-decrease key in constant time.

In the case of planar networks, we make further improvements for the solution of the single-source shortest path problem with nonnegative real edge weights. In [18] a simple parallel algorithm for this problem is given that runs in $O((n^{2\epsilon} + n^{1-\epsilon})\log n)$ time and performs $O(n^{1+\epsilon}\log n)$ work on an EREW PRAM, for any $0 < \epsilon < 1/2$. The strength of our algorithm is its simplicity which makes it easy to implement since it is based on fundamental, well-understood routines that will be available on any parallel machine. Our algorithm can be further improved to run in $O(n^{2\epsilon} + n^{1-\epsilon})$ time while performing $O(n^{1+\epsilon})$ work, if we use as a subroutine the parallel implementation of Dijkstra's algorithm [4] mentioned above. The algorithm in its basic form could be a candidate for future implementation in the PAD library of PRAM algorithms (see Section 9.9).

Parallel Priority Queues

Investigator: Gerth Stølting Brodal

The construction of priority queues is a classical topic in data structures. Recently several papers have also considered how to implement priority queues on parallel machines. In [3] we focus on how to achieve optimal speedup for the individual priority queue operations known from the sequential setting. The operations supported in [3] are all the commonly needed priority queue operations from the sequential setting (MakeQueue, Insert, Meld, FindMin, ExtractMin, Delete, DecreaseKey and Build) and the parallel insertion of k elements (MultiInsert).

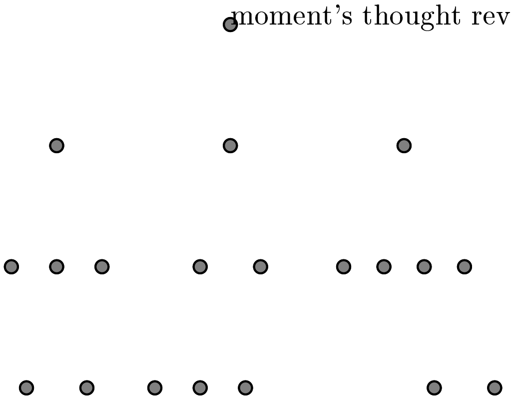
The main result of [3] is that on a CREW PRAM priority queues exist which support FindMin in constant time with one processor, and MakeQueue, Insert, Meld, ExtractMin, Delete and DecreaseKey in constant time with $O(\log n)$ processors. Build is supported in time $O(\log n)$ with $O(n/\log n)$ processors and MultiInsert in time $O(\log k)$ with $O((\log n+k)/\log k)$ processors.

In [3] we also investigated the implementation of a priority queue on a processor array. That a systolic processor array with $\Theta(n)$ processors can implement a priority queue supporting the operations Insert and ExtractMin in constant time is parallel computing folklore, see [13, Exercise 1.119]. Recently Ranade *et al.* [16] showed how to achieve the same bounds on a processor array with only $O(\log n)$ processors. In [3] we describe how the CREW PRAM priority queues can be modified to allow operations to be performed via pipelining. As a result, we get an implementation of priority queues on a processor array with $O(\log n)$ processors, supporting the operations MakeQueue, Insert, Meld, FindMin, ExtractMin, Delete and DecreaseKey in constant time. This extends the result of [16].

Tree Contraction

Investigator: Torben Hagerup

Consider a processor network in the form of a rooted tree and suppose that the root processor generates a message that it wants to broadcast to all other processors in the network. In each step after it receives the message, a processor can transmit it to one and only one neighboring processor. The *broadcast problem* is to determine the *broadcast number* of the network, the number of steps required for the complete broadcast. An example is shown below. Each node v is labeled with the broadcast number of the maximal subtree rooted at v, and the broadcast number of the whole tree is 6.



moment's thought reveals the broadcast numer of a nontrivial subtre investigated the parallel complexity of the broadcast problem and showed (among other results) that the broadcast number of an n-node tree can be computed in $O(\log n)$ time using $O(n)$ operations on a CRCW PRAM. The same result holds for the (as it turns out) closely related problem of computing the register requirements of a given n-operand expression tree.

In this case, the techniques used are probably more interesting than the actual results obtained. The fundamental technique underlying our parallel algorithm is that of *tree contraction*. Tree contraction was introduced by Miller and Reif [14], and subsequent work resulted in a very simple algorithm for the common case of trees that are binary or can be "binarized" [1, 12]. There is no way to "binarize" the operator $\odot$, so that the simple algorithm is not applicable, and even the most general variants of tree contraction provided by Miller and Reif [15] are not sufficiently powerful to deal with the intricacies of $\odot$, so that we had to extend the tree-contraction technique itself to make it apply to our problem. This also motivated us to describe tree contraction in isolation from any particular application. Our interface between tree contraction and its applications is complicated and difficult to use but, as far as we know, the first of its kind. Another byproduct of our work is a *much* simpler proof of the fact that broadcast numbers of trees can be computed in linear sequential time (12 lines versus a significant fraction of the 10-page paper [17]).

References

[1] K. Abrahamson, N. Dadoun, D. G. Kirkpatrick, and T. Przytycka. A simple parallel tree contraction algorithm. *J. Alg.*, 10:287–302, 1989.

[2] R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows.* Prentice-Hall, 1993.

- [3] G. S. Brodal. Priority queues on parallel machines. In *Proc. 5th Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 1097 of *Lecture Notes in Computer Science*, pages 416–427. Springer Verlag, Berlin, 1996.

- [4] G.S. Brodal, J.L. Träff, and C. Zaroliagis. A parallel priority data structure with applications. In *Proceedings of the 11th IEEE International Parallel Processing Symposium (IPPS'97)*, pages 689–693, 1997.

- [5] G.S. Brodal, J.L. Träff, and C. Zaroliagis. A parallel priority queue with constant time operations. Technical Report MPI-I-97-1-011, Max-Planck Institut für Informatik, 1997.

[6] E. Cohen. Efficient parallel shortest-paths in digraphs with a separator decomposition. *Journal of Algorithms*, 21(2):331–357, 1996.

[7] S. Cook, C. Dwork, and R. Reischuk. Upper and lower time bounds for parallel random access machines without simultaneous writes. *SIAM Journal of Computing*, 15(1):87–97, 1986.

- [8] K. Diks and T. Hagerup. More general parallel tree contraction: Register allocation and broadcasting in a tree. In *Proc. 22nd International Workshop on Graph-Theoretic Concepts in Computer Science (WG) Lecture Notes in Computer Science, Vol. 1197*, pages 126–140, 1996.

[9] J. R. Driscoll, H. N. Gabow, R. Shrairman, and R. E. Tarjan. Relaxed heaps: An alternative to Fibonacci heaps with applications to parallel computation. *Communications of the ACM*, 31(11):1343–1354, 1988.

[10] Y. Han, V. Pan, and J. Reif. Efficient parallel algorithms for computing all pair shortest paths in directed graphs. In *Proceedings of the 4th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 353–362, 1992.

[11] P. Klein, S. Rao, M. Rauch, and S. Subramanian. Faster shortest-path algorithms for planar graphs. In *Proceedings of the 26th Symposium on Theory of Computation (STOC)*, pages 27–37, 1994.

[12] S. R. Kosaraju and A. L. Delcher. Optimal parallel evaluation of tree-structured computations by raking. In *Proc. 3rd Aegean Workshop on Computing (AWOC), Lecture Notes in Computer Science, Vol. 319*, pages 101–110, 1988.

[13] F. T. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes.* Morgan Kaufmann, 1992.

[14] G. L. Miller and J. H. Reif. Parallel tree contraction and its application. In *Proc. 26th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 478–489, 1985.

[15] G. L. Miller and J. H. Reif. Parallel tree contraction, Part 2: Further applications. *SIAM J. Comput.*, 20:1128–1147, 1991.

[16] A. Ranade, S.-T. Cheng, E. Deprit, J. Jones, and S.-I. Shih. Parallelism and locality in priority queues. In *Proc. 6th Symposium on Parallel and Distributed Processing*, pages 490–496, 1994.

[17] P. J. Slater, E. J. Cockayne, and S. T. Hedetniemi. Information dissemination in trees. *SIAM J. Comput.*, 10:692–701, 1981.

• [18] J.L. Träff and C. Zaroliagis. A simple parallel algorithm for the single-source shortest path problem on planar digraphs. In *Parallel Algorithms for Irregularly Structured Problems (IRREGULAR'96)*, volume 1117 of *Lecture Notes in Computer Science*, pages 183–194, 1996.

5.3 Alternative Models of Parallel Computation

Investigator: Leszek Gąsieniec

One of the main goals in the study of theory of parallel computation is to develop models close to real machines, but abstract from technical details, thus providing a universal platform for the design and analysis of algorithms.

We have recently investigated a model of a multiprocessor computer with a (partially shared) global memory, called *Parallel Alternating-Direction Access Machine* (PADAM). The model is an abstraction of the architecture of the existing computer ADENA/ADENART [5, 6], which was previously considered in [4] under the name of orthogonal multiprocessor (OMP). In [1] we develop a deterministic and optimal routing scheme for our model. Next we show how to implement efficient access of each processor to all memory modules. Finally we present general PRAM simulations on this model.

Another very important aspect of parallel and distributed computation is to construct fault-tolerant models and algorithms, i.e., delivering an environment which stays highly reliable in the presence of errors (e.g., hardware failures). It is also needed to test how reliable are the already existing models of computations.

ATM (Asynchronous Transfer Mode) networks is a recent but already leading technology for very efficient communication [7, 2]. It is used to accommodate various kinds of data traffic, including voice, video, image, file transfer, etc. Such systems provide true high rate file transfer, video conferencing, HDTV on demand, interface with high-speed LANs, etc.

In designing *good* communication schemes for ATM networks it is very important to give layouts that have short communication paths and reasonably low load (*congestion*) on links of the underlying network. In [3] we consider the problem of constructing layouts for an ATM network consisting of a complete network K_n of n processors in which a certain number of links may fail. Our main goal is to construct layouts which tolerate any configuration of up to f faults and have the least possible congestion. We give upper and lower bounds on the order of magnitude of the congestion. Finally, we consider a random, rather than worst case, fault distribution where links fail independently with constant probability. We deliver highly reliable layouts in K_n, with congestion $O(\log n)$.

References

- • [1] B. S. Chlebus, A. Czumaj, L. Gąsieniec, M. Kowaluk, and W. Plandowski. Parallel alternating-direction access machine. In *Proceedings 21th International Symposium on Mathematical Foundation s of Computer Science (MFCS'96)*, pages 255–266, September 1996.

[2] M. de Prycker. *Asynchronous Transfer Mode: Solutions for Broadband ISDN.* Ellis Horwood Limited, 1993.

- • [3] L. Gąsieniec, E. Kranakis, D. Krizanc, and A. Pelc. Minimizing congestion of layouts for ATM networks with faulty links. In *Proceedings 21th International Symposium on Mathematical Foundations of Computer Science (MFCS'96)*, pages 372–381, September 1996.

[4] K. Hwang, P.S. Tseng, and D. Kim. On orthogonal multiprocessor for parallel scientific computations. *IEEE Transaction on Computers*, 38:47–61, 1989.

[5] H. Kadota, K. Kaneko, I. Okabayashi, T. Okamoto, T. Mimura, Y. Nakakura, A. Wakatani, M. Nakajima, J. Nishikawa, K. Zaiki, and T. Nogi. Parallel computer ADENART - its architecture and application. In *Proceedings of 5th ACM International Conference on Supercomputing, Vol. 1*, pages 1–8, 1991.

[6] H. Kadota, K. Kaneko, Y. Tanikawa, and T. Nogi. VLSI parallel computer with data transfer network: ADENA. In *Proceedings of the International Conference on Parallel Processing, Vol. 1*, pages 319–322, 1989.

[7] D.E. McDysan and D.L. Spohn. *ATM: Theory and Applications.* McGraw-Hill, 1995.

5.4 Problems and Algorithms for Interconnection Networks

Communication on Interconnection Networks

Investigator: Jop F. Sibeyn

Most existing parallel computers have a distributed memory. On such a parallel computer, the processors or processing units (PUs), must communicate through a rather sparse interconnection network, to access data that they do not hold themselves. Therefore, the performance of a parallel computation depends to a large extend on the efficiency of the applied communication algorithms.

In several papers we have analyzed variants of the communication problem. We have concentrated on meshes (parallel computers with a grid-like interconnection network), tori (meshes in which the opposite ends of every one-dimensional submesh are connected), and hypercubes [11].

Probably the most important class of communication problems are the so-called *k-k routing* problems, in which every PU is the source and destination of at most k packets. The special case $k = 1$ is called *permutation routing.* Another fundamental problem is gossiping: the problem in which all locally available information must be made globally known.

For the permutation routing problem on two-dimensional $n \times n$ meshes, optimal (that is, requiring $2 \cdot n - 2$ steps) deterministic algorithms were already known. However, the best of these algorithms still required quite large queue sizes (more than 100). In [21], we show how this queue size can be reduced to 32, and even to 12 if we accept $O(1)$ extra routing steps.

In [8], we present optimal (to within lower-order terms) randomized algorithms for k-k routing on meshes and tori. Previous algorithms were all slower by a constant factor.

As the technology may change, it is also interesting to consider variants of meshes. For example, meshes in which the connections can be reconfigured dynamically. In [7], we show that modifications of the algorithms in [8] are also suited for routing on such networks.

In [19], we consider whether the regular two-dimensional grid, which gives a tessellation of the plane by squares is the 'best' architecture of its type. More precisely, we consider how fast one can route on networks whose underlying graph is a tessellation with triangles or sectangles (and these with diagonals). The conclusion is that there is a continuous trade-off between degree and routing power, and that no single network stands out.

The gossiping problem has been studied in numerous papers, but many of them are of a purely theoretical nature. In [4], we try to give a theoretical analysis that leads to practically useful gossiping algorithms for meshes. The algorithms were implemented on the Intel Paragon, a two-dimensional grid with 138 PUs; this computational support was given by the Nuclear Research Center (Kernforschungsanlage – KFA) in Jülich. Our algorithms performed substantially better than the built-in gossiping routine and simpler strategies.

For k-k routing on hypercubes, Valiant's algorithm performs optimally. On a so-called "full-port" hypercube, in which a PU can communicate over all its links at the same time, however, this requires that the CPU must make a logarithmic (in the total number of PUs) number of routing decisions in constant time. In [5], we analyze under what conditions on k similar performance can be achieved if we assume that a PU can handle only a constant number of packets in constant time.

Basic Problems on Interconnection Networks

Investigator: Ulrich Meyer and Jop F. Sibeyn

For the effective use of a parallel computer, it is important not only that the communication routines are optimized, but that there are also optimized algorithms for a number of basic, frequently occurring problems. Sorting is definitely an example of a problem that should be optimized, but there are several problems of comparable importance.

In [8], we show that on two-dimensional meshes sorting can be solved in essentially the same communication complexity as the underlying routing problem.

A problem in most sorting algorithms, is that they are based on the selection of a rather large sample of the elements. Sending this sample around takes only a lower-order term in total time complexity, but in practical cases, due to the involved constants, this still may be a considerable contribution to the total time consumption. In [17], we consider how the size of this sample can be minimized.

For general k-k routing problems on $n \times n$ meshes, $k \cdot n/2$ is a trivial lower bound, because of the bisection width of the network. However, there are many problems that do not explicitly require that all the data are rearranged. In [9], we consider several problems that can be solved in less than $k \cdot n/2$ steps.

Another basic problem, with several important applications, is that of performing the transpose operation: the data from the PU at position (i, j) of a two-dimensional mesh, have to be exchanged with those from the PU at position (j, i). In [6], we analyze the theoretical and practical sides of this problem: on the theoretical side, we give an optimal algorithm (matching the bisection lower bound), and on the practical side we show how the problem can be solved on the Intel Paragon.

In the theory of parallel computation, list-ranking has tremendous importance. Practically, it can be viewed as a test case for the chances of general-purpose parallel computing: one can argue that there is hardly a problem for which it is more difficult to achieve good speedup. In [18], we consider the theoretical issues of this problem. We show that if every PU holds $k = \omega(1)$ nodes of a set of lists, then the list-ranking problem can be solved optimally (except for lower-order terms) on meshes.

Another (related) class of problems are the problems on graphs. So far, there were efficient algorithms for 'expensive' problems, like all-pairs shortest paths, but for 'cheap' problems, like one-pair shortest path, not much was known. In [22], we give optimal (with respect to a bisection lower bound) algorithms for several such cheap problems. Many results for meshes are summarized in [16].

Implementations of Parallel Algorithms

Investigator: Jop F. Sibeyn

Often there is a big gap between algorithms that are theoretically appealing, and the ones that in practice work best. We have invested a considerable amount of time for an in-depth analysis of the list-ranking problem on interconnection networks, and developed several new algorithms. By now most of these algorithms have been implemented on the Intel Paragon, and in this case, the non-trivial novel algorithms from [20], are more than twice as fast as the earlier ones.

Parallel Sorting

Investigator: Christine Rüb

Sorting is one of the most studied problems in computer science. One reason for this is that it is used in many application as a subroutine. This is also true in the parallel setting, where sorting has also been proposed as an important benchmark for applications of supercomputers [2].

Two of the longest known parallel sorting "algorithms" are two sorting networks due to Batcher [1, 10], namely bitonic merge sort and odd-even merge sort. These two networks can be turned into parallel sorting algorithms where each processor holds more than one element by replacing comparisons between input elements by a split–and–merge procedure [10]. In the past, bitonic merge sort has often been used in the comparison of parallel sorting algorithms and turned out to be the fastest algorithm for small input sizes. Odd-even merge sort has, until recently, not been used.

In [13] (this work was already discussed in the last progress report) it was shown that the average running time of the odd-even merge sort algorithm can be greatly reduced by keeping the communication between the processors at a minimum, i.e., by exchanging only those elements between processors that need to be exchanged. More precisely, it was shown that the average running time of the odd-even merge sort algorithm can be $O((n/p)(\log n + \log p \log(1 + p^2/n)))$ where n is the number of input elements and p is the number of processors used. (The running time of a straight–forward implementation where, in each step, two communicating processors exchange all their elements, is $\Theta((n/p)(\log n + (\log p)^2)))$. In [15] it was then pointed out that the average running time of bitonic merge sort can also be reduced if only the input elements are stored such that the processor with the smaller index always receives the smaller elements.

In [3] and [24] the (modified version of) odd-even merge sort was used for the first time in a comparison of parallel sorting algorithms. In both cases, odd-even merge sort was the fastest algorithm for some range of input size. In [3] the derived bounds from [13] were also used to predict the running time of the implementation. However, it turned out that the predictions were much too pessimistic. In fact, the average running times of odd-even merge sort as well as bitonic merge sort can be bounded by $O((n/p)(\log n + (\log(1 + p^2/n))^2)$, as was shown in [14]. This means that the two sorting algorithms are optimal on the average if $n \geq p^2/2^{\sqrt{\log p}}$. We hope that the bounds from [14] will help to predict the running times of implementations of the two algorithms more accurately in the future.

PRAM Simulation on Meshes

Investigator: Ulrich Meyer, Jop F. Sibeyn

Most existing parallel computers dispose of a distributed memory, but for easy programming, it would be desirable to be able to nevertheless allow programming as if there were a central memory, to which all PUs have access. This requires the simulation of a PRAM (Parallel Random Access Machine) on the underlying architecture. In [12], we consider deterministic PRAM simulation on two-dimensional meshes. The simulation is theoretically as good as it can be, but the involved constants are so large that clearly this is not a very practical approach.

External Memory Computation

Investigator: Jop F. Sibeyn

Even though the size of RAM memory increases dramatically, there are still many situations in which one would like to solve even larger problems. As paging is an extremely expensive operation (in comparison to computation steps), one should carefully design algorithms such that they minimize the number of accesses to secondary storage, in order that problems that require more data than what fits into the main memory can be solved in an acceptable amount of time. This problem area is called *external-memory computation.* One can try to design external-memory algorithms from scratch, but it appears more efficient to develop methods that enable us to select them in a more or less mechanical way from the core of existing algorithms. In [23], we present such a method, which is based on the observation that with a suitable choice of the parameters, good algorithms for BSP (Bulk Synchronous Parallel) computation lead to good external-memory algorithms.

References

[1] K. E. Batcher. Sorting networks and their applications. *Proceedings of AFIPS Spring Joint Computer Conference*, pages 307–314, 1968.

[2] G. E. Blelloch, L. Dagum, S. J. Smith, K. Thearling, and M. Zagha. An evaluation of sorting as a supercomputer benchmark. Technical Report RNR-93-002, NAS Applied Research Branch (RNR), January 1993.

[3] K. Brockmann and R. Wanka. Efficient oblivious parallel sorting on the MasPar MP-1. In *Proc. 30th Hawaii International Conference on System Sciences (HICSS).* IEEE, January 1997.

• [4] B. Juurlink, P. S. Rao, and J. F. Sibeyn. Worm-hole gossiping on meshes and tori. In *Proc. 2nd Euro-Par Conference*, volume 1123 of *Lecture Notes in Computer Science*, pages 361–369. Springer-Verlag, 1996. Full version in Technical Report MPI-I-96-1018, 1996.

• [5] B. Juurlink and J. F. Sibeyn. Fast routing on weak hypercubes: Making the Valiant-Brebner algorithm more practical. In *Proc. 2nd Conference on High-Performance Computing*, pages 755–760. IEEE Press, 1995. Full version in Technical Report 95-23, Rijksuniversiteit Leiden, Vakgroep Informatica, Leiden, Netherlands, 1995.

• [6] M. Kaufmann, U. Meyer, and J.F. Sibeyn. Matrix transpose on meshes: Theory and practice. In *Proc. 11th International Parallel Processing Symposium*, pages 315–319. IEEE, 1997. To appear in *Computers and Artificial Intelligence.*

• [7] M. Kaufmann, H. Schröder, and J. F. Sibeyn. Asymptotically optimal and practical routing on the reconfigurable mesh. *Parallel Processing Letters*, 5(1):81–95, 1995.

• [8] M. Kaufmann and J. F. Sibeyn. Randomized multipacket routing and sorting on meshes. *Algorithmica*, 17:224–244, 1997.

- [9] M. Kaufmann, J. F. Sibeyn, and T. Suel. Beyond the bisection bound: Fast ranking and counting on meshes. In *Proc. 3rd European Symposium on Algorithms*, volume 979 of *Lecture Notes in Computer Science*, pages 75–88, 1995.

[10] D. E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming.* Addison-Wesley, Reading, MA, USA, 1973.

[11] F. T. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes.* Morgan Kaufmann, 1992.

- [12] U. Meyer and J. F. Sibeyn. Simulating the simulator: Deterministic PRAM simulation on a mesh simulator. In F. Breitenecker and I. Husinsky (Eds), editors, *Proc. Eurosim'95*, pages 285–290, 1995.

[13] Ch. Rüb. On the average running time of odd-even merge sort. In *12th Annual Symposium on Theoretical Aspects of Computer Science (STACS'95)*, volume 900 of *LNCS*, pages 491–502, Munich, Germany, 2–4 March 1995. Springer.

- [14] Ch. Rüb. On Batcher's merge sorts as parallel sorting algorithms. Technical Report MPI-I-97-1-012, Max-Planck-Institut für Informatik, 1997.
- [15] Ch. Rüb. On the average running time of odd–even merge sort. *Journal of Algorithms*, 22(2):329–346, February 1997.
- [16] J. F. Sibeyn. Overview of mesh results. Technical Report MPI-I-95-1018, Max-Planck Institut für Informatik, 1995.
- [17] J. F. Sibeyn. Sample sort on meshes. In *Proc. Computing Science in the Netherlands*, pages 217–228, 1995. Full version in Technical Report MPI-I-95-1-012, 1995.
- [18] J. F. Sibeyn. List ranking on interconnection networks. In *Proc. 2nd Euro-Par Conference*, volume 1123 of *Lecture Notes in Computer Science*, pages 799–808. Springer-Verlag, 1996. Full version in Technical report 11/1995, SFB 124-D6, Universität Saarbrücken, Saarbrücken, Germany, 1995.
- [19] J. F. Sibeyn. Routing on triangles, tori and honeycombs. In *Proc. 21st Symposium on the Mathematical Foundations of Computer Science*, volume 1113 of *Lecture Notes in Computer Science*, pages 529–541. Springer-Verlag, 1996.
- [20] J. F. Sibeyn. Better trade-offs for parallel list ranking. In *Proc. 9th Symposium on Parallel Algorithms and Architectures.* ACM, 1997. to appear.
- [21] J. F. Sibeyn, B. S. Chlebus, and M. Kaufmann. Deterministic permutation routing on meshes. *Journal of Algorithms*, 22:111–141, 1997.
- [22] J. F. Sibeyn and M. Kaufmann. Solving cheap graph problems on meshes. Technical Report WSI-95-9, Universität Tübingen, Tübingen, Germany, 1995.
- [23] J. F. Sibeyn and M. Kaufmann. BSP-like external-memory computation. In *Proc. 3rd Italian Conference on Algorithms and Complexity*, volume 1203 of *Lecture Notes in Computer Science*, pages 229–240. Springer-Verlag, 1997. Full version in Technical report MPI-I-97-1001, 1997.

[24] A. Wachsmann and R. Wanka. Sorting on a massively parallel system using a library of basic primitives: Modeling and experimental results. Technical Report TR-RSFB-96-011, Universität-GH Paderborn, May 1996.

5.5 Dynamic Load Balancing

Investigators: Hannah Bast and Torben Hagerup

In spite of the remarkable development of parallel and distributed systems over the past decade, it is typically still left to the programmer of such a system to organize the *scheduling* of the work on the different processing units; a task one would rather like to be taken care of automatically, e.g., by the operating system. As a simple but in many ways representative example consider the multiplication of two large matrices. It is easy to decompose the problem into independent (sub)tasks, namely one for each entry in the resulting matrix. In fact, modern compilers are able to detect such (and other sorts of) parallelism in a problem quite effectively. But which processor should execute which tasks at which time, if we want the problem to be solved as quickly as possible?

Scheduling problems have been intensively studied in the literature but, typically, tasks execution times are considered as fixed and known in advance and the challenge lies in finding a schedule that satisfies certain constraints (such as a prescribed order of execution for certain tasks). As in the example above, however, many practical problems in parallel computation show a distinct rather different structure, in that all tasks are of a similar nature and may be executed in any order. Instead, what complicates the scheduling of these tasks is that their execution times are not exactly predictable, namely for the following reasons: On the one hand, there can be variation in the complexity of the task itself, as in the example of a parallel loop with a conditional statement in its body (depending on the iteration, different code may be executed). On the other hand, the overhead induced by the system while processing a task, caused by factors like memory accesses, calls to the operating system, varying processor latencies, and other "random events" is hard to predict as well.

A very simple strategy, called *static chunking*, assigns about equally many tasks to each processor before the computation starts. With variations in processing times, however, this approach is clearly inappropriate, as it will let processors finish at widely different times. But if the processing of large batches is so unreliable, then why not schedule the work task by task, a strategy called *self scheduling*? This timid approach certainly evens out finishing times as smoothly as possible, but unfortunately incurs an enormous overhead of coordination for the handing out of tasks and, possibly, accessing remote data and communicating back the results of the local computation. Hence, if static chunking is the one extreme (long idle times), self scheduling is the other (large overhead). The goal is to find an assignment strategy that is careful enough to achieve even finishing times, while incurring as little scheduling overhead as possible.

This variant of the scheduling problem has been the focus of considerable research interest over the past two decades [6, 7, 8, 3, 2, 5]. The various strategies proposed, however, were developed on the basis of heuristic arguments and are not supported by rigorous analysis. Except for some of the very simple-minded strategies, the only evidence given for good performances were either run-time measurements of implementations on real machines or simulation results. In particular, due to the different test environments and the lack of theory, the various results were practically incomparable.

In order to remedy the latter, we studied in [4] the performance of most previously known strategies together with a new *Bold* strategy in a common experimental framework. As already suggested by previous authors, the processing time of a batch of size m was modeled as the sum of a fixed overhead and m independent random variables with a common fixed probability distribution. For given parameter settings, the quality of a trial strategy was then evaluated by running a sufficient number of Monte-Carlo simulations. In order to be able to obtain reasonable estimates, efficient schemes for the generation of random variates (i.e., numbers that are random according to some given distribution) were implemented. In this context, a new method for the fast generation of random variates from a specific family of binomial distributions was developed. Our experiments, carried out for a spectrum of parameter settings as representative as possible, revealed major weaknesses of each of the previously known strategies for certain parameter patterns, so that their general and unquestioned use does not appear advisable. The new Bold strategy, on the other hand, performed well throughout, and in fact no other strategy ever achieved a significantly lower (estimate of the) expected finishing time. As for each of the previously known schemes with good performance, however, the Bold strategy also lacks theoretical underpinnings.

In [1], we make an effort to close this long-lasting gap and provide a solid mathematical foundation for the analysis of the scheduling problem. We introduce a new probabilistic model of batch-processing behaviour that, on the one hand, is realistic and general enough to cover many of the considerations by previous authors. At the same time, we are able to determine the exact asymptotic complexity of the scheduling problem according to this model for *all* possible parameter settings. In particular, we introduce the simple *Balancing* strategy that provably always achieves minimal expected finishing time (up to a small constant factor), except when the number of tasks is small, in which case the trivial strategies do sufficiently well. As we were able to keep our analysis on a very abstract and general level, many of the previous strategies became analyzable as well. Indeed, the bounds obtained correspond well with our experimental findings in [4]. As the main result of [1] is formulated on a high level of abstraction, we shall here only mention an important special case. Namely, when the allocation overhead is constant and the time actually used to process a batch is normally distributed with mean proportional to its size, the expected average *wasted time* (that is time not spent on computing) of an optimal schedule is

$$O\Big(\log\log\frac{n}{p}+\sqrt{\log p}\Big).$$

The stated performance is provably achieved by the *Balancing* strategy, while for the *Bold* strategy, the study of [4] merely gives strong evidence for it. The latter strategy, however, appears to attain slightly smaller constant factors. It is easily provable that asymptotically, if batch processing behaves as stated, each of the previously known strategies achieves an exponentially worse expected wasted time.

References

- [1] H. Bast. Provably optimal scheduling of many similar tasks. Manuscript, 1997. Submitted.

[2] L. E. Flynn and S. F. Hummel. A mathematical analysis of scheduling parallel loops in decreasing-size chunks, manuscript. A preliminary version is available as IBM Research Report No. RC 18462, Oct. 1992.

[3] L. E. Flynn, S. F. Hummel, and E. Schonberg. Factoring: A method for scheduling parallel loops. *Communications of the ACM*, 35:8:90–101, 1992.

- [4] T. Hagerup. Allocating independent tasks to parallel processors: an experimental study. In *Proc. Parallel Algorithms for Irregularly Structured Problems (IRREGULAR'96)*, volume 1117 of *Lecture Notes in Computer Science*, pages 1–33, 1996.

[5] S. F. Hummel, J. Schmidt, R. Uma, and J. Wein. Load-sharing in heterogeneous systems via weighted factoring. In *Proc. 8th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA'96)*, pages 318–328, 1996.

[6] C. P. Kruskal and A. Weiss. Allocating independent subtasks on parallel processors. *IEEE Trans. Software Eng.* , 11:1001–1016, 1985.

[7] D. J. Kuck and C. D. Polychronopoulos. Guided self-scheduling: A practical scheduling scheme for parallel supercomputers. *IEEE Trans. Comput.*, 36:1425–1439, 1987.

[8] L. Ni and T. H. Tzen. Dynamic loop scheduling for shared-memory multiprocessors. In *Proc. International Conference on Parallel Processing (ICPP'91)*, pages II247–II250, 1991.

5.6 Distributed Computing

Distributed algorithms are designed for systems which vary in type from heterogeneous message-passing systems, "distributed" over some geographical area, to shared memory multiprocessors, or any type of system that can be described as a set of interconnected processing units. To respect and take advantage of the nature of the system, it is necessary that the algorithms be designed with the following three, very important and challenging issues in mind:

- *concurrency and efficiency*; so that the inherent parallelism of the system is retained and its price/performance ratio is preserved; towards this, it is important to *eliminate bottlenecks*, to *reduce contention* and to ensure *locality*;
- *scalability*; so that the system retains its potential of *incremental growth*;
- *correctness and fault-tolerance*; so that the system executes correctly even if different components operate at very different speeds, and it responds dynamically and consistently to failures and recoveries.

A natural requirement is that any process can complete any operation in a bounded number of its own steps, regardless of the execution speed of the other processes. This property is called *wait-freedom* and not only helps in taking advantage of the inherent parallelism in the system, but also guarantees full resilience to *halting failures.* Difficulties in ensuring all the above properties arise when there is a need for coupling of distributed procedures; such situations introduce interesting trade-offs. Moreover, the time uncertainty due to varying speeds introduces additional, sometimes insurmountable difficulties, since it is not possible for a processing unit to distinguish the case that some component became very slow from the case that the component stopped functioning. In addition, having to cope with transient failures may make things harder, since these include cases where components stop and later resume operation from arbitrary states; if the system is able to recover by itself, it is called *self-stabilizing*; naturally, one might have to trade the system performance in the absence of failures for the stabilization speed.

In our research in this area we have addressed important problems that arise in the design of distributed systems, including interprocess communication and coordination, concurrent data objects, consistent decision making, synchronization, global state detection, allocation and management of resources. Below we give descriptions of the problems on which we focussed, accompanied by summaries of the results achieved.

Concurrent Atomic Objects

Investigators: Marina Papatriantafilou and Philippas Tsigas

A concurrent atomic object —also called *linearizable* object— must ensure that, although accesses to it by different processes may be concurrent, the processes obtain responses that look as if the accesses occurred in some sequential order, consistent with the order of invocations and responses. A natural requirement in the implementations of such objects is *wait-freedom*; as mentioned in the previous paragraph, wait-freedom not only helps in taking advantage of the inherent parallelism in the system, but also guarantees full resilience to *halting failures.*

Constructions of concurrent atomic objects start with simple primitives provided directly by hardware, and successively construct more powerful objects [7]. This procedure leads to a simple and *modular system organization.* Additionally, since wait-freedom eliminates bottlenecks and dependencies between processes, in many cases it offers *scalable* and *real-time* performance guarantees [1].

In our work [5] we take a first step towards systematically exploring the relation between self-stabilization and wait-freedom in concurrent objects for systems in which both processors and memory may suffer temporary or permanent failures. Our contribution is *threefold*: First, we propose a *general definition* of a self-stabilizing wait-free shared memory object. Second, we prove that within this framework one *cannot* construct self-stabilizing wait-free concurrent atomic registers from single-reader single-writer safe bits, which, being the definition of concurrently accessible flip-flops, have been used as the basic building blocks in wait-free concurrent register implementations. The latter leads us to postulate a self-stabilizing *dual*-reader single-writer safe bit, which essentially models a concurrently accessible flip-flop with its output split in two. Using this as a basic building block, we formally prove, as a third contribution, that *simple adaptations* of well known wait-free implementations of shared registers are self-stabilizing. The results show that our general definition is neither trivial nor impractical. An important observation is that the stabilization property of the constructions given here is guaranteed with only a small constant overhead compared to their non-stabilizing counterparts.

Snapshot

Investigators: Philippas Tsigas

A snapshot object enables the "shooting" of an "instantaneous" picture of a set of shared variables by a process called scanner, while each one of them may concurrently and independently be updated by other processes. Snapshot objects are particularly useful for interprocess communication and coordination; they can also be used to solve a variety of other problems, such as monitoring, consistency checking in transaction-based systems, distributed debugging, stable property detection, concurrent timestamping [7].

In [6] we give a linear complexity wait-free algorithm for the single scanner snapshot concurrent object. By using time-stamps as a vehicle, we give a linear, very simple and easy-to-implement solution, with an elegant and easy-to-understand formal correctness proof. Besides, given the word-length achieved by today's technology, it seems that "unbounded" time-stamps can be a fair price for the simplicity attained.

Consensus

Investigators: Marina Papatriantafilou and Philippas Tsigas

The consensus problem involves a set of processes required to reach an irrevocable common decision, even if there are initial differences of opinion about what the decision should be. It is among the most fundamental problems, since it lies in the heart of exclusion, resource allocation, symmetry breaking, synchronization, distributed transaction commit/abort, and, hence, its solutions have direct and important applications in distributed operating, control and database systems. While it is a "non-issue" in an ideal, failure-free environment, it has been proven that in asynchronous systems and using simple communication primitives *not even one* processor failure can be tolerated by a deterministic solution [7]. As expected, the problem received much further attention, the effort being towards classifying system models/architectures and communication/coordination primitives, according to the number of processes that can reach wait-free (i.e. fully resilient in halting failures) consensus using them [7], since it has been proven that this is a key to the synchronization power of a system in the presence of failures.

An important open question has been whether a special multiprocessor system architecture, called *in-phase*, can support deterministic n-process consensus in the presence of faults —in such systems processors may share a common clock pulse and operate in synchronous steps, while the processes are asynchronous due to scheduling, interrupts, preemption, or stop-failures. In [10] we answered the question in the affirmative, by presenting a wait-free n-process consensus protocol for these multiprocessor architectures. The protocol is simple and easy to apply; regarding efficiency, each process reaches decision within at most $n^2/2$ steps of its own in the worst case, and more significantly, within only $n-1$ steps — i.e. without any extra overhead — in the absence of failures.

Counting Networks

Investigators: Marina Papatriantafilou and Philippas Tsigas

Counting networks are networks formed of simple computational units, called *balancers* (a balancer receives tokens from its inputs and alternately forwards them to its outputs, using a toggle mechanism), similarly to the way sorting networks are formed of comparators. They are of big importance in multiprocessor coordination, load balancing, contention resolution, buffer management, priority assignment, especially because they avoid bottlenecks, reduce contention and, more importantly, ensure scalability [2]. Of particular importance for synchronization are the *linearizable* counting networks [4], which produce successive values, in an order that respects the order they were requested. Linearizability significantly raises the capabilities of the network, but at a possible price in network size or synchronization support.

Our work [9] is one of the first steps towards a *systematic study* of *linearizability* in counting networks. We present *necessary* and *sufficient* conditions for linearizability under various timings, in the form of precise inequalities that involve certain timing, as well as *structural parameters* of the network, which are identified by our work and may also be of more general interest. Essentially, our results imply that given a counting network, we can determine classes of systems in which it can be linearizable and classes in which it cannot, by computing simple, shortest-path-like measures of the graph modeling the network. Our results significantly extend previous results on linearizability in counting networks [4, 8]. Of equal importance is the fact that our impossibility proofs use only very simple, *lock-step* and *round-robin* executions, which are common in practice.

Dynamic Bandwidth Management

Investigators: Naveen Garg, Marina Papatriantafilou and Philippas Tsigas

Bandwidth management is an important problem in networking, especially if the bandwidth for each session must be reserved in advance. A natural way to evaluate a bandwidth management policy is to measure its *request satisfiability.* One demanding paradigm of the problem arises in *mobile communication networks* [11], where, in order to avoid signal interference, it is necessary that the bandwidth available for broadcasting in each cell is used in a way that the same frequency is never concurrently used in nearby cells. What makes the problem harder are the needs (i) to cope with temporal variations in channel demand, (ii) to minimize the overhead for each communication session, (iii) to avoid introducing extra hand-overs (i.e., frequency changes during a communication session). Regarding fault-tolerance, a natural way to measure it in this setting is by the *failure locality*, i.e., the size of the network that can be affected by a faulty component.

In [3] we view the problem of dynamic *channel allocation* to mobile communication network cells as a generalized version of *list coloring*: each node (cell) of a graph (signal interference graph) has its own list of colors (free frequencies) and a request number (number of frequencies needed) associated with it. The requirement is to properly color each node with as many colors as it requests, using only colors from its list. Minimizing the list size necessary for the coloring, results in bandwidth allocation that maximizes the number of satisfiable requests. Our first solution guarantees the *best known* bound for request satisfiability, as well as response time that depends only on the degree of the interference graph, which is a local measure; it also ensures failure locality 4. We also show a solution employing randomization, which achieves response time independent of any measure of the graph and optimal failure locality. Besides having complexity measures involving only small constants and offering big improvements in efficiency and fault-tolerance over previous solutions – without introducing any hand-overs –, our protocols are simple to implement and apply in mobile communication networks.

References

[1] J. H. Anderson and S. Ramamurthy. A framework for implementing objects and scheduling tasks in lock-free real-time systems. In *Proceedings of the 17th IEEE Real-Time Systems Symposium*, 1996.

[2] J. Aspnes, M. Herlihy, and N. Shavit. Counting networks. *Journal of the ACM*, 41:1020–1048, 1994.

- [3] N. Garg, M. Papatriantafilou, and Ph. Tsigas. Distributed list coloring: How to dynamically allocate frequencies to mobile base stations. In *Proceedings of the Eighth IEEE Symposium on Parallel and Distributed Processing*, pages 18–25, 1996.

[4] M. Herlihy, N. Shavit, and O. Waarts. Linearizable counting networks. *Distributed Computing*, 9:139–203, 1996.

- [5] J-H. Hoepman, M. Papatriantafilou, and Ph. Tsigas. Self-stabilization in wait-free shared memory objects. In *Proceedings of the Ninth International Workshop on Distributed Algorithms*, pages 273–287, 1995.

- [6] L. Kirousis, P. Spirakis, and Ph. Tsigas. Simple atomic snapshots: A linear complexity solution with unbounded time-stamps. *Information Processing Letters*, 58:47–53, 1996.

[7] N. Lynch. *Distributed Algorithms.* Morgan Kaufmann, 1996.

[8] N. Lynch, N. Shavit, A. Shvartsman, and D. Touitou. Counting networks are practically linearizable. In *Proceedings of the 15th Annual ACM Symposium on Principles of Distributed Computing*, pages 280–289, 1996.

- [9] M. Mavronicolas, M. Papatriantafilou, and Ph. Tsigas. The impact of timing on linearizability in counting networks. In *Proceedings of the Eleventh IEEE International Parallel Processing Symposium*, pages 684–688, 1997.

- [10] M. Papatriantafilou and Ph. Tsigas. Wait-free consensus in in-phase multiprocessor systems. In *Proceedings of the Seventh IEEE Symposium on Parallel and Distributed Processing*, pages 312–319, 1995.

[11] R. Prakash, N. Shivarati, and M. Singhal. Distributed dynamic channel allocation for mobile computing. In *Proceedings of Fourteenth ACM Symposium on Principles of Distributed Computing*, pages 47–56, 1995.

6 Computational Geometry

In the computational geometry group we continued our work both on rather theoretical core problems in geometry and rather applied problems motivated by actual problems in application areas like robotics, geographical information systems, and computer aided manufacturing. Recently, the applied part has gained slightly increased attention in our group while core geometry problems have become a bit less attractive. This corresponds to a general trend in the field of computational geometry to become more applied. The CG task force report and the splitting of the Annual Symposium on Computational Geometry in a theoretical and an applied track (or the workshop on applied computational geometry at FCRC96) are good indicators for this fact. More than by this trend the shift of attention towards more applied geometry in our group was caused by the leave of Michiel Smid.

Finally, we increased our efforts on the implementation of geometric algorithms. Our investigation of implementation issues on geometric algorithms has now reached a state where we can harvest. Furthermore, we are participating in the European CGAL project whose goal is making the large body of algorithms developed in the field of computational geometry available for industrial application.

The next four sections describe the more theoretical work. Our work on proximity problems is given in Section 6.1. Furthermore, in this section intersection problems for groups of objects, so-called generalized intersection problems, are considered. Section 6.2 describes work on deterministic parallel algorithms for computing ϵ-nets and ϵ-approximations, and on using these sampling techniques to algorithms for 1-dimensional lower envelopes and 2-dimensional Voronoi diagrams. A problem related to sampling, the construction of weak ϵ-nets for points on a hypersphere, is discussed in the succeeding section. Section 6.4 collects work on geometric optimization problems. We give algorithms for computing approximate shortest paths that allow a trade-off between query time, preprocessing time and approximation factor. Furthermore, lower bounds for computing approximate TSP-tours are considered.

In Section 6.5 geometric techniques are applied to computing decision trees in machine learning. Sections 6.6 to 6.8 are on geometric problems in the application areas robotics, computer aided manufacturing, and geographical information systems, respectively. First, a velocity planning problem along a given path is investigated. Further work on motion planning is presented in Section 4.5, where online algorithms for several navigation problems in unknown environments are analyzed. Section 6.7 presents two independent approaches to the problem of determining the roundness of a set of points, a core problem in computational metrology. Line simplification, a classical geometric problem, is discussed in Section 6.8. Here the focus is on preserving the topology of the line in a context given by additional points.

Our efforts on the implementation of geometric algorithms are described in Section 9 on software libraries. For completeness, we briefly list the work related to geometry in Section 6.9.

6.1 Proximity and Intersection Problems

Investigators: Prosenjit Gupta and Michiel Smid

Proximity problems deal with distances in a given set of sites in a low-dimensional Euclidean space. Examples are computing or maintaining the closest pair, computing the site nearest to a query point, computing a subset of the sites of size k that are "closest", and computing a sparse graph (a so-called *spanner*) approximating the complete Euclidean graph on the sites. The computational geometry group at MPI has done much relevant work on all these problems. The details of our results have been described already in previous progress reports. Smid [4] has written a survey paper on proximity problems, which will appear in the *Handbook on Computational Geometry.*

Computing Intersections between Convex Polygons

Let $\mathcal{S}$ be a set of r bounded, convex polygons in the plane with a total of n vertices. In [1], Gupta, Janardan and Smid consider the problem of reporting or counting all intersecting pairs of polygons. This is a well-studied problem for the case when each polygon has a small number of vertices; e.g., each polygon is an axes-parallel rectangle. For general convex polygons, however, this problem has not been studied before.

Let I denote the number of intersecting pairs of polygons in $\mathcal{S}$. The challenge is to devise an algorithm for reporting the intersecting pairs of polygons in output-sensitive fashion, in time $O(f(n)+I)$, where $f(n)$ is subquadratic and small. For the more difficult counting problem, we seek an algorithm with running time $O(f(n))$.

The main difficulty is the fact that the boundaries of two convex polygons P and Q may intersect many times. In our problem, however, we count this as only one intersection between P and Q.

Our result is an algorithm counting (resp. reporting) the pairs (resp. report the I pairs) of intersecting polygons in time $O(n^{4/3+\epsilon})$ (resp. $O(n^{4/3+\epsilon}+I)$), for any constant $\epsilon > 0$. This algorithm is based on an interesting characterization of the intersection of two convex polygons in terms of intersecting pairs of trapezoids from their trapezoidal decomposition. We also give an alternative algorithm for the reporting problem, which runs in time $O(n^{4/3}\log^{O(1)} n + I)$. This algorithm is based on a different characterization of the intersection of a pair of convex polygons in terms of their upper and lower chains and their leftmost vertices.

Generalized Intersection Searching Problems

Geometric searching problems [3] arise in a large variety of application areas, such as computer graphics, robotics, VLSI layout design, and databases. In such a problem, a set S of n geometric objects has to be stored in a data structure, such that for any given query object q, we can efficiently report all objects of S that intersect q. The efficiency of such a data structure is typically expressed by its size and query time, where the latter has the form $O(f(n)+K\cdot g(n))$, for some "small" functions f and g. In this expression, K denotes the number of elements of S that intersect the query object.

In many applications, a more general form of searching problem arises: The objects in S come aggregated in disjoint groups and of interest are questions regarding the intersection of q with the groups rather than with the objects. Here, we say that q intersects a group if and only if it intersects at least one object in the group. We will associate with each group a different color and imagine that all the objects in the group have that color. Then, in the *generalized searching problem*, we want to report the distinct colors of those elements of S that intersect q. Note that the generalized problem reduces to the standard one when each color class has cardinality one.

We can solve a generalized searching problem by first determining the objects of S that intersect q and then reading off the distinct colors. However, the query time can be very high since q could intersect $\Omega(n)$ objects but only $O(1)$ distinct colors. For a generalized searching problem, we want to obtain query times that are sensitive to the number C of distinct colors intersected, rather than K. Typically, our goal is to obtain query times of the form $O(f(n)+C)$ or $O(f(n)+C\cdot g(n))$, where f and g are polylogarithmic.

In [2], Gupta, Janardan and Smid consider the problem of transforming generalized searching problems into other problems by *adding a range restriction.* Roughly speaking, this means that if an efficient data structure for a generalized searching problem on d-dimensional objects is known, then the transformation yields an efficient data structure for a similar generalized problem on $(d+1)$-dimensional objects.

As an example, the general transformation, together with geometric transformations, give efficient data structures for the following two problems: Store a set of colored points (resp. fat triangles) in a data structure, such that for any query fat triangle (resp. point) q, we can report the distinct colors of those points (resp. triangles) that are contained in (resp. contain) q. Here, a triangle is called *fat*, if all its angles are at least equal to some fixed constant α.

References

- [1] P. Gupta, R. Janardan, and M. Smid. Efficient algorithms for counting and reporting pairwise intersections between convex polygons. In *Proc. 8th Canad. Conf. Comput. Geom.*, pages 8–13. Carleton University Press, Ottawa, Canada, 1996.
- [2] P. Gupta, R. Janardan, and M. Smid. A technique for adding range restrictions to generalized searching problems. Report MPI-I-96-1-017, Max-Planck-Institut für Inform., Saarbrücken, Germany, 1996.

[3] J. Matoušek. Geometric range searching. *ACM Comput. Surv.*, 26:421–461, 1994.

- [4] M. Smid. Closest point problems in computational geometry. In J.-R. Sack and J. Urrutia, editors, *Handbook on Computational Geometry.* North Holland, 1997.

6.2 Geometric Sampling and Geometric Divide-and-Conquer Algorithms

Investigator: Edgar Ramos

Randomization has been a very successful tool in the design of geometric algorithms during the last ten years. It has been most successful in the design of incremental algorithms in which some geometric structure determined by a set of geometric objects is constructed by adding the objects one by one in an order determined by a random permutation of the set [3, 7]. This approach has had an impact in practice as it has been used for the actual implementation of algorithms.

Our research has been concerned with the approach that uses random sampling to derive divide-and-conquer algorithms. This approach also dates back to the early work in [3]. Its advantage lies in three aspects: The inherent parallelism leads to fast parallel algorithms and to applications in geometric optimization where the *parametric search* technique [6] is used; it is easier to derandomize to produce deterministic algorithms (in comparison to the incremental approach); and the hierarchical decomposition of the space that it usually produces leads to efficient data structures for point location, range searching, ray shooting, etc. Unfortunately, thus far, these results are regarded as merely theoretical because of the size of the constants involved. This is even worse for the deterministic algorithms obtained by derandomization.

Two main parts are distinguished in our work: First, there are general sampling tools, like *nets*, *approximations* and *cuttings* which can be used as building blocks for algorithms. Second, there are specific algorithms for different problems (although there is a general approach in their design, it is necessary to take advantage of problem specific features to obtain efficient algorithms). We have designed general tools for parallel algorithms [4], and obtained parallel algorithms (both deterministic and randomized for fine grained machines) and deterministic sequential algorithms for several problems [10] (previously [8, 9, 1]). For future work, we believe that there is some potential for practical relevance of the randomized divide-and-conquer approach, as there is growing interest in design and implementation of algorithms in parallel coarse grained machines, and for large data sets in which the number of input/output accesses is an important parameter of performance. In both cases parallelism is important, and this cannot be provided by the incremental approach.

In the rest of this section, we describe our recent work and also briefly mention some work in progress.

Parallel Geometric Sampling

A *range space* is a pair $(X, \mathcal{R})$ where X is a finite set and $\mathcal{R}$ is a collection of subsets of X, called the *ranges*. For example, X can be a set of points in $\mathbb{R}^d$ and $\mathcal{R}$ the collection of all sets $X \cap h$ where h is a halfspace. Two main sampling tools are ϵ-nets and ϵ-approximations of range spaces. An ϵ-net for $(X, \mathcal{R})$ is a set $N \subseteq X$ such that for all $R \in \mathcal{R}$, if $|R| > \epsilon|X|$ then $R \cap N \neq \emptyset$. An ϵ-approximation for $(X, \mathcal{R})$ is a set $A \subseteq X$ such that for all $R \in \mathcal{R}$, $||R \cap A|/|A| - |R|/|X|| \leq \epsilon$. For range spaces satisfying certain property – *bounded VC-exponent*, which is satisfied by essentially all the range spaces that appear in geometric applications – $(1/r)$-nets and $(1/r)$-approximations of sizes $O(r \log r)$ and $O(r^2 \log r)$ respectively exist and can be computed deterministically and sequentially in time $O(nr^c)$ where c is a certain constant depending on the range space [5]. Our joint work with M.T. Goodrich [4] describes deterministic parallel algorithms for the construction of $(1/r)$-nets and $(1/r)$-approximations of size $O(r^{1+\delta})$ and $O(r^{2+\delta})$, for any $\delta > 0$, in the EREW PRAM model that run in time $O(\log n)$ while using an amount of work $O(nr^C)$ (the constant C is somewhat larger than the one in the sequential algorithm). Although the sizes are not optimal, this has no effect in the applications. In the more powerful CRCW PRAM model, the construction can be carried out in time $O(\log \log n)$ using the same amount of work (if the definition of approximation is weakened to allow for its verification using approximate counting).

Recently, as part of work still in progress, we have been able to design algorithms that compute $(1/r)$-nets and $(1/r)$-approximations of optimal size (matching the sizes $O(r \log r)$ and $O(r^2 \log r)$). They are however slower, taking time $O(\log^2 n)$ in the EREW PRAM model.

Geometric Divide-and-Conquer Algorithms

Usually, one considers a collection X of n geometric objects in the d-dimensional Euclidean space and is interested in computing a particular region $\mathcal{S}(X)$ determined by those objects. An *ϵ-cutting* is a set of disjoint regions $\mathcal{T}(X)$ that cover $\mathcal{S}(X)$ and such that for each region σ in $\mathcal{T}(X)$ at most ϵn of the objects in X are needed to compute $\mathcal{S}(X)$ inside σ. Cuttings can be efficiently computed, using the construction of an approximation as an intermediate step (a $(1/2r)$-cutting for a $(1/2r)$-approximation of X is a $(1/r)$-cutting for X). Continuing previous work [8, 9, 1], we have developed further applications of the divide-and-conquer paradigm using cuttings.

In [10], we consider the problem of computing the lower envelope (the minimum) of n constant degree algebraic functions of one variable. The lower envelope has size $O(n\beta(n))$, where $\beta(n)$ is a certain nearly constant function, and it can easily be computed in time $O(n\beta(n) \log n)$ by a simple deterministic divide-and-conquer algorithm [11]. We give an alternative simple (modulo a derandomization black box) approach using divide-and-conquer based on cuttings that results in a deterministic sequential algorithm that runs in the same time bound. This algorithm uses *sparse* cuttings, introduced in [2]. The new approach, however, allows us to obtain the following results (using further tools developed in [8, 9, 4]): A deterministic sequential algorithm that is output sensitive and runs in time $O(n \log f)$ if $f \leq n^\epsilon$, or $O(n\beta(f) \log f) = O(n\beta(n) \log n)$ otherwise, where f is the size of the output; a randomized parallel EREW algorithm that runs in time $O(\log n)$ and uses nearly optimal work $O(n\beta^2(n) \log n)$ with high probability; and output sensitive deterministic parallel algorithms.

We use this algorithm as a component in the algorithm in [1] for a restricted class of planar algebraic Voronoi diagrams to obtain: An output sensitive deterministic sequential algorithm that runs in time $O(n \log f)$ where f is the size of the output; an optimal work randomized parallel CREW algorithm that runs in time $O(\log n)$ with high probability; and output sensitive deterministic parallel algorithms using work $O(n \log f)$.

We also consider a restricted subset of the type of optimization problems usually solved using parametric search, in which the algorithm that is executed generically follows a divide-and-conquer approach based on cuttings, and in which the predicates involved (an object intersects a cell) are *linearizable*. We obtain the following results: An algorithm that solves the optimization problem for 1-d lower envelopes using $O(\log n)$ oracle queries; and, using a variation of the algorithm for planar Voronoi diagrams mentioned above, a deterministic algorithm for computing the diameter of a three-dimensional set of n points (furthest distance pair) that runs in time $O(n \log^2 n)$.

Recently, as part of work still in progress, we have looked at applications in the particular case in which the underlying space is 1-dimensional, the objects are points and the ranges are determined by intervals. This has led to faster parallel algorithms for 2-dimensional linear programming and output sensitive 2-dimensional convex hulls in the EREW PRAM model.

References

[1] N.M. Amato and E.A. Ramos. On computing diagrams by divide-prune-and-conquer. In *Proc. 12th Annu. ACM Sympos. Comput. Geom. (SoCG'96)*, pages 672–682, 1996.

[2] B. Chazelle. Cutting hyperplanes for divide-and-conquer. *Discrete Comput. Geom.*, 9:145–158, 1993.

[3] K. L. Clarkson and P. W. Shor. Applications of random sampling in computational geometry II. *Discrete Comput. Geom.*, 4:387–421, 1989.

• [4] M.T. Goodrich and E.A. Ramos. Bounded-independence derandomization of geometric partitioning with applications to parallel fixed-dimensional linear programming. *Discrete Comput. Geom.*, 1997. To appear.

[5] J. Matoušek. Approximations and optimal geometric divide-and-conquer. *J. Comput. Syst. Sci.*, 50:203–208, 1995.

[6] N. Meggido. Applying parallel computation algorithms in the design of serial algorithms. *J. Assoc. Comput. Mach.*, 30:852–865, 1983.

[7] K. Mulmuley. *Computational Geometry: An Introduction Through Randomized Algorithms.* Prentice Hall, 1993.

[8] M.T. Goodrich N.M. Amato and E.A. Ramos. Parallel algorithms for higher-dimensional convex hulls. In *Proc. 35th Annu. IEEE Sympos. Found. Comput. Sci. (FOCS'94)*, pages 683–694, 1994.

[9] M.T. Goodrich N.M. Amato and E.A. Ramos. Computing faces in segment and simplex arrangements. In *Proc. 26th Annu. ACM Sympos. Theory Comput. (STOC'95)*, pages 683–694, 1995.

• [10] E.A. Ramos. Construction of 1-d lower envelopes and applications. In *Proceedings 13th ACM Symposium on Computational Geometry (SoCG'97)*, June 1997. To appear.

[11] M. Sharir and P.K. Agarwal. *Davenport-Schinzel Sequences and Their Geometric Applications.* Cambridge University Press, 1995.

6.3 Weak ϵ-nets for Points on a Hypersphere

Investigators: Phil Bradford and Vasilis Capoyleas

A *weak ϵ-net* for a set of points M, is a set of points W (not necessarily in M) where every convex set containing $\epsilon|M|$ points in M must contain at least one point in W. Weak ϵ-nets have applications in diverse areas such as computational geometry, learning theory, optimization and statistics; see [4], for example. So, estimating the smallest possible size of weak ϵ-nets has attracted considerable attention recently.

In [2] we show that, if M is a set of points quasi-uniformly distributed on a unit sphere S^{d-1}, then there is a weak ϵ-net $W \subseteq \mathbb{R}^d$ of size $O\Big(\frac{d^2 \log(d) k_d}{\epsilon} \log\big(\frac{k_d}{\epsilon}\big) \log\big(\frac{1}{\epsilon}\big)\Big)$ for M, where k_d is exponential in d. A set of points M is *quasi-uniformly* distributed on S^{d-1}, if for any spherical cap $\mathcal{C} \subseteq S^{d-1}$ with $\mathrm{Vol}(\mathcal{C}) \geq c_1/|M|$, we have

$$c_2 \mathrm{Vol}(\mathcal{C}) \leq |\mathcal{C} \cap M| \leq c_3 \mathrm{Vol}(\mathcal{C})$$

for three positive constants c_1, c_2, and c_3. Further, we show that reducing our upper bound by asymptotically more than a $\log(1/\epsilon)$ factor directly implies the solution of a long unsolved problem of Danzer and Rogers, see unsolved-problem **E14** in [3] and also the discussions in [1].

References

[1] N. Alon, I. Bárány, Z. Füredi, and D. Kleitman. Point selections and weak ϵ-nets for convex hulls. *Combinatorics, Probability and Computing*, 1:189–200, 1992.

• [2] P. G. Bradford and V. Capoyleas. Weak ϵ-nets for points on a hypersphere. To appear in *Discrete and Computational Geometry.*

[3] H. T. Croft, K. J. Falconer, and R. K. Guy. *Unsolved Problems in Geometry*, volume 2 of *Unsolved problems in intuitive mathematics.* Springer Verlag, 1991.

[4] J. Matoušek. Epsilon-nets and computational geometry. In J. Pach, editor, Chapter 3 in *New Trends in Discrete and Computational Geometry*, volume 10 of *Algorithms and Combinatorics.* Springer Verlag, 1993.

6.4 Geometric Optimization

Planar Spanners and Approximate Shortest Paths

Investigators: Srinivasa Arikati, Michiel Smid, and Christos Zaroliagis

A fundamental topic in computational geometry is finding shortest (with respect to some L_p metric) obstacle-avoiding paths between any two points s and t in the plane amidst a set of disjoint polygonal obstacles with a total of n vertices. In many applications it is also sufficient to find an obstacle-avoiding path which is not necessarily shortest, but whose length is within a small constant factor c (the *stretch factor*) of the length of the shortest path. Such a path is called a *c-short path.* The goal is to preprocess the obstacle-scattered plane by creating an efficient data structure that enables fast reporting of a c-short path (or its length). The previous approaches for solving the problem either required an $\Omega(n^2)$ preprocessing in order to achieve a polylogarithmic length query time (even for the seemingly simpler case of the L_1 metric), or had a quite high stretch factor ($6 + \epsilon$, where ϵ is an arbitrarily small positive constant) in the case of an $o(n^2)$ preprocessing [5].

In [1], we give a family of algorithms for the above all-pairs short paths problem that achieve an interesting trade-off between the stretch factor, the query time and the preprocessing bounds. Our main results are algorithms that achieve logarithmic length query time, after subquadratic time and space preprocessing. The stretch factor is $1+\epsilon$ for any L_p metric.

Short paths are closely related to sparse graph structures called *spanners*. Given a set S of n points, a *τ-spanner* is a graph G having the points of S as its vertices and such that for any two points s and t of S there is a τ-short s-t path in G; the parameter τ is called the stretch factor of the spanner. The problem of constructing spanners has attracted a considerable amount of attention recently and the main goal is to construct spanners that contain a linear number of edges (and that possibly satisfy some other constraints [3]). If, in the construction of a spanner G, additional vertices are used (i.e., except for those in S), then G is called a Steiner spanner.

In [1], we also consider the problem of constructing a planar spanner, i.e., a spanner that is a planar graph. We prove that a planar L_1 2-spanner among polygonal obstacles can be constructed in $O(n \log n)$ time without using Steiner vertices. This result is not only optimal with respect to time, but also with respect to the stretch factor: we show that there are sets of polygonal obstacles in the plane such that *any* planar L_1 τ-spanner which does not use Steiner vertices has a stretch factor $\tau \geq 2$. In view of the above result, if we want a planar L_1 spanner with a stretch factor less than 2, then we *have to* use Steiner vertices. In this case, we prove that, for any $\epsilon > 0$, a planar Steiner L_1 $(1+\epsilon)$-spanner with $O(n)$ Steiner vertices can be constructed in $O(n \log n)$ time. Surprisingly enough, we (non-trivially) generalize this result to hold for any L_p-metric; i.e., we can construct a planar Steiner L_p $(1+\epsilon)$-spanner with $O(n)$ Steiner vertices in $O(n \log n)$ time. This along with some new results for finding shortest paths in planar graphs constitute the main building blocks in the efficient solution of the all-pairs short paths problem mentioned above.

The Complexity of Geometric Network Problems

Investigator: Michiel Smid

The *Traveling Salesman Problem* (*TSP*) is one of the best known combinatorial optimization problems. In the geometric version of this problem, we are given a set S of n points in $\mathbb{R}^d$, where $d \geq 1$ is a constant. A *tour* is a closed path that visits each point of S exactly once and returns to its starting point. Each edge of such a tour has a *length* that is equal to the Euclidean distance between its endpoints. The *length* of a tour is the sum of the lengths of all its edges. The *TSP* is to compute a tour along the points of S of minimal length. Since this problem is NP-hard for dimension $d \geq 2$ (see [7]), it is natural to consider the weaker problem of designing efficient algorithms that approximate the optimal tour. We call a tour having length at most r times the length of an optimal tour an *r-approximate TSP-tour* .

It is well known that for $d = 2$, a 2-approximate (resp. (3/2)-approximate) *TSP*-tour can be computed in $O(n \log n)$ (resp. $O(n^3)$) time. Recently, Arora [2] gave a polynomial-time approximation scheme for the planar *TSP*. That is, there is an algorithm that for any constant $\epsilon > 0$, computes a $(1+\epsilon)$-approximate *TSP*-tour in $n^{O(1/\epsilon)}$ time.

It is known that for any dimension $d \geq 3$ and any $\epsilon > 0$, a $(2+\epsilon)$-approximate *TSP*-tour can be computed in $O(n \log n + n\,(1/\epsilon)^d \log(1/\epsilon))$ time.

On the other hand, an n-approximate *TSP*-tour can be computed in $O(n)$ time. This follows from the fact that *any* tour is an n-approximate *TSP*-tour.

This leads to the question of determining, for any dimension $d \geq 1$, the complexity of computing an r-approximate *TSP*-tour for sufficiently large values of n and r. In [6], Das, Kapoor and Smid answer this question for algorithms that belong to the algebraic computation tree model. In particular, they prove the following result: Any algorithm that, given a set S of n points in $\mathbb{R}^d$ and a sufficiently large real number $r < n$, computes an r-approximate *TSP*-tour for S, takes

$\Omega(n \log(n/r))$ time in the worst case. In fact, this lower bound even holds in dimension $d = 1$.

The lower bound is proved using Ben-Or's theorem. In [6], it is also shown that the lower bound is tight for constant values $r \geq 2$ and $d = 2$.

These results immediately imply that the complexity of the related problem of approximating the minimum spanning tree of a set of n points in $\mathbb{R}^d$ is equal to $\Theta(n \log(n/r))$.

In [4], Chen, Das, and Smid show that the technique used to prove the lower bound for TSP can be generalized to prove that constructing a τ-spanner for a set of n points takes $\Omega(n \log n)$ time. This lower bound is tight, and it holds for any real number $\tau > 1$; this value may even depend on n.

Maximum Empty Polygon

Investigator: Jop F. Sibeyn

In [8], we give an improved parallel algorithm for the following interesting problem. For a given rectangle containing n points find the largest L-shaped polygon with sides parallel to those of the original rectangle, which contains none of the given points. An L-shaped polygon is an orthogonal polygon with exactly one reflex angle. The improved algorithm, which is based on the parallel prefix idea, runs on an EREW PRAM in $O(\log n)$ time using $O(n^3/\log n)$ processors. This is optimal compared to the best known sequential algorithm.

References

• [1] S. Arikati, D.Z. Chen, L.P. Chew, G. Das, M. Smid, and C. Zaroliagis. Planar spanners and approximate shortest path queries among obstacles in the plane. In *Algorithms (ESA'96)*, pages 514–528. Lect. Notes in Comp. Sc. 1136, Springer-Verlag, 1996.

[2] S. Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. In *Proc. 37th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 2–11, 1996.

[3] S. Arya, G. Das, D.M. Mount, J.S. Salowe, and M. Smid. Euclidean spanners: short, thin, and lanky. In *Proc. 27th ACM Symp. on Theory of Comp. (STOC'95)*, pages 489–498, 1995.

• [4] D. Z. Chen, G. Das, and M. Smid. Lower bounds for computing geometric spanners and approximate shortest paths. In *Proc. 8th Canad. Conf. Comput. Geom.*, pages 155–160. Carleton University Press, Ottawa, Canada, 1996.

[5] D.Z. Chen. On the all-pairs Euclidean short path problem. In *Proc. 6th ACM-SIAM Symp. on Discrete Algorithms (SODA'95)*, pages 292–301, January 1990.

• [6] G. Das, S. Kapoor, and M. Smid. On the complexity of approximating Euclidean traveling salesman tours and minimum spanning trees. In *Proc. 16th Conf. Foundations of Software Technology and Theoretical Comput. Sci.*, volume 1180 of *Lecture Notes in Computer Science*, pages 64–75. Springer-Verlag, 1996.

[7] C.H. Papadimitriou. The Euclidean traveling salesman problem is NP-complete. *Theoretical Computer Science*, 4:237–244, 1977.

• [8] A. J. Preetham, K. Krithivasan, C. P. Rangan, and J. F. Sibeyn. Parallel algorithm for maximum empty L-shaped polygon. In *Proc. 3rd Conference on High-Performance Computing.* IEEE, 1996.

6.5 Geometric Problems in Machine Learning

Investigator: Dimitrios Gunopulos

An interesting problem in Machine Learning is computing decision trees from a given training set of known examples. The traditional approach is to use a greedy divide and conquer algorithm to find a tree that fits the training set perfectly, and then prune it to avoid the problem of overfitting the data. This approach does not give any guarantee on how well the tree will classify new points.

A different approach is to find an optimal tree. We concentrate on the class of binary trees of constant depth. Such trees have finite VC-dimension: it is therefore easy to show that with high probability and if the training set is large enough, a tree that is optimal with respect to the training set will perform well with any other sample. Since a binary tree of constant depth might not suffice to correctly classify all examples in the training set, the optimal tree is the tree that minimizes the number of incorrectly characterized examples. In [1] we introduce geometric techniques to the problem of computing the optimal tree of depth 2. We present an algorithm to find such a tree in time $O(n \log n + en)$ if a tree that classifies at most e training points incorrectly exists, where n is the number of examples.

The problem with this approach is that it is computationally expensive to find the optimal instance for an expressive class of functions. For example, if we want to compute the optimal tree of depth k, the best known algorithm has a $O(n^{k-1} \log n)$ running time. On the other hand, if we use a very simple class, even the optimal instance may not be sufficiently accurate. We present some interesting geometric problems that arise in this context, namely giving approximate and exact algorithms to find the instance that minimizes the number of incorrectly characterized examples of a given training set, for certain geometric families (for example halfspaces, rectangles) in two or three dimensions.

References

- [1] D. Dobkin and D. Gunopulos. Geometric problems in machine learning. In Ming C. Lin and Dinesh Manocha, editors, *Applied Computational Geometry : Towards Geometric Engineering (WACG96)*, pages 121–132. Springer LNCS 1148, 1996.

6.6 Velocity Planning

Investigator: Stefan Schirra

The basic motion planning problem is to determine a motion for a robot from a given source to a desired destination, avoiding collision with any obstacles in the environment [5]. In the dynamic motion planning problem the environment varies over time, cf. [1, 5]. We assume here that the environment is completely known, especially the trajectories of the obstacles are known in advance. Since the collision-freeness of a path depends on time a motion must be specified as a function of time. Thus the velocity along a path must be specified as well.

The intrinsic complexity of dynamic motion planning problems [6] has moved attention to heuristic approaches. A number of these approaches, starting with [3], are based on decoupling path and velocity planning: In a first phase, a path avoiding stationary obstacles is constructed. In the second phase a velocity profile for the path constructed in the first phase is planned, such that collision with the moving obstacles is avoided and no velocity constraint of the robot is violated. Note that decoupled planning might fail to produce a solution although one exists. The second phase problem is also called "fixed path trajectory planning" [2].

In the path planning phase usually shortest paths among the stationary obstacles are constructed. In the case of polygonal obstacles in the plane the resulting path will be a chain of straight line segments. In [4] we consider velocity planning along one piece of such a path: Given a planar environment containing polygonal obstacles in translational motion with constant non-zero speed, a start position S of a point robot Z and a goal position G, a start time t_S, and a bound v_{max} on the maximum speed of Z, determine the speed profile for a shortest motion of Z along the straight line segment SG starting at time t_S, such that the time taken to reach G is minimized, where collision with any of the moving obstacles is avoided and at all stages Z travels with a speed less than or equal to v_{max}.

Since we are looking for a shortest motion, the robot is not allowed to move backwards along the given path. We show in [4] that a velocity profile for a time-minimal shortest length motion along a straight line path among polygonal obstacles moving with constant velocity can be planned in $O(n \log n)$ time, where n is the number of obstacle corners. We apply sweep-line techniques to solve the problem. Previous approaches to this velocity planning problem were based on pruned visibility graphs and had worst-case time complexity $\Theta(n^2)$ [7].

References

[1] K. Fujimura. *Motion Planning in Dynamic Environments.* Springer Verlag, 1991.

[2] K. Kant and S.W. Zucker. Trajectory planning problems I: Determining velocity along a fixed path. In *Proc. of the 7th Int. Conf. on Pattern Recognition*, pages 196–198, 1984.

[3] K. Kant and S.W. Zucker. Toward efficient planning: the path-velocity decomposition. *Int. Journal of Robotics Research*, 5:72–89, 1996.

• [4] K. Krithivasan, A. Rema, S. Schirra, and P.I. Vijaykumar. Velocity planning for a robot moving along the shortest straight line path among moving obstacles. In *Proc. of 8th Canadian Conference on Computational Geometry*, pages 149–154, 1996.

[5] J.C Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, 1991.

[6] J. Reif and M. Sharir. Motion Planning in the Presence of Moving Obstacles. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science*, pages 144–154, 1985.

[7] A. Rema and K. Krithivasan. Path Planning Amidst Movable Obstacles. In *Proceedings of the 3rd International Conference on Automation, Robotics and Computer Vision*, pages 237–241, 1994.

6.7 Computational Metrology

Investigators: Kurt Mehlhorn and Edgar Ramos

Computational metrology is a term used to denote the computational issues of *tolerancing metrology*, the methodology for determining whether the geometric shapes obtained as result of a manufacturing process are within acceptable deviation from the ideal shape. The problems arising in this field have naturally attracted the attention of researchers in computational geometry. Yap has advocated that the contribution of Computational Geometry is not restricted to providing algorithms, but also can help define and solve the questions themselves [4].

A basic problem in metrology is the *roundness problem*, to decide whether a given planar object is "round" within some specified bound. The original algorithmic problem is, given a set S of n points in the plane corresponding to measurements from the boundary, to determine the minimum width w of a circular annulus (for some r, the region between concentric circles of radius r and $R = r + w$) containing S. This problem has already been the focus of substantial research culminating in a randomized algorithm that runs in time $O(n^{3/2+\epsilon})$ for any $\epsilon > 0$ [1]. This was a considerable improvement from early algorithms that run in time $O(n^2)$. However, the algorithm is quite involved and far from practical. This work reflects the approach of simply providing the tolerance metrology practitioners with an algorithm for exactly solving the problem as they formulate it and for the data provided from the measurement process.

Our work on computational metrology has been dedicated in great part on the roundness problem, and is the result of two independent efforts towards more integral contribution to metrology. Ultimately, the success of such endeavor depends on a direct communication with the tolerancing metrology practitioners. The first effort [2] concentrated on proposing a new notion of roundness together with an efficient algorithm for its computation, and also on providing an approximation algorithm for the original notion; it also considered the problems of roundness in 3-dimensional space and flatness in arbitrary dimension. The second effort [3] concentrated in providing a "complete" roundness classification procedure that integrates the measurement step with the computational decision step. In the rest of this section, we briefly elaborate on these works.

Referenced Roundness and Approximated Roundness

The data is a set S of n points in the plane, the result of the measurement process. In [2], we propose the notion of *referenced roundness* as a more appropriate notion of roundness: a *reference radius* r_0 is given, and one is interested in the minimum width w_0 so that S is contained in an annulus of radius r_0 and width w_0 (the region between concentric circles of radius $r_0 - w_0/2$ and $r_0 + w_0/2$). Simple algorithms (both randomized and deterministic) are given for this notion; they run in time $O(n \log n)$. As a byproduct, an approximation algorithm for the original minimum width annulus problem is obtained, under certain natural quality assumption on the data (it fits in certain coarse annulus and is distributed on all angles around a center): for a parameter $\epsilon > 0$, the output width is guaranteed to be within a factor $1 + \epsilon$ of the true width, and the algorithm runs in time $O((1/\epsilon)n \log n)$ (the algorithm first obtains an estimate of the radius corresponding to the minimum width and then runs the referenced roundness algorithms for $O(1/\epsilon)$ different values of the radius).

Other metrology problems are also considered. For the referenced roundness problem in 3-dimensional space, an algorithm is given that runs in time $O(n^2 \log^c n)$ for a small constant c, a problem that had not been considered before. For the flatness problem in arbitrary dimension d (determine the smallest distance between a pair of hyperplanes so that all the points lie on or between them), an approximation algorithm is obtained that runs in time $O(\alpha^d n)$ and produces a result with an absolute error of $\sqrt{d}/\alpha$, for any constant $\alpha > 0$.

A Complete Roundness Classification Procedure

In [3], we consider an object I that is a compact simply-connected subset of the plane with boundary $\mathrm{bd}\, I$. As a "quality measure" *qual*(I) for the roundness of I, we take $\epsilon_0 - w_0$ where w_0 is the referenced roundness of $\mathrm{bd}\, I$ as defined above (actually it is defined somewhat differently but it turns out to be equivalent) and $\epsilon_0 > 0$ is a parameter. Nevertheless, we show that our results extend to other natural quality measures (except that for them a fast $O(n \log n)$ time algorithm is not known). We use the finger probing model in which the measurement device can identify a point in the interior of I and can probe along any ray. The goal is a classification procedure that determines whether *qual*(I) ≥ 0 (that is, whether $w_0 \leq \epsilon_0$) and whose policy is correct with respect to the original object, not with respect to the measurements. For this to be possible, the procedure integrates the steps of measurement and decision and requires a minimum quality assumption for

I: it is contained within certain coarse annulus and it is convex (although the first restriction is quite reasonable, it would be desirable to weaken the second one).

Let c_I be a *center* of I, the center of an annulus that achieves $\mathit{qual}(I)$, and for a point c_0 let S_n consist of the n measurements obtained by probing along rays towards c_0 for the n angles $2\pi i/n$, $i = 0, \ldots, n-1$. The classification procedure is based on the following two results. First, a point c_0 close to c_I can be easily determined; in fact, just 6 probes are needed. Second, using this c_0 as the center for a sample S_n, $|\mathit{qual}(S_n) - \mathit{qual}(I)|$ is $O(1/n)$. As a result, it suffices to determine $\mathit{qual}(S_n)$ for $n = O(|1/\mathit{qual}(I)|)$. Thus, the classification procedure simply doubles n until it is guaranteed that $\mathit{qual}(I) \neq 0$. By the algorithm for reference roundness mentioned above, this requires time $O(|1/\mathit{qual}(I)| \log |1/\mathit{qual}(I)|)$.

References

[1] P.K. Agarwal and M. Sharir. Efficient randomized algorithms for some geometric optimization problems. In *Proceedings 11th ACM Symposium on Computational Geometry (SoCG'95)*, pages 326–335, 1995.

- [2] C.A. Duncan, M.T. Goodrich, and E.A. Ramos. Efficient approximation and optimization algorithms for computational metrology. In *Proceedings 8th ACM-SIAM Symposium on Discrete Algorithms (SODA'97)*, pages 121–130, 1997.

- [3] K. Mehlhorn, T.C. Shermer, and C.K. Yap. A complete roundness classification procedure. In *Proceedings 13th ACM Symposium on Computational Geometry (SoCG'97)*, June 1997. To appear.

[4] C.K. Yap. Exact computational geometry and tolerancing metrology. In D. Avis and J. Bose, editors, *Snapshots in Computational and Discrete Geometry, vol. 3*, 1995. Tech. Rep. SOCS-94.50, McGill School of Comp. Sci.

6.8 Line Simplification with Constraints

Investigator: Stefan Schirra

The line simplification problem is an old and important problem in the generalization (scaling) of geographical maps. It is a well-studied problem in various disciplines including geographical information systems, digital image analysis, and computational geometry. Often the input is a polygonal chain and a maximum allowed error ϵ, and methods are described to obtain another polygonal chain with fewer vertices that lies at distance at most ϵ from the original polygonal chain. Some methods yield chains of which all vertices are also vertices of the input chain, other methods yield chains where other points can be vertices as well. Another source of variation on the basic problem is the error measure that is used. Well known criteria are the parallel strip error criterion, Hausdorff distance, Fréchet distance, areal displacement, and vector displacement.

Our main motivation for studying the line simplification problem stems from reducing the storage space needed to represent a map in a geographical information system. We assume the map is modeled as a subdivision of the plane or a rectangular region thereof. The focus is on the reduction of the complexity of the subdivision. The processing time may be a little higher, but within reason. The description size of the subdivision is a permanent cost in a geographical information system, whereas the processing time is spent only once in many applications. Moreover, in many situations in cartography a high level of detail is unnecessary or even unwanted.

One of the most important requirements of subdivisions for maps is that they must be simple. No two edges of the subdivision may intersect, except at the endpoints. This poses two extra conditions on the line simplification method. Firstly, when a polygonal chain is reduced in complexity, the output polygonal chain must be a simple polygonal chain. Several of the line simplification methods

Figure 6.2: Part of a map of Western Europe, and an inconsistent simplification of the subdivision.

presented so far don't satisfy this constraint [1, 3, 4, 5]. The second condition that needs to be satisfied when simplifying a polygonal chain in a subdivision is that the output chain does not intersect any other polygonal chain in the subdivision. In other words, the simplification method must respect the fact that the polygonal chain to be simplified has a context. Usually the context is more than just the other chains in the subdivision. On a map with borders of countries and cities, represented by polygonal chains and points, a simplification method that does not respect the points can result in a simplified map in which cities close to the border lie in the wrong country. In Figure 6.2, Maastricht has moved from the Netherlands to Belgium, Canterbury has moved into the sea, and near the top of the border between The Netherlands and Germany, two coast lines intersect. Such topological errors in the simplification lead to inconsistencies in geographical information systems.

In [2] we show that both conditions can be enforced after reformulating the problem into an abstract geometric setting and present a nearly quadratic time algorithm for the following line simplification problem: Given a polygonal line, a set of extra points, and a real $\epsilon > 0$, compute a simplification that guarantees (i) a maximum error ϵ, (ii) that the extra points remain on the same side of the simplified chain as of the original chain, and (iii) that the simplified chain has no self-intersections.

References

[1] W.S. Chan and F. Chin. Approximation of polygonal curves with minimum number of line segments. In *Proc. 3rd ISAAC'92*, pages 378–387. Springer LNCS 650, 1992.

• [2] M. de Berg, M. van Kreveld, and S. Schirra. A new approach to subdivision simplification. In *Proceedings ACSM/ASPRS Annual Convention, volume 4: Auto-Carto 12*, pages 79–88, 1995.

[3] D.H. Douglas and T.K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10:112–122, 1973.

[4] D. Eu and G. Toussaint. On approximating polygonal curves in two and three dimensions. *Graphical Models and Image Processing*, 5:231–246, 1994.

[5] Z. Li and S. Openshaw. Algorithms for automated line generalization based on a natural principle of objective generalization. *Int. J. Geographical Information Systems*, 6:373–389, 1992.

6.9 Implementation Issues of Computational Geometry

Investigators: Christoph Burnikel, Rudolf Fleischer, Kurt Mehlhorn, Stefan Schirra, Michael Seel, Michiel Smid, and Christian Uhrig

Only a few of the many algorithms developed over the past two decades in computational geometry found their way into practice. The implementation of computational geometry algorithms is made difficult by the dissimilarity between fast floating-point arithmetic normally used in practice and exact arithmetic over the real numbers assumed in theoretical papers, the lack of explicit handling of degenerate cases in these papers, and the inherent complexity of many (theoretically) efficient solutions.

Partially supported by the CGAL project (cf. Section 15.1.3) we have continued our work on implementation issues of geometric algorithms. LEDA now provides a substantial collection of exact and efficient algorithms for basic geometric tasks, which handles all degenerate cases. Our work concerning the implementation of geometric algorithms is described in Section 9 on software libraries. Besides an implementation of an algorithm for computing the minimum diameter of a set of moving points using parametric search we present our work on efficient and exact arithmetic, geometric program checking, and on geometry kernels, which will be the basis of future implementation efforts.

7 Combinatorial Optimization and Applications

The activity of our group in combinatorial optimization has increased considerably during the last two years. Our research mainly focuses on approximation algorithms and on practical problem solving using branch-and-cut algorithms, including the necessary theoretical polyhedral investigations. We are working on applications in Computational Biology and Physics (see Section 8) and in Automatic Graph Drawing.

7.1 Combinatorial Optimization

Most of our work in this area has concentrated on the design of efficient algorithms for the solution of NP-hard optimization problems. One approach that has attracted significant interest in the theory community is the design of approximation algorithms — algorithms which return provably good solutions. This has been the thrust of activity at AG1 too, with group members involved in designing algorithms for a wide variety of problems. Some of these results, for instance the improved approximation for maximum planar subgraphs, were important breakthroughs while others, like the 3-approximation for the minimum tree spanning k vertices, can be viewed more as a culmination of a long series of research.

Well-designed approximation algorithms perform really well in practice and their worst case bounds, which at 2 or $\log n$ may not be very appealing to the practitioner, are achieved only for pathological instances which almost never arise in practice. Establishing this is another area that we have been involved in and the results obtained from an implementation of an edge-separator algorithm for planar graphs have been very encouraging.

Besides designing approximation algorithms for graph-theoretic problems, we have also been actively involved in scheduling problems. Our work here encompasses both theory and practice. Our achievements include an improved approximation algorithms for minimizing weighted completion time for single machine scheduling with release dates and for dynamic storage allocation. Besides we have also been involved in solving real world scheduling problems using LP based polyhedral approaches.

7.1.1 Approximation Algorithms for Problems on Graphs

Investigators: Ulrich Finkler, Naveen Garg, Pierre Kelsen, Sanjeev Mahajan, and Ramesh Hariharan

Sparsest Cuts and Separators

The problem of embedding a finite metric space in a normed space so as to minimize distortion was first considered by Bourgain [2] who showed that any n-point metric space can be embedded in a normed space (L_1-norm) with a distortion of at most $O(\log n)$. In [6] we use a 'variational approach' to identify the metric which yields the worst possible distortion when embedded using Bourgain's technique. This provides a simple proof to show that the distortion of Bourgain's embedding is actually no more than $2\mathcal{H}(n)$. As observed by Aumann and Rabani and London et.al. this fact implies that the max-flow min-cut ratio for multicommodity flow is no more than $2\mathcal{H}(n)$. We also give a randomized polynomial time algorithm for finding a cut of sparsity no more than $4\mathcal{H}(n)$ times the optimum. Computation of sparsest cuts is a first step in the solution of a host of VLSI-related problems.

Given an undirected graph with edge costs, a separator is a partition of the vertices into sets each containing at least 1/3 of the vertices. The cost of the separator is the sum of the costs of the edges whose end-points are in different sets of the partition and the problem is to find a separator of low cost. This problem of breaking a graph into roughly equal sized pieces by removing a small set of edges has attracted significant attention since it opens up the possibility of using a divide-and-conquer strategy for the solution of several problems on the graph. Many of these problems pertain to planar graphs. However even for planar graphs no polynomial time algorithm is known for computing the minimum separator. The best polynomial time solution for this problem is a 2-approximation algorithm; however its running time is $O(n^6)$ [8].

In [7] we improved the running time of this algorithm to $O(n^5)$ and implemented it in C++ using the LEDA Library. A host of ideas improved the running time significantly in practice so that we could run the algorithm for graphs of up to a 1000 nodes. The quality of the solutions returned by the algorithm was much better than the pessimistic 2-approximation; in 374 of the 435 instances we found optimum solutions and for the remaining the solution was never more than 10.6% away from the lower bound.

Graph Coloring

In [11] we looked at the problem of understanding and applying the technique of *Semi-definite Programming* due to Goemans and Williamson [9] to obtaining approximation algorithms for various NP-Hard problems. Goemans and Williams obtained a randomized approximation algorithm for the MAX-CUT problem. We show how this algorithm can be *derandomized* [11] by applying the method of conditional probabilities to give a polynomial time deterministic algorithm with the same performance ratio.

We also show how to use this method together with other techniques to *color 2-colorable hypergraphs* having n vertices and dimension d using only $O\big(n^{1-1/d}\log^{1-1/d} n\big)$ colors [10]. This is the first polynomial time algorithm that achieves a sublinear number of colors. The algorithm is based on a new technique for reducing degrees in a hypergraph that should be of independent interest. For the special case of hypergraphs of dimension three, we could do better, namely color with only $O(n^{2/9}\log^{\frac{17}{8}} n)$ colors. This result makes essential use of semidefinite programming.

Minimum Tree Spanning k Vertices

Given a collection of points and distances between them, often one is interested in identifying a subset of points that are ‘close’. Well studied measures of ‘closeness’ include the diameter of the point set or the radius of the minimum disc that covers the points. However, in applications where the subset of points need to be connected, a natural measure of ‘closeness’ would be the length of the minimum tree spanning the points. More formally, given a graph with edge lengths we consider the problem of finding a minimum length tree spanning any k-vertices.

The problem is NP-complete and in [5] we gave a 3-approximation algorithm improving on the previously best-known ratio of 35 [1]. The key technical contribution was to introduce the following lower bounding technique: consider an assignment of potentials to vertices such that for any tree containing a root vertex r, the length of the tree is at least as large as the sum of the potentials of the vertices it spans. Then the sum of the k-lowest potentials is a lower bound on the length of a minimum tree that spans k-vertices and contains r. This lower bound is related to the value obtained by considering an **LP**-relaxation of the problem and we also demonstrate an integrality gap of 3 for this relaxation. Our approach also gives a 3-approximation algorithm for finding a minimum tour in a graph that visits any k-vertices.

Maximum Planar Subgraphs

A popular technique in graph drawing is to draw first a planar subgraph, i.e. a graph that can be drawn in the plane without crossing edges, and then add the remaining edges so as to minimize the number of crossings. Given a graph G, the *Maximum Planar Subgraph* problem is to find a planar subgraph of G with the maximum number of edges.

The *Maximum Planar Subgraph* problem is NP-complete and in [12] we develop valid inequalities for the corresponding polytope and present a branch-and-cut approach to the problem. Consider the trivial algorithm which outputs a spanning tree of the graph. While the spanning tree has $n-1$ edges, any planar subgraph has at most $3n-6$ edges; this implies a $1/3$-approximation ratio. Finding a maximal planar subgraph does not achieve a better worst case approximation ratio than a spanning tree [4].

Improving upon this trivial $1/3$-approximation algorithm remained a tantalizing open problem for a long time. In [3] two new approximation algorithms achieving an approximation ratio exceeding $1/3$ are presented. The higher approximation ratio is $4/9 = 0.\bar{4}$ and is achieved by an algorithm which invokes an algorithm for the graphic matroid parity problem as a subroutine and which runs in time $O(m^{3/2} n \log^6 n)$. A greedy variant of this algorithm has an approximation ratio $7/18 = 0.3\bar{8}$ and runs in linear time on graphs of bounded degree.

The key observation is that maximal planar subgraphs with less edges have long chordless-cycles. Thus the two approximations compute a *triangular structure*, which is a special planar graph composed of triangles (the smallest possible chordless-cycle). The higher approximation ratio is achieved by computing the *maximum triangular structure* as against the *maximal triangular structure* which can be obtained by a greedy algorithm.

References

[1] A. Blum, R. Ravi, and S. Vempala. A constant factor approximation for the k-MST problem. In *Proceedings, ACM Symposium on Theory of Computing*, pages 442–448, 1996.

[2] J. Bourgain. On Lipschitz embedding of finite metric spaces in Hilbert space. *Israel Journal of Mathematics*, 52(1):46–52, 1985.

• [3] G. Călinescu, C. Fernandes, U. Finkler, and H. Karloff. A better approximation algorithm for finding planar subgraphs. In *Proceedings 7th ACM-SIAM Symposium on Discrete Algorithms*, pages 16–25, January 1996. To appear in *Journal of Algorithms.*

[4] M.E. Dyer, L.R. Foulds, and A.M. Frieze. Analysis of heuristics for finding a maximum weight planar subgraph. *European J. Oper. Res. 20, 1*, pages 102–114, 1985.

• [5] N. Garg. A 3-approximation for the minimum tree spanning k vertices. In *Proc. IEEE Symposium on Foundations of Computer Science (FOCS'96)*, pages 302–309, 1996.

• [6] N. Garg. Approximating sparsest cuts. Technical Report MPI-I-97-1-002, Max-Planck-Institut für Informatik, 1997.

• [7] N. Garg and C. Manß. Evaluating a 2-approximation algorithm for edge-separators in planar graphs. Technical Report MPI-I-97-1-010, Max-Planck-Institut für Informatik, 1997.

[8] N. Garg, H. Saran, and V.V. Vazirani. Finding separator cuts in planar graphs within twice the optimal. In *Proceedings, IEEE Symposium on Foundations of Computer Science*, pages 14–23, 1994.

[9] M. Goemans and D. Williamson. 0.878 approximation algorithms for Max Cut and Max 2SAT. In *Proceedings 26th Annual ACM Symposium on Theory of Computing*, pages 422–431, 1993.

• [10] R. Hariharan, P. Kelsen, and S. Mahajan. Coloring 2-colorable hypergraphs with a sublinear number of colors. In *Proc. Scandinavian Workshop on Algorithm Theory (SWAT'96)*, pages 41–52. Lect. Notes in Comp. Sc. 1097, Springer-Verlag, 1996.

• [11] R. Hariharan and S. Mahajan. Derandomizing semidefinite programming based approximation algorithms. In *Proceedings of the 36th IEEE Symposium on Foundations of Computer Science*, pages 162–169, 1995.

[12] M. Jünger and P. Mutzel. Maximum planar subgraphs and nice embeddings: Practical layout tools. *Algorithmica*, 16(1):33–59, 1996. Special Issue on Graph Drawing.

7.1.2 Scheduling Problems and Polyhedral Combinatorics

Investigators: Jordan Gergov and Yaoguang Wang

Single Machine Scheduling

Consider a one machine scheduling problem with n jobs each having a weight, a processing time and a release date. The objective is to minimize the sum of the weighted completion times of the jobs. In [6] we derive a new $O(n \log n)$ algorithm which for any $\beta \in [0,1]$ finds a schedule whose weighted completion time, resp. makespan are within $1 + \frac{1}{\beta}, resp.\ 1 + \beta$ of the minimum possible, where makespan refers to the time at which all jobs are completed. Our algorithm is based on a canonical decomposition of the optimum preemptive schedule. For the case of integer processing times and release dates, the algorithm has the bicriteria performance guarantee $\left(2 - \frac{1}{p_{max}}, 2 - \frac{1}{p_{max}}\right)$, where p_{max} is the maximum processing time.

Goemans [4] suggested an algorithm Best_α for this problem and conjectured that it had a performance guarantee of $e/(e-1)$. In [6] using randomization we show that Best_α finds a schedule with weighted completion time no more than 1.7735 times the minimum.

The following research was motivated by a personnel scheduling problem from a big company (BASF). We are given a set of jobs with precedence relations among them. Associated with each job is a positive processing time p_j and *job profile* $(l_1, l_2, \ldots, l_{p_j})$, where l_j denotes the resource requirement of the job at time j. A *feasible* schedule is a job sequencing order which obeys the precedence constraints and at any time t the total resource used by the processing jobs does not exceed the available resource R_t. The objective is to find a schedule with the minimum makespan. We investigated various formulations and did some computational experiments.

We also performed an in-depth theoretical, algorithmic and computational study of a linear programming relaxation to the precedence constrained single machine scheduling problem $1|\text{prec}|\sum_j w_j C_j$ where the objective is to minimize a weighted sum of job completion times. On the theoretical side, we study the structure of tight parallel inequalities in the LP relaxation and show that *every* permutation schedule which is consistent with the so-called Sidney's decomposition has total cost no more than twice the optimum. On the algorithmic side, we provide a parametric extension to Sidney's decomposition and show that a finest decomposition can be obtained by essentially solving a parametric minimum cut problem. Finally, we report results obtained by an algorithm based on these developments on randomly generated instances with up to 2,000 jobs.

Dynamic Storage Allocation

The objective of the *dynamic storage allocation* (DSA) problem is, given a set of requests for blocks of contiguous memory cells as well as allocation and liberation time for each block, to compute a memory allocation such that the maximum address of a used memory cell is minimized. The dynamic storage allocation problem comes in three flavors: the NP-hard *off-line DSA* (the whole input is known in advance), *on-line DSA* (the size of a block is known at the time of its allocation), and *on-line DSA with known life times* (the size and life time of a block are known at its allocation time).

The two previous approximation algorithms for off-line DSA given independently by Kierstead and Slusarek are based on on-line coloring of interval graphs and have approximation ratios of 6 and 80. In [3], we give an affirmative answer to the important open question of whether the approximation ratio of DSA can be improved below the bound implied by on-line coloring of interval graphs. Our approach is based on the novel concept of a 2-allocation in which a memory cell can, at any point in time, belong to at most two blocks. We then design an efficient transformation of a 2-allocation to a memory allocation that uses 5/2 times as much space thereby obtaining a 5-approximation.

Given an instance of DSA, denote by M, k and n the maximum number of simultaneously occupied memory cells, the maximum number of blocks simultaneously in storage, and the maximum size of a block, respectively. We show that the competitive ratio of on-line DSA is $\Theta(\log(2+kn/M))$, improving the best previous result [5]. For $n = O(M^{1-\epsilon})$ where ϵ is an arbitrary small positive real number, the competitive ratio of on-line DSA is $\Theta(\log n)$. We show that the competitive ratio of existing algorithms [1] for on-line DSA with known life times is also $\Theta(\log n)$ and propose a new algorithm with constant competitive ratio, provided that the life time of a block is polynomial in M. We have used our ideas to implement an experimental storage allocator as proposed in [1, 7].

Polyhedral Combinatorics

The optimal k-restricted 2-factor problem consists of finding, in a complete undirected graph K_n, a minimum cost 2-factor (subgraph having degree 2 at every node) with all components having more than k nodes. The problem is a generalization of the TSP, and is equivalent to it when $\frac{n}{2} \leq k \leq n-1$. In [2] we study the k-restricted 2-factor polytope. We present a large class of valid inequalities, called bipartition inequalities, and describe some of their properties; some of these results are new even for the TSP polytope. For the case $k = 3$, the triangle-free 2-factor polytope, we derive a necessary and sufficient condition for such inequalities to be facet inducing.

References

[1] L.L. Beck. A dynamic storage allocation technique based on memory residence time. *Comm. ACM*, 25(10):714–724, 1982.

• [2] W.H. Cunningham and Y. Wang. Restricted 2-factor polytopes. Technical Report MPI-I-97-1-006, Max-Planck-Institut für Informatik, 1997.

• [3] J. Gergov. Approximation algorithms for dynamic storage allocation. In *Proceedings Europ. Symp. Alg.*, pages 52–61, September 1996.

[4] M. Goemans. Improved approximation algorithms for scheduling with release dates. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms*, pages 591–598, 1997.

[5] M.G. Luby, J. Naor, and A. Orda. Tight bounds for dynamic storage allocation. In *Proceedings 5th ACM-SIAM Symposium on Discrete Algorithm*, pages 724–732, January 1994.

• [6] Y. Wang. Bicriteria job sequencing with release dates. Technical Report MPI-I-97-1-005, Max-Planck-Institut für Informatik, 1997.

[7] P.R. Wilson, M.S. Johnstone, M. Neely, and D. Boles. Dynamic storage allocation: A survey and critical review. In *Proc. International Workshop on Memory Management*, pages 1–116, September 1995.

7.2 Automatic Graph Drawing

About $2\frac{1}{2}$ years ago, we have started to build, within AG1, a group that works on graph drawing. Graph drawing is a new evolving field within computer science that deals with the problem of (automatically) drawing graphs in a nice way (under various aesthetic criteria). This has significant applications in practice, since graphs are widely used to represent structures in many fields such as economics (entity-relationship diagrams), social sciences, mathematical and computer sciences (algorithm animation, databases). A good visualization of structural information allows the reader to focus on the information contents of the diagram.

Graph drawing methods (in two dimensions) can roughly be classified into four different types: spring-embedder methods, hierarchical methods, orthogonal methods, and methods using planarization. While a lot of software exists for the first three methods, there are almost no implementations for the latter one, although in many applications this method leads to the most pleasant drawings. The reason for this is that a lot of knowledge about planarity testing, embedding and planarization is needed and that many complicated data structures and algorithms need to be implemented. Given the strong expertise of our group on such issues, we have naturally focussed our research on graph drawing using planarization.

There are still many unsolved theoretical and practical questions. The drawing method using planarization transforms a given non-planar graph into a planar graph, and then uses planar graph drawing algorithms to draw the graph. Although, theoretically, many questions seem to be solved, when looking into practice, some of these solutions are of no practical value, e.g., for straight-line drawing algorithms (see below).

Hierarchical graph drawing methods, in particular, the layer-based methods, are amongst the most commonly used. Here, in a first step, the vertices are partitioned into layers, and in a second step, the vertices within the layers are permuted such that the number of crossings is small. We are focussing on the second step, which is the crucial one. Our experiments on k-layer crossing minimization have shown that the practitioners' methods of iteratively solving 2-layer crossing minimization lead to solutions with crossing numbers far away from the optimum. Our research aims at improving the drawings generated by these layer-based methods significantly.

It turned out that certain problems, even though NP-hard, can often be solved to optimality in short computation times via branch-and-cut methods. All of our algorithms are implemented using the LEDA library; also, some of them are implemented using the ABACUS framework (a framework for implementing algorithms using the branch-and-cut approach). We will provide our implementations of data structures, tools and algorithms in the form of the AGD (Algorithms for Graph Drawing) Library as a LEDA Extension Package to any interested user. Besides this, we cooperate with outside researchers, partly in industry, on various practical projects.

7.2.1 Combinatorial Algorithms for Planar Graphs

Investigators: Christoph Hundack, Kurt Mehlhorn, and Petra Mutzel

A great deal of the mathematical theory of graphs and graph drawing focuses on planar graphs. An essential prerequisite for this is planarity testing, to which part of our research is devoted.

Many graph drawing algorithms for planar graphs are restricted to certain planar graph classes, such as triconnected planar graphs or planar graphs with maximum degree four. In our research we extended some of these algorithms to a wider class of graphs. We consider straight-line algorithms and related algorithms that are based on a data structure called canonical ordering for triconnected planar graphs. We generalized this data structure to a new canonical ordering for biconnected planar graphs such that it is now possible to use several planar graph drawing algorithms for biconnected planar graphs.

Moreover, we have considered the orthogonal drawing algorithm by [12] which produces a bend-minimal drawing for the given embedding of a 4-planar graph using network-flow techniques. We have defined a new model for quasi-orthogonal planar graph drawing which is an extension of the original bend minimization algorithm. The survey [4] contains detailed descriptions of the mentioned algorithms.

Planarity Testing and Embedding

Existing algorithms for planarity testing and embedding can be grouped into two principal classes. Either, they run in linear time, but at the expense of complex algorithmic concepts or data structures, or they are easy to understand and implement, but require more than linear time.

Based on a new graph decomposition we [7] developed a simplified linear time algorithm for simultaneous planarity testing and construction of a combinatorial embedding. A DFS–based preprocessing splits up the graph into disjoint components and generates a partial embedding of the graph. Within each step, this partial embedding is enlarged by one component; thereby the planarity test is reduced to solve an easy bipartition problem.

In case of non-planarity a Kuratowski subgraph is returned. Experimental results have shown that this algorithm is among the fastest existing ones. The decomposition technique can be applied to accelerate or simplify a number of other graph algorithms, e.g., for determining triconnected components.

Straight-Line Planar Drawing Algorithms

A classical result independently established by various mathematicians, see e.g., [13], shows that every planar graph admits a planar straight-line drawing. However, the early approaches placed the vertices exponentially close together. In 1988, the first algorithms appeared using layout area of only polynomial size when the coordinates of the vertices are integral [3, 11]. Both algorithms work for triangulated planar graphs. The algorithm in [3] has two steps. First, the vertices are sorted according to an ordering called canonical ordering, and then, the vertices are placed incrementally on grid points. Any planar graph can be augmented to a planar triangulated graph by adding artificial edges. Since the newly added edges are considered in the drawing algorithm, but do not appear in the final drawing, the actual drawing appears strange. Therefore, in practice, algorithms are needed that draw the given graph directly without adding too many artificial edges.

Kant [8] defined a canonical ordering for triconnected planar graphs. Since he orders sets of vertices rather than single vertices, the algorithm is able to produce convex straight-line drawings of triconnected planar graphs in linear time. We [6] have generalized the canonical ordering to biconnected planar graphs, that is, we have defined a new canonical ordering for biconnected planar graphs, and we have given an algorithm for constructing the ordering. Hence, for the first time it is possible to use the algorithms given in [8, 2] to produce straight-line planar drawings of biconnected planar graphs. Our implementations of the above algorithms are useful in practice (see Figure 7.3(a) and (b)).

The Mixed-Model Algorithm

Although the drawings produced by the straight-line algorithms have a relatively good vertex resolution, they are not very readable, because of two reasons: the angles between adjacent edges can get quite small, and the layout area is still too big in practice. These two significant problems can be overcomed by allowing bends in the edges. Kant [8] modified the above algorithm to obtain a mixed-model algorithm. This algorithm constructs a polyline drawing of a triconnected planar graph such that the size of the minimal angle is at least $\frac{2}{d}$ radians, where d is the maximum degree of a vertex. The algorithm proceeds similar to the straight-line algorithms. In a first step, the vertices are ordered according to the canonical ordering, and in the second step, the vertex boxes are placed incrementally at grid points. Roughly speaking, a vertex box consists of the vertex plus the first parts of the incident edges, that are regularly distributed around the vertex.

In [6], we came up with a variation of Kant's algorithm for biconnected planar graphs using our new canonical ordering for the straight-line algorithms mentioned above. To our knowledge, we produced the only implementation of the mixed-model algorithm. The drawings are aesthetically pleasing, since they are almost orthogonal, the non-orthogonal lines are only short, and the minimal angles are quite big. Figure 7.3(c) shows the same graph as in Figure 7.3(a) drawn with our algorithm. The grid size of the straight-line drawing is 42 × 21, whereas the size for the mixed-model drawing is 12 × 10.

Figure 7.3: Sparse graph drawn with the algorithms (a) in [3] (original) (b) in [6] (c) in [8, 6] (d) in [9, 10]; by allowing bends, the layout area gets smaller leading to a better resolution.

Quasi-Orthogonal Drawing with Few Bends

Orthogonal drawings are among the most pleasant ones. Naturally, such drawings exist only for graphs with maximum degree four. Tamassia [12] has given an algorithm that produces a bend-minimal orthogonal drawing for a planar graph with a fixed combinatorial embedding. Since in practice, also vertices with high degree occur, several efforts have been made to extend the bend-minimization algorithm for high-degree graphs, e.g., [1, 5]. However, none of them leads to satisfying results. Hence, we decided to define a new model. We [10, 9] have introduced the concept of quasi-orthogonal grid embeddings which allow certain edges to run locally between grid points in order to cope with graphs that have an arbitrarily high degree. Our new model extends the original bend minimization algorithm in the sense that a graph with maximum degree four is drawn as in the original algorithm. Moreover, our model requires that the sizes of the vertices cannot be changed by the algorithm. We have developed a ring-model algorithm which produces such a quasi-orthogonal drawing. Our implementation of the ring-model algorithm shows that for some graph classes, our new model is much better compared to the previous models, in particular, if there are only few vertices with high degree (see Figure 7.3(d)). In many cases however, further improvements are necessary.

References

[1] C. Batini, E. Nardelli, and R. Tamassia. A layout algorithm for data-flow diagrams. *IEEE Trans. Softw. Eng.*, SE-12(4):538–546, 1986.

[2] M. Chrobak and G. Kant. Convex grid drawings of 3-connected planar graphs. Technical Report RUU-CS-93-45, Dept. of Comp. Sci., Utrecht University, 1993.

[3] H. De Fraysseix, J. Pach, and R. Pollack. How to draw a planar graph on a grid. *Combinatorica*, 10(1):41–51, 1990.

• [4] P. Eades and P. Mutzel. *Graph Drawing Algorithms*, CRC Handbook of Algorithms and Theory of Computation, Chapter 23, M. Atallah (Ed.). CRC Press, 1997. To appear.

[5] U. Fößmeier and M. Kaufmann. Drawing high degree graphs with low bend numbers. In F. J. Brandenburg, editor, *Graph Drawing (Proc. GD '95)*, volume 1027 of *Lecture Notes in Computer Science*, pages 254–266. Springer-Verlag, 1996.

• [6] C. Gutwenger and P. Mutzel. Grid embedding of biconnected planar graphs. Extended Abstract, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1997.

• [7] C. Hundack, K. Mehlhorn, and H. Stamm-Wilbrandt. Efficient planarity testing and embedding. Manuscript, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1997.

[8] G. Kant. Drawing planar graphs using the canonical ordering. *Algorithmica*, 16(1):4–32, 1996.

• [9] G. Klau. Quasi-orthogonales Zeichnen planarer Graphen mit wenigen Knicken. Diplomarbeit, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1997.

• [10] G. Klau and P. Mutzel. Quasi-orthogonal drawing of planar graphs. Manuscript, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1997.

[11] W. Schnyder. Embedding planar graphs on the grid. In *Proc. 1st ACM-SIAM Sympos. Discrete Algorithms*, pages 138–148, 1990.

[12] R. Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM J. Comput.*, 16(3):421–444, 1987.

[13] K. Wagner. Bemerkungen zum Vierfarbenproblem. *Jahresbericht der Deutschen Mathematiker--Vereinigung*, 46:26–32, 1936.

7.2.2 Graph Drawing Using Planarization

Investigators: Ulrich Finkler, Christoph Hundack, Kurt Mehlhorn, Petra Mutzel, and Thomas Ziegler

Although planar graphs are the main subject of the mathematical theory of graph drawing, visualization systems, however, must in general deal with graphs which are not planar. To exploit the theory of planar graphs and methods for drawing planar graphs, it is necessary to planarize non-planar graphs, that is, to transform them to planar graphs. Surveys of our research on this topic are described in [12, 1]. In order to planarize a graph, some NP-hard problems have to be attacked, namely, the maximum planar subgraph problem and the (restricted) crossing minimization problem.

Since many algorithms only work for biconnected planar graphs, edges need to be added such that the resulting graphs are biconnected and planar. This problem is called the planar augmentation problem. Note, that for graph drawing it is not sufficient to simply decompose the graph into its biconnected components and draw the components separately.

Planarization

We consider the following planarization technique. In a first step, the minimum number of edges is deleted from the given graph so that the remaining graph is planar. This problem is called the maximum planar subgraph problem. In a second step, a combinatorial embedding of the planar subgraph is determined, that is, the faces are fixed. This can be done using algorithms in [3, 11].

In a third step, the removed edges are reinserted into our combinatorial embedding so that the number of crossings is minimized. This problem is a generalization of the crossing minimization problem, and hence, is NP-hard. So far, we use an iterative method that guarantees the minimum crossing number for every step. This can be achieved by solving a shortest path problem in the geometric dual graph. In the fourth step, the crossings are substituted by artificial vertices, and the resulting graph is planar. Any planar graph drawing algorithm can be used to draw the planarized graph. If the graph drawing algorithm needs as input a biconnected planar graph, some dummy edges need to be added to the graph. In the final drawing, the dummy edges and the artificial vertices are not shown.

Each of the four steps needs complicated data structures or algorithms. Due to this, at present only one implementation exists, which is not publicly available and does not use exact algorithms. Hence, work in progress aims at implementing the above planarization technique and make it publicly available through our AGD Library (see Section 7.2.4).

In earlier work [7], we suggested an integer linear programming (ILP) formulation for the weighted maximum planar subgraph problem. Moreover, we have developed and implemented a branch-and-cut algorithm using mainly the inequalities in the ILP formulation and some other inequalities derived from the polyhedral structure of the polytope associated with the set of planar subgraphs. Up to now, this is the only algorithm that is able to solve instances of the maximum planar subgraph problem of moderate size to provable optimality.

The most important inequalities appearing in the ILP-formulation are the Kuratowski inequalities, coming from the requirement that no Kuratowski subgraph, that is no subdivision of $K_{3,3}$ or K_5, is allowed to be completely contained in any feasible solution.

Since, in general, there may be exponentially many Kuratowski subgraphs in a graph, the Kuratowski inequalities need to be generated during the run of our algorithm. In every iteration, the separation problem has to be solved: Given a vector $\bar{x}$, either find a Kuratowski subgraph whose inequality is violated at $\bar{x}$, or prove that no such Kuratowski inequality exists. It is still unknown, if the separation problem for Kuratowski inequalities is NP-hard or solvable in polynomial time.

In the case that the values of $\bar{x}$ are all integral, the separation problem reduces to find a Kuratowski subgraph in a given graph. Using the planarity test of Hopcroft and Tarjan this has been achieved in [14]. We have extended the planarity test of Lempel, Even, and Cederbaum such that it returns a Kuratowski subgraph for non-planar graphs in linear time [5] (a similar method was independently discovered in [10]). We are currently implementing our new algorithm.

But, in general, the values of $\bar{x}$ are not integral. In our heuristic separation routine, we do not take the values of $\bar{x}$ into account. So, it happens quite often, that, although we find a Kuratowski subgraph, the corresponding inequality is not violated. Recently, we started to formulate the separation problem as a vertex disjoint shortest path problem between multiple sources and sinks. For the $K_{3,3}$, we have developed in [2] a heuristic separation routine based on a k-shortest disjoint path algorithm. The behaviour of our branch-and-cut algorithm improves using the new separation routines.

Planar Augmentation

Many planar graph drawing algorithms need as input a biconnected planar graph. If our graph is not biconnected, the planar augmentation problem needs to be solved. Given a planar graph, the planar augmentation problem is to add the minimum number of edges such that the resulting graph is still planar and biconnected. The problem has been introduced by Kant and Bodlaender [9], who also showed that it is NP-hard and gave a 1.5 approximation algorithm. Since, in general, a non-planar graph is given and we would like to use a drawing algorithm for biconnected planar graphs, the following problem arises: Given a graph G which is not necessarily planar and not necessarily biconnected, determine a new graph H by removing r edges and adding a edges such that the new graph H is planar, biconnected, and $r + a$ is minimum. Recently [6], we identified this new NP-hard problem and called it the minimum planarizing augmentation problem. We have given a polyhedral formulation for the new problem and the planar augmentation problem by defining two different linear objective functions over the same polytope, namely, the 2-node connected planar spanning subgraph polytope. The integer points of this polytope are identical to the integer points in the intersection of two well studied polytopes, namely, the planar subgraph polytope for the maximum planar subgraph problem and the 2-node connected spanning subgraph polytope for the network-survivability problem.

In [6, 13], we have investigated the structure of the intersection polytope. We could show that all facet-defining inequalities of the planar subgraph polytope are still facet-defining for our new polytope. Based on the theoretical results on the structure, we have designed and implemented a branch-and-cut algorithm that is able to solve many instances coming from practice to optimality within a few seconds. Very recently [4], we are re-implementing the branch-and-cut code using the system ABACUS [8]. A version for the planar augmentation problem is already finished. The results are encouraging: instances of up to 200 vertices can be solved to optimality within a few seconds. This is the first time that instances of these sizes can be solved to optimality at all.

References

- [1] F. Brandenburg, M. Jünger, and P. Mutzel. Algorithmen zum automatischen Zeichnen von Graphen. Technical Report MPI-I-97-1-007, Max-Planck-Institut für Informatik, Saarbrücken, 1997. To appear in: Informatik Spektrum (3) 1997.

- [2] R. Brockenauer. Separierung von Kuratowski-Ungleichungen für das größte planare Untergraphenproblem. Diplomarbeit, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1997.

[3] N. Chiba, T. Nishizeki, S. Abe, and T. Ozawa. A linear algorithm for embedding planar graphs using PQ-trees. *J. Comput. Syst. Sci.*, 30(1):54–76, 1985.

- [4] S. Fialko. Das Planare Augmentierungsproblem. Diplomarbeit, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1997.
- [5] C. Hundack, K. Mehlhorn, and S. Näher. A simple linear time algorithm for identifying Kuratowski subgraphs of non-planar graphs. Manuscript, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1996.
- [6] M. Jünger and P. Mutzel. The polyhedral approach to the maximum planar subgraph problem: New chances for related problems. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing (Proc. GD '94)*, volume 894 of *Lecture Notes in Computer Science*, pages 119–130. Springer-Verlag, 1995.

[7] M. Jünger and P. Mutzel. Maximum planar subgraphs and nice embeddings: Practical layout tools. *Algorithmica*, 16(1):33–59, 1996. Special Issue on Graph Drawing.

[8] M. Jünger and S. Thienel. The design of the branch and cut system ABACUS. *Technical Report No. 97.260, Institut für Informatik, Universität zu Köln*, 1997.

[9] G. Kant and H. L. Bodlaender. Planar graph augmentation problems. In *Proc. 2nd Workshop Algorithms Data Struct.*, volume 519 of *Lecture Notes in Computer Science*, pages 286–298. Springer-Verlag, 1991.

[10] A. Karabeg. Classification and detection of obstructions to planarity. *Linear and Multilinear Algebra*, 26:15–38, 1990.

[11] K. Mehlhorn and P. Mutzel. On the embedding phase of the Hopcroft and Tarjan planarity testing algorithm. *Algorithmica*, 16(2):233–242, 1996.

- [12] P. Mutzel. *Automatisiertes Zeichnen von Diagrammen*, pages 425–431. Jahrbuch 1995 der Max-Planck-Gesellschaft, Vandenhoeck & Ruprecht Verlag, Göttingen, 1995.
- [13] P. Mutzel. A polyhedral approach to planar augmentation and related problems. In P. Spirakis, editor, *Algorithms – ESA '95, Third Annual European Symposium*, volume 979 of *Lecture Notes in Computer Science*, pages 494–507. Springer-Verlag, 1995.

[14] S. G. Williamson. Depth-first search and Kuratowski subgraphs. *J. ACM*, 31(4):681–693, 1984.

7.2.3 Hierarchical Graph Drawing

Investigators: Petra Mutzel

We like to draw hierarchical graphs in a way that represents the hierarchy in the graph. If the graph is acyclic, then in a first step, the vertex set is partitioned into subsets V_i, such that all arcs are directed from V_i to V_j with $i \leq j$. So, the vertices in each subset V_i can be placed on a horizontal line such that all arcs point into one direction, say downward. In a second step of the algorithm, the vertices in each layer V_i are permuted such that the overall number of crossings, when the arcs are drawn as straight-lines, should be minimized.

We focus on the crossing minimization step. Even for only two layers, the crossing minimization problem is NP-hard. Practitioners use the following method. They fix the first layer of vertices and permute the second layer such that the number of crossings is reduced. Then the second layer is fixed, and the next layer is permuted, and so on.

The arising problem is called the 2-layer crossing minimization problem with one layer fix. Since this problem is also NP-hard, a vast amount of heuristics has been published in the literature, see, e.g., [6, 1]. In [2], we formulated the 2-layer crossing minimization problem with one layer fix as a linear ordering problem, which can be solved very efficiently in practice via branch-and-cut. So, for the first time, we have been able to solve instances of sizes up to 250 vertices on the free layer to optimality. After implementing many heuristics from the literature, we ran extensive computational experiments. For instances with up to 60 vertices, our algorithm needs less than one second and, hence, with respect to running time, is compatible with all of the heuristics. However, it quite often gives much better results.

Our ultimate goal was to solve the k-layer crossing minimization problem. For two layers, we designed and implemented a branch-and-bound algorithm based on our branch-and-cut algorithm. Unfortunately, this algorithm is able to solve only small instances of the 2-layer crossing minimization problem. A surprising result coming out of our experiments was that for 2-layers the practitioners' approach of iteratively solving the one-fixed problem leads to a number of crossings that is much higher than the optimal 2-layer crossing number.

Hence, alternative methods have to be designed. One idea is to investigate a polyhedral approach for the k-layer crossing minimization problem. We are investigating in this direction. But a lot of effort will be needed to get an efficient algorithm that will be able to solve the k-layer crossing minimization problem for moderately sized instances such as they occur in graph drawing.

We suggest to solve an alternative problem that may be easier to attack. Instead of solving the k-layer crossing minimization problem, we suggest solving the k-layer planarization problem, namely, removing the minimal number of edges such that the resulting graph is k-layer planar. In the final diagram the removed edges are reinserted straight-line into the k-layer planar drawing.

In contrast to the 2-layer crossing minimization problem, the 2-layer planarization problem (NP-hard) can be attacked via branch-and-cut quite easily. In [4], we investigated the associated polytope and found some big classes of facet-defining inequalities. For most of them, the separation problems can be solved in polynomial time, hence they can be used in a branch-and-cut algorithm [7]. Our algorithm is implemented using ABACUS [3]. We plan to investigate the problem for more than two layers in the near future.

So far, for graphs with more than two layers, we use the practitioners' approach and iteratively solve the 2-layer planarization problem when one layer is fixed. In [5], we investigated the polytope associated with this problem. It is related to the linear ordering polytope and the 2-layer planar subgraph polytope. The facet-defining inequalities of both polytopes are still valid for the new polytope. Surprising is the fact that all facet-defining inequalities of the linear ordering polytope are still facet-defining for the new polytope. The facial structure can be used in our branch-and-cut algorithm [7] that is implemented using ABACUS [3]. Our computational experiments show that the hierarchical drawings based on our iterative k-layer planarization method are of similar quality as those obtained by the iterative k-layer crossing minimization method. Our hope is that the k-layer planarization problem will be easier to solve in practice than the k-layer crossing minimization problem.

Surprisingly, the 2-layer planarization problem for two fixed layers is equivalent to the maximum weight trace problem for 2 sequences that we investigate in the context of computational biology (see Section 8.1).

References

[1] P. Eades and D. Kelly. Heuristics for reducing crossings in 2-layered networks. *Ars Combinatoria*, 21-A:89–98, 1986.

[2] M. Jünger and P. Mutzel. Exact and heuristic algorithms for 2-layer straightline crossing minimization. In F. J. Brandenburg, editor, *Graph Drawing (Proc. GD '95)*, volume 1027 of *Lecture Notes in Com-*

puter Science. Springer-Verlag, 1996. Revised version to appear in Journal of Graph Algorithms and Applications (JGAA).

[3] M. Jünger and S. Thienel. The design of the branch and cut system ABACUS. *Technical Report No. 97.260, Institut für Informatik, Universität zu Köln*, 1997.

• [4] P. Mutzel. An alternative approach for drawing hierarchical graphs. In S. North, editor, *Graph Drawing (Proc. GD '96)*, Lecture Notes in Computer Science. Springer-Verlag, 1997. To appear.

• [5] P. Mutzel and R. Weiskircher. Two-layer planarization in graph drawing. Extended Abstract, Max-Planck-Institut für Informatik, Saarbrücken, Germany, submitted, 1997.

[6] K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical systems. *IEEE Trans. Syst. Man Cybern.*, SMC-11(2):109–125, 1981.

• [7] R. Weiskircher. 2-Schicht Planarisierung von bipartiten Graphen. Diplomarbeit, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1997.

7.2.4 Practical Projects and Software

Investigators: Christoph Hundack, Heikki Mannila, Petra Mutzel, and Thomas Ziegler

Practical Projects

We have produced software for various practical projects.

In archaeology the dating of objects is not only obtained by absolute dating evidence (like radiocarbon dating) but also by relative dating in archaeological contexts (e.g., in excavations). The latter is known as stratigraphy. The goal is to display the graph representing the chronological relationships between archaeological items; this graph is usually very large. Previous existing software can only output crossing free drawings [3]. (Edges that produce crossings are simply removed.) In order to display all relationships, we implemented a layer-based drawing algorithm using the transitive reduction of the graph. The problem of further reducing the number of edges and crossings has been handled by a heuristic for the edge-concentration problem (see also [6]). We are in contact with archaeologists, who frequently test our software.

In data mining applications, certain rules found by the data mining software need to be visualized. In our current project, a simple approach would lead to a highly dense graph, which cannot be visualized nicely. We came up with ideas of sparsifying the graph by introducing additional vertices. The drawings look very pleasant.

We are getting industry funds for cooperating on a project involving drawings of finite state machines that describe the control of computer integrated manufacturing processes. Here, we have the opportunity to show that the drawing method using planarization is indeed practical. Recently, related to this project, we got a grant of the German Ministry of Education, Science, and Technology. Within this project we will investigate the labelling problem in connection with graph drawing. (See Section 15 for details on these projects.) Very recently, we have been contacted by a software company for a new cooperation on graph drawing.

AGD Library

We will provide implementations of our graph drawing algorithms and tools via the AGD (Algorithms for Graph Drawing) Library [1, 4] to interested users. Tools include data structures and algorithms that can be used as modules in graph drawing algorithms. The algorithms access common graphical attributes of the graph via a generic layout interface in order to use the algorithms with various applications. The algorithms in the library are represented by classes. We introduce a hierarchy of abstract base classes to group algorithms, and define a generic interface to the common services offered by an algorithm.

We provide the library with layout interfaces for the graph editors Graphlet [2] and GraphWin (LEDA) [5], and with graphical user interfaces so that the algorithms can be directly used with both graph editors. The AGD Library is a product of our cooperation within the “DFG-Schwerpunkt” program (see Section 15) with partners in Halle, Köln, and Passau.

References

• [1] D. Alberts, C. Gutwenger, P. Mutzel, and S. Näher. The design of the AGD Library. Manuscript, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1997.

[2] F. Brandenburg and M. Himsolt. *Graphlet.* Available via the URL `http://www.fmi.uni-passau.de/~himsolt/Graphlet`.

[3] I. Herzog. *Computer-Aided Harris Matrix Generation*, pages 201–217. E.C. Harris (Ed.), Academic Press Ltd., London, 1993.

• [4] Max-Planck-Institut Saarbrücken, Universität Halle, Universität Köln, Universität Passau. *The AGD Library User Manual*, 1997. Available via the URL `http://www.mpi-sb.mpg.de/~mutzel/dfgdraw/dfgdraw1.html`.

[5] K. Mehlhorn, S. Näher, and C. Uhrig. *The LEDA User Manual, Version R 3.5*, 1997.

[6] F.J. Newbery. Edge concentration: A method for clustering directed graphs. In *International Workshop on Software Configuration Management*, pages 76–85, 1990.

8 Computational Biology, Chemistry and Physics

Our work in this area spans a wide range, from multiple sequence alignment to protein docking and from molecular dynamics simulations for synthetic polymers to the computation of ground states for Ising spin glasses.

In 1995 a successful cooperation within our group has started between researchers in computational molecular biology and researchers in combinatorial optimization. The two strongest results that came out of this cooperation are new integer linear programming formulations and branch-and-cut algorithms for two fundamental problems in computational molecular biology (Section 8.1).

The first problem is the Multiple Sequence Alignment Problem. Sequence alignments are of central importance in the interpretation of protein and DNA sequences. The comparison of a set of protein or DNA sequences using multiple alignment techniques may yield information about the patterns in the sequences that are functionally important. We have presented a linear programming formulation for a special instance of the Multiple Sequence Alignment problem, the so-called Maximum Weight Trace problem. The intensive study of the solution polyhydron leads us to a branch-and-cut algorithm that enables us to solve to optimality problem instances that cannot be handled by dynamic programming approaches.

The second problem is the Physical Mapping Problem. Given a set of fragments of a DNA sequence and overlap information (overlap between fragments), a physical mapping algorithm tries to reconstruct the complete DNA sequence. The Physical Mapping Problem is a central problem in the area of genome research. We have studied a special instance of the Physical Mapping Problem, the so-called STS-Content Physical Mapping Problem. The new branch-and-cut algorithm enables the user for the first time to solve small mapping instances to optimality even in the presence of a high level of error in the input. Comparison with the widely-used transformation to a Hamming-Distance Traveling Salesman Problem suggests that the new betweenness formulation may be the better approach for solving the physical mapping problem.

In the area of protein structure research, we have worked on the Protein Docking Problem. The goal of protein docking research is the development and implementation of algorithms that enable the user to predict reactions between proteins and to compute the three-dimensional structure of the resulting protein complexes. Protein docking algorithms can be used, for example, to screen protein databases for possible inhibitors of a given virus enzyme. The main result of our protein docking research was the development and implementation of new contact measures for judging possible docking conformations. We have tested the contact measures with "real world" docking examples. The quality of the docking results is very good. To be more precise, there is only one other docking algorithm that yields results of similar quality; that algorithm, however, needs more running time. A few exciting new features were added to our docking suite that enable the user to screen databases for suitable reaction partners for a given protein.

In the research area "Computational Chemistry" (Section 8.2), we have worked on efficient parallel algorithms for molecular dynamics (MD) simulations for synthetic polymers, which are the base of all varieties of plastic for example, PVC, nylon, and polyester. In an MD simulation the motions of the atoms of a molecular system are simulated using classical mechanics. MD simulations have become an important tool for testing hypotheses about chemical processes. Since the required amount of computing power is tremendous, there is a strong interest in parallel algorithms. We have developed and implemented an efficient parallel MD simulation algorithm for synthetic polymers that makes use of their special properties and behaviour. Our algorithm runs on distributed memory architectures and has turned out to be very efficient for up to 16 processors.

Finally, in the area of "Computational Physics" (Section 8.3), we continued our research on the computation of exact ground states for the Ising spin glass model, which is one of the main models used in the modeling of real dynamic systems.

8.1 Computational Molecular Biology

Multiple Sequence Alignment

Investigators: Hans-Peter Lenhof, Kurt Mehlhorn, Petra Mutzel, and Knut Reinert

The *Multiple Sequence Alignment Problem* is one of the fundamental problems in computational biology. It can be formulated as follows. Let Σ be a finite alphabet and let $\hat{\Sigma} = \Sigma \cup \{-\}$, where "$-$" (dash) is a symbol to represent "gaps" in strings. The input of a multiple sequence alignment algorithm is a set $S = \{S_1, S_2, \ldots, S_k\}$ of finite strings over the alphabet Σ with lengths $n_1, \ldots, n_k$. A set $\hat{S} = \{\hat{S}_1, \hat{S}_2, \ldots, \hat{S}_k\}$ of strings over the alphabet $\hat{\Sigma}$ is called an *alignment* of S if the following two properties hold: (1) all strings in $\hat{S}$ have the same length, and (2) ignoring dashes, the string $\hat{S}_i$ is identical with the string S_i. An alignment can be interpreted as an array with k rows, one row for each $\hat{S}_i$. Two letters of distinct strings are called aligned under $\hat{S}$ if they are located in the same column. The Multiple-Sequence Alignment problem asks to find a "good" alignment $\hat{S}$ of S. There are many ways to measure the quality of an alignment. We study a specific multiple sequence alignment formulation, the so-called *Maximum Weight Trace Problem (MWT).* It was introduced in 1991 by Kececioglu [2], originally as a model for the final alignment phase of DNA sequence assembly.

In the MWT the letters of the strings $S_i = (s_{i1}, \ldots, s_{in_i})$ of S are viewed as vertices V in a subgraph of a complete k-partite graph $G = (V, E)$. Every edge $e \in E$ has a non-negative weight representing the gain of aligning the endpoints of the edge. We say that an alignment realizes an edge if it places the endpoints into the same column of the alignment array. The set of edges realized by an alignment $\hat{S}$ is called the *trace* of $\hat{S}$, denoted $\mathrm{trace}(\hat{S})$, and the weight of an alignment is the sum of the weights of the edges that it realizes. The goal is to compute an alignment $\hat{S}$ of maximum weight.

Like most multiple alignment problems, MWT can be solved by dynamic programming, which yields an algorithm with time complexity $O(k^2 2^k N)$ and space complexity $O(N)$, where $N = \prod_i n_i$. This is only feasible for very small problem instances. While MWT is NP-complete, Kececioglu [3] presented a branch-and-bound algorithm whose implementation can optimally align six sequences of length 250 in a few minutes. Larger examples, however, require excessive space.

In [6] we presented a new approach. We formulated the MWT as an integer linear program and applied branch-and-cut techniques. A trace can be represented by an $|E|$-dimensional $\{0, 1\}$ vector, where a 1 indicates that the corresponding edge is realized. Define the *trace polytope* $P_T(G)$ as the convex hull of all incidence vectors of traces in G. In order to use linear programming techniques we have to get a description of $P_T(G)$, at least a partial one, in terms of linear inequalities (also called constraints). The most effective constraints are the ones that define facets of the trace polytope $P_T(G)$, since these are the constraints that appear in an irredundant description of $P_T(G)$ by linear inequalities.

In [6] we gave a complete description of the trace polytope for the two-sequence case, i.e., we gave the complete set of the linear inequalities. We also gave a polynomial time separation algorithm for them (a separation algorithm for a class of inequalities takes a point in $\mathbb{R}^{|E|}$ and returns a violated inequality from the class if there is one). Our complete description implies the existence of a polynomial-time, non-dynamic-programming algorithm for pairwise sequence alignment and thus gives another answer to a longstanding problem in stringology [5]. Furthermore, we gave a partial description of the trace polytope for the multiple-sequence case. More specifically, we showed that all facet-defining inequalities of the two-sequence case lift to the multiple-sequence case and derived three more classes of facet-defining inequalities.

We implemented the algorithm using the branch-and-cut framework ABACUS [1] and the Library of Efficient Data types and Algorithms LEDA [4]. We were able to align 15 sequences of prion proteins to optimality. To our knowledge, there is no dynamic programming algorithm that can align these many sequences optimally, even if the sequences are very similar.

References

[1] M. Jünger and S. Thienel. The design of the branch and cut system ABACUS. *Technical Report No. 97.260, Institut für Informatik, Universität zu Köln*, 1997.

[2] J. D. Kececioglu. *Exact and approximation algorithms for DNA sequence reconstruction.* PhD thesis, U. Arizona, 1991.

[3] J.D. Kececioglu. The maximum weight trace problem in multiple sequence alignment. In *Proc. 4-th Symp. Combinatorial Pattern Matching*, number 684 in Lecture Notes in Computer Science, pages 106–119, 1993.

[4] K. Mehlhorn and S. Näher. LEDA, a platform for combinatorial and geometric computing. *Communications of the ACM*, 38(1):96–102, 1995.

[5] P.A. Pevzner and M.S. Waterman. Generalized sequence alignment and duality. *Advances in Applied Mathematics*, 14:139–171, 1993.

• [6] K. Reinert, H.P. Lenhof, P. Mutzel, K.Mehlhorn, and J. Kececioglu. A branch-and-cut algorithm for multiple sequence alignment. In *Proceedings of the First Annual International Conference on Computational Molecular Biology (RECOMB'97)*, pages 241–250, 1997.

Sequencing and Mapping

Investigator: Dany Breslauer and Petra Mutzel

The goal of the Human Genome Project is the sequencing (the determination of the sequence of nucleotides A, C, G, and T) of the entire human genome and the determination of the location of the human genes on the DNA. In total, the human genome contains about 3 billion letters distributed over 23 chromosomes. Biochemical techniques, however, allow the researcher only to handle comparatively small segments of DNA at a time. Thus, a DNA molecule is broken into smaller pieces. The information one obtains on small pieces needs to be put into a larger context again. To this end one has to infer the order in which the segments occur on the underlying DNA. This task is called mapping. Depending on the scale at which it is done, the techniques and the corresponding computational problems vary. For an overview, see e.g., [8].

Sequence assembly

Current technology permits experimentors to directly determine the sequence of a DNA strand of approximately 500 nucleotides. To sequence a long piece of DNA many such smaller segments are sequenced and subsequently re-assembled to produce the original sequence. Computationally, this gives rise to the problem of assembling the fragments using the overlap information. Overlaps are deduced from sequence similarity. Since 1%–10% of the nucleotides in the fragment data are missing or incorrect, and since a fragment's orientation is not known, these overlaps cannot be perfectly determined. The formal *Sequence Reconstruction Problem* can be stated as follows: Given a collection of fragment sequences $\mathcal{F}$ and an error rate $0 \leq \epsilon < 1$, find a shortest sequence S such that every fragment $F \in \mathcal{F}$, or its reverse complement, matches a substring of S with at most $\epsilon|F|$ errors.

A strongly idealized formalization of sequence assembly assumes that fragment orientation is known and no errors occur. The resulting problem is the *Shortest Common Superstring Problem* for which NP-hardness and MAX-SNP hardness have been shown.

In [3], the first linear approximation factor with respect to length n is given. A variation of the greedy algorithm is suggested that leads to an approximation factor of 3. A series of papers which subsequently improved the approximation factor for the shortest common superstring problem followed.

We came up with two approximation algorithms for the shortest common superstring with approximation factors of 2.75 and 2.596 [6]. Our algorithms are simpler than the previous approximation algorithms that lead to a factor less than three. The algorithms are similar in spirit to the previous algorithms for the problem but use a novel combinatorial lemma on strings. We also construct a superstring by computing some optimal cycle covers on the distance graph of the given strings, and then break and merge the cycles to finally obtain a Hamiltonian path. The key to the improvement are new bounds on the overlap between two strings. Independently of our work and about at the same time, Armen and Stein [2] discovered a 2.75 approximation algorithm and Sweedyk [9] gave a 2.5 approximation algorithm.

Mapping using hybridization data

For large pieces of DNA, mapping techniques based on the experimental determination of overlaps between fragments (also called *clones*) are used. Several experimental methods to obtain the required overlap information are in use. One approach detects certain very specific sequence segments in the fragments. This is done using probes hybridizing to specific bits of the sequence. The hope is that two clones share a probe if and only if they overlap. If a probe is absolutely unique it is called a *Sequence Tagged Site* (*STS*). Ideally, the matrix describing which clones contain such a probe will have the consecutive ones property. Hence, the correct orderings can be found very easily using the PQ-tree data structure to generate the set of all arrangements of probes consistent with the data. Alternatively, there is another experiment that allows the determination of overlap between two clones directly.

However, both techniques are error prone. In practice, false positive overlaps will be reported and overlaps may not be recognized, i.e., there are false negatives. A further complication is that sometimes two clones are merged into one. This phenomenon is called chimerism.

The problem is to reconstruct a DNA fragment in the presence of errors. Alizadeh, Karp, Weisser and Zweig [1] first considered a maximum-likelihood model of the problem equivalent to finding an ordering of the probes that minimizes a weighted sum of errors and developed several effective heuristics including the Hamming-Distance Travelling Salesman Problem.

In [4] we show that by exploiting information about the end-probes of clones, the maximum-likelihood model can be formulated as a weighted *Betweenness Problem.* Given a collection of betweenness and nonbetweenness constraints on a set of elements to be linearly ordered, the Weighted Betweenness Problem asks for an ordering π of the elements that minimizes a weighted sum of the constraints violated by π. We give an integer linear programming formulation of Weighted Betweenness based on the Linear Ordering Problem.

However, when the integer linear program is relaxed by dropping integrality constraints, we need additional inequalities to get good solutions with a linear programming method. In order to get these additional inequalities, we use a new technique introduced in [5]. Instead of theoretically investigating the polytope associated with an instance of the betweenness problem, we use the structure of polytopes associated with small instances. The inequalities describing small polytopes can be computed by enumerating all feasible solutions and applying the double-description method. These inequalities are then projected to the bigger instance in which we are interested in. Our implementation of the branch-and-cut algorithm uses the framework ABACUS [7].

In our experiments on generated data we could verify the maximum-likelihood model given in [1] for instances with false positive and false negative errors. Our computational results suggest that introducing end-probe information to the experimental protocol could significantly improve the quality of the physical maps that are constructed, while tolerating a much higher experimental error rate. For the first time we present exact solutions for small mapping instances even in the presence of a high level of error. We also show that by combining the optimal solution of many small overlapping betweenness problems, one can effectively screen errors from larger instances and solve the edited instance to optimality as a Hamming-Distance Traveling Salesman Problem. This suggests a new combined approach to physical map construction.

References

[1] F. Alizadeh, R.M. Karp, D.K. Weisser, and G. Zweig. Physical mapping of chromosomes using unique probes. *J. Comput. Biol.*, 2(2):159–184, 1995.

[2] C. Armen and C. Stein. Shortest superstrings and the structure of overlapping strings. *J. Comput. Biol.*, 2(2):307–332, 1995.

[3] A. Blum, T. Jiang, M. Li, J. Tromp, and M. Yannakakis. Linear approximation of shortest superstrings. *J. ACM*, 41(4):630–647, 1994.

• [4] T. Christof, M. Jünger, J. Kececioglu, P. Mutzel, and G. Reinelt. A branch-and-cut approach to physical mapping with end-probes. In *Proc. of the First Intern. Conf. on Computational Molecular Biology (RECOMB'97)*, pages 84–92, 1997.

[5] T. Christof and G. Reinelt. Combinatorial optimization and small polytopes. *TOP, Spanish Statistical and Operations Research Society*, 4(1), 1996.

• [6] T. Jiang, Z. Jiang, and D. Breslauer. Rotation of periodic strings and short superstrings. Technical report, Max-Planck-Institut f. Informatik, Saarbrücken, Germany, 1996. To appear in: J. Algorithms.

[7] M. Jünger and S. Thienel. The design of the branch and cut system ABACUS. *Technical Report No. 97.260, Institut für Informatik, Universität zu Köln*, 1997.

[8] R.M. Karp. Mapping of the genome: Some combinatorial problems arising in molecular biology. In *Proc. 25-th Annual ACM Symp. Theory of Comput.*, pages 278–285, 1993.

[9] E.S. Sweedyk. *A 2 1/2 appoximation algorithm for shortest common superstring.* PhD thesis, University of California, Berkeley, Department of Computer Science, 1995.

Protein Docking

Investigator: Hans-Peter Lenhof

Proteins are a major component of cells, accounting for nearly half their dry mass. They play key roles in almost all intercellular and intracellular biochemical reactions. Proteins are chain molecules composed of twenty amino acids. The linear amino acid chain of a protein folds in a very specific way and confers upon the protein a specific three-dimensional structure crucial to its chemical activity.

The goal of protein docking research is the development and implementation of algorithms that enable the user to predict reactions between proteins and the stability of the resulting protein complexes. The protein docking problem can be formulated as follows: Given two proteins A and B and their three-dimensional structures, compute the three-dimensional structures of possible protein docking complexes AB. A protein docking algorithm outputs a list of possible docking complexes AB sorted with respect to a certain fitness or energy function that measures the stability of the complexes. The examination and determination of protein complex structures often takes years. Protein docking algorithms accelerate the structure determination process by proposing possible conformations that may help to form hypotheses about the complex structure. A generalization of the protein docking problem plays an important role in the research area of drug design: Given a protein A, search in structure databases for molecules that build a stable complex with protein A. Algorithms that solve the generalized protein docking problem can be used, for example, to screen structure databases for possible inhibitors of a given virus enzyme.

Although the mechanisms of docking reactions are not well known, two complementarity principles seem to be important for the recognition of docking partners. The first principle is the *shape complementarity principle*: the shapes of the molecules that build a docking complex are locally geometrically complementary, that is, there is a good fit between the surfaces of the docking partners. The second principle is the *chemistry principle*. It states that there is a strong chemical complementarity between the sites of docking partners.

Most protein docking algorithms use the following approach (see for example [9, 7, 5, 8, 2]): First, a set of possible docking conformations that hopefully contains the right conformation(s) is computed. Second, the set of conformations is tested with respect to their geometric fitness: Select the conformations of the test set that have a good fit between the surface of A and the surface of B and no overlap between the interior of A and the interior of B. The remaining conformations are tested with different chemical filters. At the end the algorithm outputs a short list of possible conformations that are sorted with respect to a special fitness function or with respect to energy values (see for example [2]).

Connolly [1] proposed the first geometry-based approach for the protein docking problem. The algorithm of Connolly computes the test set of conformations by matching quadruples of "critical" points on the contact surface of molecule A against quadruples on the contact surface of B. Wang's algorithm [9] matches only pairs of critical points on the surfaces. Norel et al. [7, 8], Lin et al. [5] and Fischer et al. [2] use pairs of critical points and the corresponding surface normals to compute surface point matches.

In 1995 we presented a docking algorithm [3] that uses triples of points to match the surfaces of A and B. Furthermore, we introduced a new geometric scoring function that counts the number of van der Waals contacts between atoms of A and atoms of B. We have implemented a parallel version of our algorithm and tested it with 52 "real world" docking examples. In 42 tests the best conformation with respect to our scoring function was very close of the real conformation (the root mean square deviation was smaller than 3 Å). Our test set included the examples of Norel et al. [7]. The ranking of our results was significantly better than the rankings of the previous best algorithms, due to Norel et al. [7, 8], and Fischer et al. [2]. The sequential running time of our algorithm is on the order of the running time of the algorithm of Norel et al. The parallel version has a good speed-up and modest communication requirements.

In 1996 Meyer et al. [6] published a new docking algorithm that uses correlation techniques and makes heavy use of possible hydrogen bonds. The ranking of their docking results is of the same quality as the ranking of our results, but the running times of their algorithm are much larger than the running times of our algorithm.

In [4] we presented a new chemical scoring function for judging the chemical fitness of conformations. The new fitness measure is also based on the van der Waals contact principle: We consider all atom pairs that have a van der Waals contact, but instead of adding a constant for each atom pair (a, b) we add a chemical weight that depends on the atom pair. The weights are calculated by making a statistical analysis of van der Waals contacts in protein complexes. Our algorithm uses the new chemical scoring function as a second filter after the geometric fitness function. The conformations in the resulting list of the algorithm are sorted with respect to the sum of two fitness functions. The new algorithm successfully identified the real docking conformations in more than 90% of the test examples, that is, the best conformation with respect to the sum of the fitness functions was an approximation of the real conformation. The parameter set was optimized so that good results are attained for examples where molecule B is of small or medium size. The rationale behind choosing the parameter set to perform well on such examples is the following: We plan to use the docking program for database screening; i.e., searching databases for possible small or medium sized docking partners for a given protein. Further a graphical user interface for marking parts of the surface of A was implemented. If a part of the surface of A is marked, then the algorithm carries out a local docking search testing only the marked area; i.e., the user can, for example, mark the active site of an enzyme of a virus and search for possible inhibitors in databases.

The pharmaceutical company Boehringer Mannheim and the Max-Planck-Institut für molekulare Physiologie (Dortmund) have shown strong interest in the docking package. In the near future we will release a first version for testing purposes.

References

[1] M.L. Connolly. Shape complementarity at the Hemoglobin $\alpha_1\beta_1$ subunit interface. *Biopolymers*, 25:1229–1247, 1986.

[2] D. Fischer, S.L. Lin, H.L. Wolfson, and R. Nussinov. A geometry-based suite of molecular docking processes. *J. Mol. Biol.*, 248:459–477, 1995.

• [3] H.P. Lenhof. An algorithm for the protein docking problem. In Dietmar Schomburg and Uta Lessel, editors, *Bioinformatics: From nucleic acids and proteins to cell metabolism: GBF Monographs Volume 18*, pages 125–139, 1995.

• [4] H.P. Lenhof. New contact measures for the protein docking problem. In *Proc. of the First Annual International Conference on Computational Molecular Biology (RECOMB'97)*, pages 182–191, 1997.

[5] S.L. Lin, R. Nussinov, D. Fischer, and H.J. Wolfson. Molecular surface representations by sparse critical points. *PROTEINS: Structure, Function and Genetics*, 18:94–101, 1994.

[6] M. Meyer, P. Wilson, and D. Schomburg. Hydrogen bonding and molecular surface shape complementarity as a basis for protein docking. *J. Mol. Biol.*, 264(1):199–210, 1996.

[7] R. Norel, S.L. Lin, H.J. Wolfson, and R. Nussinov. Shape complementarity at protein-protein interfaces. *Biopolymers*, 34:933–940, 1994.

[8] R. Norel, S.L. Lin, H.J. Wolfson, and R. Nussinov. Molecular surface complementarity at protein-protein interfaces: The critical role played by surface normals at well placed, sparse points in docking. *J. Mol. Biol.*, 252:263–273, 1995.

[9] H. Wang. A grid-search molecular accessible algorithm for solving the protein docking problem. *J. Comp. Chem.*, 12:746–750, 1991.

8.2 Computational Chemistry

Molecular Dynamics Simulation for Synthetic Polymers

Investigators: Hans-Peter Lenhof, Peter Müller, and Christine Rüb

Molecular dynamics (MD) simulations have become an important tool for testing hypotheses about chemical and physical processes. In an MD simulation, the motions of the atoms of a molecular system (i.e., molecule(s) and an environment, typically a solvent) are simulated using classical mechanics. Given the atomic positions and velocities at time t, interaction forces are calculated according to a physical model (the force field). Then, by numerical integration of Newton's equations of motion, the positions and velocities of the atoms at time $t + \Delta$ can be computed. The time step Δ must be chosen carefully and is typically on the order of 1 fs ($= 10^{-15}$s). Unfortunately, in order to study the dynamics of macromolecules, simulation periods of up to milliseconds or even seconds are desirable. The tiny time steps mean that the above-mentioned force field evaluation and integration — which constitute one iteration in an MD simulation — must be executed very often. Therefore, these simulations impose a heavy drain on computing resources.

One promising way to accelerate MD simulations is using parallel computers. Over the last ten years many parallel algorithms have been developed and implemented (e.g., [5, 2]; a comprehensive overview can be found in [1]). Common to almost all of these is the fact that they are designed for simulating biopolymers, especially proteins. Since proteins fold up to a very compact structure, all these algorithms use a spatial decomposition method where the protein is placed into a small box which is subdivided into cells. These cells are distributed among the available processors. Each processor is responsible for the atoms in its cells. Due to the compact structure and the fact that proteins do not move much during a typical simulation, the spatial decomposition method allows good load balancing and little communication overhead.

However, using such an algorithm for the simulation of another class of macromolecules — synthetic polymers — has turned out to be quite inefficient. The main reason is a completely different behaviour. Synthetic polymers do not build compact structures but form long loose coils and their typical trajectories are three dimensional random walks with significant movement. Furthermore, the dynamics of such a polymer can be studied by simulating only one single polymer chain; surrounding atoms of a solvent or other polymer molecules need not be included explicitly (for more details see [4]). Hence, in our view spatial decomposition methods are not the ideal way of doing MD simulations for synthetic polymers.

We have developed and implemented parallel algorithms that take the special properties and behaviour of synthetic polymers into consideration. These algorithms are platform independent and run on any parallel architecture with a moderate number of processors, distributed memory and message passing as means of communication.

Up to now, we have developed and implemented an algorithm that uses a straightforward molecule decomposition (i.e., the polymer chain is divided into segments of roughly the same size which are allocated to the processors). Most of the data exchange that is necessary between neighbouring processors (with respect to the subdivision of the chain) can be hidden by non–blocking send/receive operations. Another necessity for communication arises because of loops and bends of the polymer chain. This leads to fast changing and irregular communication patterns between some of the processors. In order to deal with this, we use a master method where one processor collects sufficient information from all other processors and detects the necessary data exchange. The master then informs the other processors which data they have to exchange with which processor. This algorithm has turned out to be very efficient for up to 16 processors (for more details, see [4]).

An often neglected aspect of developing MD simulation algorithms is the verification of simulation results. We attached great importance to this in order to see if the computed output matches known theory, thus proving the correctness and usefulness of the implementation. In [3] we showed that the calculated mean end–to–end distances and diffusion coefficients for a polyethylene chain in good solvent correspond to the expected values. Currently, we use our algorithm to simulate the collapse of polymer chains in poor solvents, which has not been done so far in a satisfactory way.

Since January 1995, this project has been supported by the DFG Research Cluster "Efficient Algorithms For Discrete Problems and Their Applications", grant LE 952/1–2 (see Section 15).

References

[1] D. Beazley, P. Lomdahl, N. Grønbech-Jensen, R. Giles, and P. Tamayo. Parallel algorithms for short–range molecular dynamics. *World Scientific's Annual Reviews in Computational Physics*, 3, 1995.

[2] H. Heller, H. Grubmüller, and K. Schulten. Molecular dynamics simulation on a parallel computer. *Molecular Simulation*, 5:133–165, 1990.

• [3] B. Jung, H.P. Lenhof, P. Müller, and C. Rüb. Langevin dynamics simulations of macromolecules on parallel computers. *Macromolecular Theory and Simulation*, 6:507–521, 1997.

• [4] B. Jung, H.P. Lenhof, P. Müller, and C. Rüb. Parallel MD-simulations of synthetic polymers. In *Proc. of the 8th SIAM Conference on Parallel Processing for Scientific Computing*, 1997. Full version as Tech. Rep. MPI-I-97-1-003, 1997.

[5] M. Nelson, W. Humprey, A. Gursoy, A. Dalke, L. Kale, R. Skeel, K. Schulten, and R. Kufrin. MDscope: A visual computing environment for structural biology. *Comp. Phys. Comm.*, 1995.

8.3 Computational Physics

Computation of Exact Ground States for Ising Spin Glasses

Investigator: Petra Mutzel

Studying the properties of dynamic systems is an important problem in physics. A model called the Ising spin glass model is one of the most commonly used, for both its simplicity and its accuracy in representing real dynamic systems. When cooling down an Ising spin glass system to zero temperature, a state of minimum energy is reached, the so-called ground state. For Ising spin glasses, the description of the phase transition and the nature of the ground state are still controversial. The starting point for most theoretical work is the Edwards-Anderson model, where the spins σ_i lie on a regular lattice, and there are random interactions between the spins σ_i and σ_j. We consider the important case in which only nearest neighbour interactions are considered.

Two-dimensional spin glass problems can be modelled as optimization problems ("Find the state of minimum energy") in a 2-dimensional grid graph. In order to approximate the infinite size of spin glasses by a finite model on $L \times L$ spins, periodic boundary conditions are introduced via glueing together the opposite sides of the planar grid graph. This leads to a toroidal graph. Although the ground state can be computed in polynomial time [1], the algorithm is not practical, since the running time is $O(|V|^7)$ for a system of size $|V| = L \times L$. Often an external magnetic field is introduced, which can be modelled as a supernode adjacent to all the spin vertices. In this case the ground state problem is NP-hard [1]. Due to the difficulty of the problem, the most popular methods described in the literature are Monte Carlo simulation, simulated annealing, and evolutionary and genetic algorithms. Only few authors have reported on computational results for methods to compute the exact ground states [8, 2].

The ground state problem can be transformed to the problem of determining a cut of maximum weight in a toroidal grid graph with a supernode. In [2], it is shown that cutting plane techniques are able to compute the exact ground states for spin glass instances up to size 35×35. The facial structure of the polyhedra associated with all cuts is well-studied in the case of complete graphs. However, only a few papers investigate the polyhedral structure for sparse graphs [3]. In particular, we investigated the polyhedral structure for toroidal grid graphs [7]. These investigations and a redesign of the branch-and-cut code of [2] lead to big improvements of our computational results.

In the last progress report we presented results for the Gaussian model, where the interactions between the spins are taken from a Gaussian distribution. In [5] we have been able to solve 100×100 grids in moderate computation times. Indeed, we could solve single instances of much bigger size, say 150×150 (that means 22500 nodes).

Continuing along the same line of research, we have studied recently interactions that are not Gaussian, but have fixed magnitude $|J|$, and only the sign is random, thus getting a so-called $\pm J$ Ising spin glass. Computationally, these problems are much harder than the Gaussian model. The biggest $\pm J$ spin glass problem that was solved until recently has size 50×50. The authors reported that it needed about 17 hours of CPU-time on a MIPS M-120 workstation [4]. Using numerical techniques and methods from software engineering in our new branch-and-cut code, we can solve $\pm J$ spin glasses of size up to 70×70 [6]. For the 50×50 instance mentioned above our branch-and-cut code needs 11 minutes on a SUN SPARC-station 10.

Besides our extensive computational tests that give a new estimation of the ground state energy of infinite spin glass systems, we also made our code available on the web in

`http://www.informatik.uni-koeln.de/ls_juenger/projects/spinglass.html`.

Recently, we were asked by well known physicists for running additional computational experiments in order to confirm their conjectures.

References

[1] F. Barahona. On the computational complexity of Ising spin glass models. *J. Phys. A: Math. Gen.*, 36:3241 – 3253, 1982.

[2] F. Barahona, M. Grötschel, M. Jünger, and G. Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Oper. Research*, 36:493 – 513, 1988.

[3] F. Barahona and A.R. Mahjoub. On the cut polytope. *Math. Progr.*, 36:157–173, (1986).

[4] F. Barahona and H. Titan. Max mean cuts and max cuts. Technical report, DIMACS, Technical Report 18, 1991.

[5] C. De Simone, M. Diehl, M. Jünger, P. Mutzel, G. Reinelt, and G. Rinaldi. Exact ground states of Ising spin glasses: New experimental results with a branch and cut algorithm. *Journal of Statistical Physics*, 80:487–496, 1995.

• [6] C. De Simone, M. Diehl, M. Jünger, P. Mutzel, G. Reinelt, and G. Rinaldi. Exact ground states of two-dimensional $\pm J$ Ising spin glasses. *Journal of Statistical Physics*, 84(5/6):1363–1371, 1996.

[7] C. De Simone, M. Jünger, P. Mutzel, and G. Rinaldi. An approach to a combinatorial algorithm for max cut on toroidal grid graphs. Extended Abstract, 1994.

[8] L. Saul and M. Kardar. Exact integer algorithm for the two-dimensional $\pm J$ Ising spin glass. *Phys. Rev. E*, 48, 1993.

9 Software Libraries and Supporting Theory

We have continued our work on the LEDA platform of combinatorial and geometric computing and we started the PAD project of building a library of PRAM algorithms. In the PAD-project we aim at showing that many non-trivial algorithms can be made efficient in a practical setting where both memory size and number of processors are limited, and we investigate the language features required for expressing complex parallel algorithms. We report on PAD in Section 9.9 and discuss different aspects of LEDA in the other subsections.

All work on LEDA is joint work with Stefan Näher's group in Halle. LEDA is now used at about 1500 academic sites (see `http//www.mpi-sb.mpg.de/LEDA` for a listing). A commercial version of LEDA is marketed by the spin-off company LEDA Software GmbH. There are license holders in the telecommunication industry (ATR (Japan), Comptel (Finland), E-Plus (Germany), France Télécom (France), MCI (USA)), in the graphics industry (Aristo Technologies (USA), Cadabra (Canada), Compass Design (USA), Fuji (Japan), Mentor Graphics (USA), MUS (Germany)), in the automotive industry (Daimler Benz (Germany), Ford (USA), Honda (Japan)), in the computer industry (DEC (USA), IBM (USA), Siemens AG (Germany), Silicon Graphics (USA), SUN (USA)), and other industries (Chevron (USA), CFP (Germany), Dolphin (The Netherlands), Howmedica (Germany), Lufthansa (Germany), Neovista (USA), Prediction (USA), Sony (Japan), VTT (Finland)).

In the past two years we have considerably expanded the functionality of LEDA, in particular, in the area of computational geometry. We now have several low-dimensional geometry kernels and a kernel for higher-dimensional geometry (see Section 9.5), algorithms for Voronoi diagrams, Delaunay diagrams, and convex hulls in arbitrary dimensions, minimum diameter (see Section 9.2), and many other basic geometric problems. Many of these extensions would not have been possible without the theoretical work on how to handle degeneracies in geometric computations and how to realize exact arithmetic efficiently about which we reported in the last progress report. We have also improved the library in some software engineering aspects, e.g., documentation (see Section 9.6).

On the theoretical side we advanced in three respects: We have developed program checkers for many geometric and combinatorial problems (see Section 9.4), we have extended the range of efficient exact arithmetic to a subset of the algebraic numbers (see Section 9.1), and we have started an investigation of running time prediction of real programs (see Section 9.3).

The idea that researchers in algorithms should make their work available in software libraries has caught on with other researchers. In particular, the EU-projects ALCOM-IT and CGAL devote a considerable fraction of their manpower to software development. We participate in both projects (see Section 15). We have recently introduced the concept of LEDA extension packages (LEPs) (see Section 9.7) through which we hope to channel some of this development (and also implementations done through others) into a form compatible with LEDA. We have already succeeded for a significant part of the development in ALCOM-IT and for a small part of the development in CGAL.

9.1 Exact and Efficient Arithmetic Computation

Investigators: Christoph Burnikel, Rudolf Fleischer, Kurt Mehlhorn, and Stefan Schirra

Many computer programs (and basically all programs that perform geometric computations) perform branches based on the sign of arithmetic expressions. Given the fact that the hardware implementations of the basic arithmetic operations incur rounding errors and hence cannot compute signs exactly, it is not surprising that floating point arithmetic is one of the common sources of failure in geometric programs. We believe that the easiest way to robust and correct programs is *exact computation* and have set out to show that exact computation can be achieved at reasonable cost.

Our two-dimensional geometry kernel for objects with rational coordinates performs all numerical computations with floating point arithmetic combined with an on-line error analysis. Exact high-precision integer computation is only used if the error analysis indicates that the sign of the result of a floating point computation is not reliable. This use of so-called floating point filters was first advocated by [5]. Our experience shows that for most algorithms the use of floating point filters incurs a very small overhead in running time.

In [2] we describe an efficient method to compute the sign of real algebraic numbers that are expressible by an explicit expression E using the operations $+$, $-$, $*$, $/$, and $\sqrt[k]{}$ over the integers. The basic method of computing the sign of E is the use of *separation bounds.* A separation bound $sep(E)$ for an expression E is a positive real number that is a lower bound for the value of the expression whenever this value is nonzero. Once a separation bound for E is known, it is easy to compute the true sign of the expression using standard techniques of numerical approximation. In [2] we propose a new separation bound that is strong and, at the same time, easy to compute *automatically.* Previous bounds are difficult to compute [6] or weaker [7, 4]. For example, consider the example given by Ramanujan:

$$E = \sqrt[6]{9(7\sqrt[3]{20} - 19)} - \sqrt[3]{5} - \sqrt[3]{2}$$

Surprisingly, the expression E has value zero. Applying our method gives a separation bound of $sep(E) \approx 2.3 \cdot 10^{-121}$ and the method of Canny [4] yields $sep(E) < 10^{-135{,}000}$.

The LEDA package **real** [3] combines the techniques of separation bounds, multiple-precision arithmetic and floating point filters to a number type that can be used for any explicit algebraic expression. Reals are as easy to use as any native type like `double`. Reals have already been used in the implementation of Voronoi diagrams of line segments and the implementation of parametric search (see Section 9.2).

Still there is much work to do to make exact arithmetic computation more usable and efficient. Firstly, the performance of floating point filters can be improved for specific types of expressions such as $2d$ and $3d$ determinants [1]. Such fine-tuning is however not easy to do for expressions that appear on-line, such as in the LEDA kernel for d-dimensional geometry. Secondly, exact arithmetic in LEDA is restricted to algebraic numbers representable by explicit expressions. In solid modeling, for example, object coordinates are usually given in implicit form as systems of polynomial equations. It would be interesting to design an efficient LEDA package for manipulating this most general type of algebraic numbers.

References

[1] F. Avnaim, J-D. Boissonnat, O. Devillers, F. Preparata, and M. Yvinec. Evaluating signs of determinants using single-precision arithmetic. *Algorithmica*, 17:111–132, 1997.

• [2] C. Burnikel, R. Fleischer, K. Mehlhorn, and S. Schirra. A Strong and Easily Computable Separation Bound for Arithmetic Expressions Involving Square Roots. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 702–709, 1997.

• [3] C. Burnikel, K. Mehlhorn, and S. Schirra. The LEDA class real number. Technical Report MPI-I-96-1-001, Max-Planck-Insitut für Informatik, Saarbrücken, 1996. Available via the URL `http://data.mpi-sb.mpg.de/internet/reports.nsf`.

[4] J.F. Canny. *The Complexity of Robot Motion Planning.* PhD thesis, MIT, 1987.

[5] S. Fortune and C.J. van Wyck. Static analysis yields efficient exact integer arithmetic for computational geometry. *ACM Transactions on Graphics*, 15(3):223–248, 1996.

[6] M. Mignotte. Identification of Algebraic Numbers. *Journal of Algorithms*, 3(3):1982, 1982.

[7] M. Mignotte. *Mathematics for Computer Algebra.* Springer Verlag, 1992.

9.2 Minimum Diameter by Parametric Search

Investigators: Stefan Schirra and Michiel Smid

Parametric search [2] is a powerful algorithmic technique that can be used to solve a large variety of geometric optimization problems. It was folklore that it is impossible to implement. In [3], Schwerdt, Smid, and Schirra report on the implementation of an algorithm due to Gupta, Janardan, and Smid [1] that is based on parametric search. As far as we know, this is the first implementation of a parametric search algorithm.

The algorithm solves the following problem. We are given a set of n points in the plane that are moving at constant but possibly different velocities. The diameter of the points at time t is the largest Euclidean distance among all pairs of points at time t. Our goal is to compute the time t^* at which the diameter is minimum.

This problem can be solved trivially as follows. For each pair of points, the square of their distance defines a quadratic function in t, i.e., a parabola. The minimum diameter is obtained by finding the lowest point on the upper envelope of these $\binom{n}{2}$ parabolas. This gives an algorithm with running time $O(n^2 \log n)$, and using $O(n^2)$ space. It was shown by Gupta, Janardan, and Smid that the problem can be solved, using parametric search, in $O(n \log^3 n)$ time, using $O(n)$ space. The latter algorithm is complicated and not practical; it uses e.g. an optimal parallel sorting algorithm. Therefore, we have implemented a variant of it, which has running time $O(n \log^6 n)$.

In a first implementation we used the number type `double` in all numerical computations. For many problem instances the implementation could not find the minimum diameter. During parametric search an interval containing t^*, the time at which the diameter is minimum, is maintained and repeatedly made smaller until it is finally shrunken to t^*. For many problem instances the interval did not collapse to a single point due to rounding errors and a fairly small interval was reported in the end. We got rid of these precision problems by using the number type `real` described above. The observed running times were reasonable.

Our implementation shows that parametric search is not at all a purely theoretical and totally impractical technique in algorithm design in general and computational geometry in particular.

References

[1] P. Gupta, R. Janardan, and M. Smid. Fast algorithms for collision and proximity problems involving moving geometric objects. *Computational Geometry: Theory and Applications*, 6:371–391, 1996.

[2] N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *Journal of the ACM*, 30:852–865, 1983.

• [3] J. Schwerdt, M. Smid, and S. Schirra. Computing the minimum diameter for moving points: an exact implementation using parametric search. In *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, 1997. To appear.

9.3 Runtime Prediction

Investigators: Ulrich Finkler and Kurt Mehlhorn

Big-O analysis of algorithms is concerned with the asymptotic analysis of algorithms, i.e., with the behaviour of algorithms for large inputs. It does not allow the prediction of actual running times of real programs on real machines and therefore its predictive value is limited.

- An algorithm with running time $O(n)$ is faster than an algorithm with running time $O(n^2)$ for sufficiently large n. Is $n = 10^6$ large enough? Asymptotic analysis of algorithms is of little help to answer this question. It is however true that a well-trained algorithms person who knows program and analysis can make a fairly good guess.
- Algorithms are more and more made available as part of software libraries. For the user of such a library statements of asymptotic running times are almost meaningless as he has no way to estimate the constants involved. After all, the purpose of a tool kit is to hide the implementations from the end user.

The two items above clearly indicate that we need more than asymptotic analysis in order to have a theory with predictive value. *The ultimate goal of analysis of algorithms must be a theory that allows to predict the actual running time of an actual program on an actual machine* with reasonable precision (say within a factor of two). We must aim for the following scenario: When a program is installed on a particular machine a certain number of well-chosen tests is executed in order to learn about machine parameters relevant for the execution of the program. This knowledge about the machine is combined with the analysis of the algorithm to predict running time on specific inputs. In the context of an algorithms library one could even hope to replace statements about asymptotic execution times by statements about actual execution times during installation of the library.

Asymptotic analysis of algorithms approximates the running time $T(x)$ of an algorithm on input x as

$$T(x) \approx \sum_i C_i f_i(x) \tag{9.1}$$

where $f_i(x)$ measures the frequency of execution of a certain set of operations M_i and C_i measures the execution time of M_i on an idealised Random Access Computer. The theoretical analysis provides approximations for the execution frequencies $f_i(x)$, for example in the worst or average case.

Consequently, the goal of a theory which allows the prediction of actual running times is the determination of the constants C_i. One of the main problems in this task is the context dependency of the execution time of an instruction. The execution time of an instruction may vary due to caching etc. by a machine dependent factor which is about 30 for modern workstations. The context dependency contributes a systematic component to the error in time measurements, such that the approach of regression analysis (e.g., [1]) does in general not allow a quantitative statement about the accuracy [2]. We showed recently [2], that it is possible to determine a feasible set of constants despite of the context dependency. The achieved accuracy was sufficient to predict the execution times in a hierarchy of algorithms within a constant factor of less than two if the execution frequencies are known.

The determination of the constants is performed in two phases, a decomposition of the code into fragments and the design of a corresponding set of so-called ‘equivalent code fragments’ and the timing of these equivalent fragments. The timing phase is automated. The main idea is that the fragments in a feasible decomposition have their own ‘fingerprint’ of locality, such that the real context (the program) of a fragment is approximated well.

The next goal must be the extension of the approach of timing equivalent code fragments into a ‘theory of constant factors’. The concern of this theory are the following two problems:

- The automatic decomposition of a program. This is closely related to compiler design, a modified compiler has to analyse and decompose the program into a feasible set of fragments. A first draft of a set of rules for this decomposition is given in [2].
- The calculation of the execution time of these fragments based on the architecture rather than timing. The structural analysis has to provide approximations for the probability of cache misses and pipeline stalls based on the ’fingerprint’ of the code fragment. This breaks the execution time of a fragment down into two components, the effective execution time of each instruction (CPU clock-cycles) and the times to load the data from the different levels of the memory hierarchy into the CPU.

References

[1] R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows.* Prentice-Hall, 1993.

• [2] U. Finkler and K. Mehlhorn. Runtime prediction of real programs on real machines. In *Proceedings 8th ACM-SIAM Symposium on Discrete Algorithms (SODA’97)*, January 1997.

9.4 Program Checking

Investigators: Kurt Mehlhorn, Thomas Schilz, Stefan Schirra, Michael Seel, and Christian Uhrig

Programming is a notoriously error-prone task; this is even true when programming is interpreted in a narrow sense: going from a (correct) algorithm to a program. The standard way to guard against coding errors is program testing. The program is exercised on inputs for which the output is known by other means, typically as the output of an alternative program for the same task. In geometric programming visual inspection of outputs is a powerful testing method, it is however limited to programs operating in low-dimensional space. Program testing has severe limitations:

- It is usually only done during the testing phase of a program. Also, it is difficult to determine the “correct” suite of test inputs.
- Even if appropriate test inputs are known it is usually difficult to determine the correct outputs for these inputs: alternative programs may have different input and output conventions or may be too inefficient to solve the test cases.

Given that program verification, i.e., formal proof of correctness of an implementation, will not be available on a practical scale for some years to come, *program checking* has been proposed as an extension to testing [1, 2]. The cited papers explored program checking in the area of algebraic, numerical, and combinatorial computing.

In [4] and [6] we explore it for geometric computing, data structures and combinatorial computing and we also report on first implementations within the LEDA system.

The first paper has two parts: In the first part we give program checkers for a number of problems in computational geometry: convexity of polygons and polytopes, convex hulls in arbitrary dimensions, Voronoi and Delaunay diagrams and trapezoidal decompositions. We experienced that program checking is sometimes facilitated by augmenting the program checked; for example, a convex hull program that delivers a triangulation of the hull is much easier to check than a program that only returns the hull polytope.

In the second part we report about our practical experience with program checking within the LEDA system. We implemented most checkers described in this paper [5, 3]. In particular, we added checking to a program that computes convex hulls in arbitrary dimensional space. The hull program is structured into four layers: arbitrary precision integer arithmetic, linear algebra over the rationals, geometric primitives for higher dimensional space, and finally the convex hull program. We made the last three layers self-checking. The addition of self-checking increased the running time by less than a factor of two, decreased debugging time considerable, and greatly increased our confidence that the implementation produces correct results.

Large software systems are built in layers. As the number of layers increases debugging becomes harder as errors become harder to locate. If all programs in a system are self-checking any program can safely assume that all the lower level programs used by it are correct. *We believe that program checking will play a prominent role in the construction of software libraries and have started to add checking facilities to LEDA. We also believe that programs in a library should be designed for check ability.* The exploration of this thesis will be a topic of future work.

In [6] checkers for a large number of graph and network algorithms are developed. For network problems that can be described as linear programs efficient checkers can usually be derived from linear programming duality. For problems that cannot be specified as linear programs, e.g., the strongly connected component problem, the development of checkers is more demanding.

References

[1] M. Blum and S. Kannan. Programs That Check Their Work. In *Proc. of the 21th Annual ACM Symp. on Theory of Computing*, 1989.

[2] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. In *Proc. 22nd Annual ACM Symp. on Theory of Computing*, pages 73–83, 1990.

• [3] K. Mehlhorn, M. Müller, St. Näher, and J. Ziegler. An implementation of a convex hull algorithm. Available via the URL `http://www.mpi-sb.mpg.de/~mehlhorn/Programs.html`, 1997.

• [4] K. Mehlhorn, S. Näher, T. Schilz, S. Schirra, M. Seel, R. Seidel, and Ch. Uhrig. Checking Geometric Programs or Verification of Geometric Structures. In *Proc. of the 12th Annual Symposium on Computational Geometry*, pages 159–165, 1996.

• [5] K. Mehlhorn and St. Näher. Checking geometric structures. Available via the URL `http://www.mpi-sb.mpg.de/~mehlhorn/Programs.html`, 1997.

• [6] Matthias Metzler. Ergebnisüberprüfung bei Graphenalgorithmen. Master's thesis, Max-Planck-Insitut für Informatik, 1997.

9.5 Geometry Kernels and Geometric Algorithms

Investigators: Kurt Mehlhorn, Stefan Schirra, Michael Seel, and Christian Uhrig

At the time of the last progress report we had only a very small number of reliable geometric algorithms. We had a two-dimensional kernel for objects with rational coordinates and we had correct algorithms for convex hulls, Delaunay diagrams, Voronoi diagrams, and line segment intersection. We had no kernel for higher dimensional geometry, we had no algorithms in dimensions higher than two, and the two-dimensional kernel could only be used with the integer arithmetic provided by LEDA; it could not be used in conjunction with any of the exiting other developments in exact geometric computation. All of this has changed.

A new kernel [2] for two- and three-dimensional geometry has been developed in cooperation with the CGAL-sites FU Berlin, INRIA Sophia-Antipolis, Utrecht University, and ETH Zürich.

A major design issue for the new lower-dimensional kernel was to support the development of *generic code*. Geometric algorithms consist of different layers, the bottommost layers being geometric primitives, i.e., basic objects and predicates, and the arithmetic used to do the calculations in the predicates. The kernel is designed to work with a variety of implementations of predicates and subtasks, and representations of geometric objects. That allows to easily interchange components as long as they have the same syntax. Generalization could have been achieved by using inheritance and virtual functions or by using templates. We opted for writing template code as it has the advantage that it incurs no extra runtime costs. Perhaps the most important aspect of the generalization of the kernel is the arithmetic. Basic geometric types are parametrized by number types. This idea was already investigated in a master thesis [4] at MPI in 1993. By selecting a number type the user can influence the efficiency, the exactness, and hence the correctness of the geometric computations. Especially, all the LEDA number types concerned with efficient exact computation can be used. Furthermore, a user can choose between implementations of the geometric objects based on representation of points by Cartesian (or affine) coordinates and representations based on homogeneous coordinates. The latter allows to reduce many computations involved in geometric algorithms to calculations over the integers, since divisions can be avoided with this representation. The generalization of the kernel allows the user to select the representations and arithmetic that are most appropriate for his application. A user can even partially replace kernel primitives by his own implementations.

The higher-dimensional kernel [3] was developed together with an implementation of a randomized incremental higher-dimensional convex hull algorithm [1]. The original version of the higher-dimensional kernel was based on arbitrary precision integer arithmetic, a version with objects parametrized by number types is now available as well. The higher-dimensional kernel is layered into a linear algebra component, the geometric objects, and an application and test layer.

Both kernels use a handle-rep scheme and the LEDA memory management to improve memory consumption and optimize runtime behavior. The part of the lower-dimensional kernel developed at MPI as well as the higher-dimensional kernel will become LEDA extension packages. Since many current compilers do not yet support the latest enhancements of C++ related to templates very well, the higher-dimensional kernel will be distributed in the version based on LEDA's integer arithmetic only.

References

[1] K.L. Clarkson, K. Mehlhorn, and R. Seidel. Four results on randomized incremental constructions. *Comput. Geom. Theory Appl.*, 3(4):185–212, 1993.

- [2] A. Fabri, G.-J. Giezeman, L. Kettner, S. Schirra, and S. Schönherr. The CGAL kernel : a basis for geometric computation. In Ming C. Lin and Dinesh Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering (WACG96)*, pages 191–202. Springer LNCS 1148, 1996.

- [3] K. Mehlhorn, M. Müller, S. Näher, S. Schirra, M. Seel, C. Uhrig, and J. Ziegler. A computational basis for higher-dimensional computational geometry and its applications. In *Proceedings of 13th Annual ACM Symp. on Computational Geometry*, 1997. (to appear).

[4] M. Neukirch. Grundlegende geometrische Datenstrukturen und Algorithmen für LEDA. Diplomarbeit (in German), 1993.

9.6 New Functionality in LEDA

Investigators: Kurt Mehlhorn, Michael Seel, and Christian Uhrig

We improved several already existing datatypes and features of LEDA:

- we adapted several basic data types to fit into the STL (Standard Template Library) philosophy; lists and arrays got iterators and STL-compliant operations, graphs got iterators and accessors.
- several graphical functions got a postscript interface to enable the transfer of graphical information towards a common graphics description language.
- the number type real was improved and got new features (e.g., n-th roots in expressions)
- the LEDA memory management can be adapted to handle arbitrary block size.
- the 2-dimensional geometry was extended and 3-dimensional geometry is introduced.
- the documentation features of LEDA have been improved. All interface datatypes have the documentation included in the header files in form of manual comments. This enforces consistency between the specification and implementation and enables us to provide different documentation schemes. We provide online help by a program called `xlman`, which gives the manual view sorted by data types. Additionally there's an HTML online manual.

9.7 LEDA Extension Packages

Investigators: Kurt Mehlhorn, Michael Seel, and Christian Uhrig

LEDA extension packages are a new feature of the LEDA project structure. Up to two years ago, most of LEDA has been developed by a small group of persons under the tight supervision of Stefan Näher; no code went into the system that was not thoroughly understood by either Stefan Näher or Christian Uhrig. The growing numbers of contributors and the fact that Stefan Näher has new responsibilities as a professor has forced us to a change of the project structure. We decided to split LEDA into a core system (the actual LEDA version) and to shift enhancements into additional software packages.

LEDA extension packages (LEPs) extend LEDA into particular application domains and areas of algorithmics not covered by the core system. LEPs satisfy requirements, which guarantee compatibility with the LEDA philosophy. LEPs have a LEDA-style documentation, they are implemented as platform independent as possible and the installation process allows a close integration into the LEDA core library.

Currently, there are no released LEPs available, but there are several LEPs under construction right now:

- PQ-Trees (coordinated by Sebastian Leipert, Köln)
  PQ-trees are a data structure being used to represent the permutations of a set U in which various subsets of U occur consecutively. Along with the data structure, efficient algorithms for manipulating PQ-trees are given, requiring linear time in the size of the input.
- Dynamic Graph Algorithms (coordinated by David Alberts, Halle)
  A dynamic graph algorithm is a data structure operating on a graph supporting two types of operations: updates and queries. An update is a local change of the graph and a query is a question about a certain property of the current graph. The aim of such a data structure is to use structural information about the current graph in order to handle an update faster than by the obvious solution, that is recomputing everything from scratch with a static algorithm. Usually, queries take less time than updates, and the sequence of operations (updates and

queries) is not known in advance. More information on this package can be found in `http://www.informatik.uni-halle.de/~alberts/lepdga.html`.

- Homogeneous planar CGAL Geokernel (coordinated by Stefan Schirra, Saarbrücken)
  This package realizes geometric objects and primitives on the objects to implement algorithms within planar computational geometry. The objects use a homogeneous coordinate scheme and are parametrized by a numbertype. Thus they extend the arithmetically "hard-wired"" homogeneous geometric 2-dimensional objects of the LEDA core system.

- Homogeneous d-dimensional Geokernel (coordinated by Michael Seel, Saarbrücken)
  This package provides d-dimensional geometric objects and primitives on the objects. They extend the homogeneous 2-dimensional kernel of the LEDA core system to d dimensions.

- Automatic Graph Drawing Library
  This package is developed in the frame of the DFG-project on Automatic Graph Drawing (see Section 15.2 for details).

9.8 Plans and Open Problems in LEDA

We mentioned already a number of open problems and plans of further development in the preceding sections. In this section we address two additional issues.

All algorithms in LEDA assume that they are executed in main memory and many of the algorithms will become extremely slow if they are not. We plan to extend LEDA to secondary memory algorithms. To this end, Andreas Crauser and Kurt Mehlhorn taught a course on secondary memory algorithms in WS96/97 and run a programming lab on the subject in the summer of 97. They hope to have a small system up at the end of 97. Jop Sibeyn and Jesper Träff bring their expertise in parallel algorithms (see Section 5.4) into this undertaking.

A second issue are guarantees. Our algorithms are carefully designed and tested and the recent development of checkers has further increased their reliability. But, of course, none of the developers of LEDA would be willing to sign a legally binding contract making any claims concerning correctness and/or efficiency. Can we change this?

9.9 PAD: A library for the implementation of PRAM algorithms

Investigator: Jesper Larsson Träff

For more than a decade the PRAM ("Parallel Random Access Machine") has been the primary theoretical model for the design of parallel algorithms, and by now a wealth of fast and (work) efficient algorithms that can utilize a large number of processors exist for a very diverse set of problems [5]. Moreover a number of tools and techniques have evolved which enable the development of algorithms for similar problems in a routine fashion. These facts speak strongly in favor of the PRAM as a *design model* for parallel algorithms: no other model of parallel computation rivals the PRAM in these respects.

In order to make PRAM algorithms more accessible to the practitioner, and in order to identify the language features required to express these algorithms in a clean and natural fashion, it seems a natural step to construct a library of concretely implemented fundamental PRAM algorithms, which also supports some of the basic PRAM design paradigms. Due to the intricateness of the algorithms even for fundamental, easy-to-specify parallel operations (list ranking being the paradigmatic example) library support is even more essential than is the case for sequential algorithms. Skepticism about the physical realizability of some reasonable approximation of the PRAM is probably what has so far held back such an undertaking.

A parallel algorithm is useful in practice if: a) it exhibits speed-up when compared to a good sequential algorithm for realistic problem sizes and realistic number of processors; and b) it can be implemented with a reasonable effort. PRAM algorithms, which are theoretically efficient (work-optimal), satisfy the first requirement asymptotically, but often the magnitudes of the constants involved are not known, and may render the algorithm uninteresting. Although many PRAM algorithms are conveniently expressible in pseudo-code, the issue of ease of implementation is rarely considered on a larger scale. Notable exceptions are the NESL project of Blelloch [2], and the MasPar implementations of Hsu, Ramachandran and Dean [4].

The **PAD** library of *PRAM Algorithms and Data Structures* aims at supporting the concrete implementation of parallel algorithms from the theoretical literature, and showing that non-trivial algorithms can be made efficient in a practical setting where both memory size and number of available processors are limited. **PAD** is implemented in the C-based PRAM programming language **Fork95** [6], and efficiency is measured using a simulator for a realistic PRAM. The first phase of **PAD** has been concerned with making fundamental algorithms and basic paradigms available in a clean and efficient fashion.

The **PAD** library is intended for use with the SB-PRAM, a 4096 (virtual) processor hardware emulation of a CRCW PRAM which is currently reaching completion at the University of Saarbrücken [1, 3]. However, **PAD** and its implementation language **Fork95** should not be thought of as restricted to this machine. Even in the absence of a concrete PRAM the library can contribute to clarify the extent to which PRAM algorithms can be made useful in practice by making it possible to experiment with and compare different algorithms, and stimulate renewed interest in areas where existing algorithms are too inefficient. In more abstract terms, a collection of efficient PRAM algorithms can serve as an intermediate step between abstract parallel algorithms and their implementation on machines different from the PRAM (see for example [9]). For the theoretician, a well-engineered and extensive library makes it possible to implement and experiment with new ideas. Finally, a PRAM algorithms library is of considerable value in the communication and teaching of parallel algorithms.

The **PAD** project has been running for about two years, and a first version, consisting of the **Fork95** compiler, a simulator for the SB-PRAM and the **PAD** library package, including a manual [11] is available via the internet:

`http:/www.mpi-sb.mpg.de/guide/activities/alcom-it/PAD/`

The first version of **PAD** consists of the following sublibraries:

Prefix library: Operations on arrays, mostly of the prefix sums type: all prefix sums, segmented (grouped) prefix sums, searching in unordered arrays, support for range queries [13].

Merge library: Operations for searching and merging of ordered arrays; sorting (merge sort, quicksort, bucket sorting for small integers) [12].

List library: List ranking (pointer jumping and various deterministic, work-optimal algorithms from the literature) and other operations on lists [14].

Tree library: Euler tour technique, lowest common ancestors, generic (binary) tree contraction and other operations on trees [15].

Parallel Dictionaries: Parallel dictionaries based on 2-3 trees [10].

Short overviews of **PAD** and **Fork95** were presented in [7, 8].

References

[1] F. Abolhassan, R. Drefenstedt, J. Keller, W. J. Paul, and D. Scheerer. On the physical design of PRAMs. *The Computer Journal*, 36(8):756–762, 1993.

[2] G.E. Blelloch. Programming parallel algorithms. *Communications of the ACM*, 39(3):85–97, 1996.

[3] A. Formella, J. Keller, and T. Walle. HPP: A high performance PRAM. In *Euro-Par'96 Parallel Processing*, volume 1124 of *Lecture Notes in Computer Science*, pages 425–434, 1996.

[4] T.-S. Hsu and V. Ramachandran. Efficient massively parallel implementation of some combinatorial algorithms. *Theoretical Computer Science*, 162(2):297–322, 1996.

[5] J. JáJá. *An Introduction to Parallel Algorithms.* Addison-Wesley, 1992.

[6] C.W. Keßler and H. Seidl. The Fork95 Parallel Programming Language: Design, Implementation, Application. *International Journal of Parallel Programming*, 25(1):17–50, 1997.

• [7] C.W. Keßler and J.L. Träff. A library of basic PRAM algorithms and its implementation in FORK. In *8th Annual ACM Symposium on Parallel Algorithms and Architechtures (SPAA)*, pages 193–195, 1996. Research Summary.

• [8] C.W. Keßler and J.L. Träff. Language and library support for practical PRAM programming. In *5th Euromicro Workshop on Parallel and Distributed Processing (PDP'97)*, pages 216–221, 1997.

[9] A. Ranade, S. Cheng, E. Deprit, J. Jones, and S. Shih. Parallelism and locality in priority queues. In *Sixth IEEE Symposium on Parallel and Distributed Processing*, pages 490–496, 1994.

• [10] J.L. Träff. Explicit implementation of a parallel dictionary. Technical Report SFB 124-D6 10/95, Universität des Saarlandes, Sonderforschungsbereich 124, VLSI Entwurfsmethoden und Parallelität, 1995. 53 Pages.

• [11] J.L. Träff. *PAD – A Library of PRAM Algorithms and Data structures. User Manual*, October 1996. Version 1. Available via the URL `http://www.mpi-sb.mpg.de/guide/activities/alcom-it/PAD/`.

• [12] J.L. Träff. Parallel searching, merging and sorting. Technical Report SFB 124-D6 1/96, Universität des Saarlandes, Sonderforschungsbereich 124, VLSI Entwurfsmethoden und Parallelität, 1996. 45 Pages.

• [13] J.L. Träff. Prefix sums and other operations on parallel arrays. Technical Report SFB 124-D6 6/96, Universität des Saarlandes, Sonderforschungsbereich 124, VLSI Entwurfsmethoden und Parallelität, 1996. 49 Pages.

• [14] J.L. Träff. List ranking and other operations on parallel lists. Technical Report SFB 124-D6 3/97, Universität des Saarlandes, Sonderforschungsbereich 124, VLSI Entwurfsmethoden und Parallelität, 1997. In preparation.

• [15] J.L. Träff. Parallel tree operations. Technical Report SFB 124-D6 4/97, Universität des Saarlandes, Sonderforschungsbereich 124, VLSI Entwurfsmethoden und Parallelität, 1997. In preparation.

10 Visitors

Since March 1995, a total of 86 researchers have visited our group.

Prof. Tetsuo Asano	10/05/96 – 12/05/96	Osaka Electro-Comm. University Osaka, Japan
Prof. Gerhard Barth	10/02/95	Daimler-Benz AG Ulm, Germany
Sebastiano Battiato	04/04/95 – 29/04/95	Acicatena, Italy
Therese Biedl	08/01/97 – 10/01/97	Rutgers University Piscataway, New Jersey, USA
Dany Breslauer	23/10/95 – 31/10/95 15/03/96 – 30/06/96	BRICS, University of Aarhus Aarhus, Denmark
Gerth Brodal	04/09/95 – 01/06/96	BRICS, University of Aarhus BRICS, University of Aarhus
Prof. Nicoló Cesa-Bianchi	29/07/96 – 01/08/96	Universitá di Milano Milano, Italy
Soumen Chakrabarti	02/06/96 – 08/07/96	University of California Berkeley, California, USA
Dr. Bogdan Chlebus	17/09/95 – 30/09/95 16/06/96 – 30/06/96	Warsaw University, Warsaw, Poland
Thomas Christof	22/10/95 – 25/10/95	Heidelberg, Germany
Artur Czumaj	10/03/97 – 21/03/97	Universität-GH Paderborn Paderborn, Germany
Prof. Gautam Das	15/09/95 – 15/01/96 01/07/96 – 15/08/96	University of Memphis Memphis, USA
Prof. Martin Dietzfelbinger	15/09/95 – 17/09/95	Universität Dortmund Dortmund, Germany
Kzysztof Diks	17/09/95 – 08/10/95	Uniwersytet Warszawski Warszawa, Poland
Prof. Yeffim Dinitz	20/05/96 – 23/05/96	Technion, Haifa, Israel
Prof. Hristo Djidjev	22/06/95 – 27/06/95 15/06/96 – 30/06/96	Rice University Houston, Texas, USA
Dr. Devdatt Dubhashi	09/08/95 – 16/08/95	SPIC Mathematical Institute Chennai, India
Prof. Peter Eades	11/06/95 – 15/06/95	University of Newcastle Newcastle, Australia
Dr. Yves Escaig	01/06/95 – 31/08/95	INSA de Rouen Mont Saint Aignan, France
Dr. Andreas Fabri	10/07/95 – 12/07/95 30/10/95 – 31/10/95 15/07/96 – 09/08/96	INRIA Sophia Antipolis cedex, France
Quingwen Feng	08/10/96 – 11/10/96	University of Newcastle, Australia
Paolo Ferragina	16/11/95 – 18/11/95	Universitá di Pisa Pisa, Italy

Paul Fischer	29/07/95 – 31/07/96	Universität Dortmund Dortmund, Germany
Ulrich Fößmeier	25/11/96 – 29/11/96 29/01/97 – 31/01/97	Universität Tübingen Tübingen, Germany
Geert-Jan Giezeman	10/07/95 – 11/07/95	Utrecht University Utrecht, The Netherlands
Prof. Andrew Goldberg	03/12/96 – 06/12/96	NEC Research Institute Princeton, New Jersey, USA
Dr. Mordecai Golin	07/06/95 – 09/07/95 12/07/96 – 09/08/96	The Hong Kong University of Science and T., Hong Kong
Magnus Halldorson	08/05/95 – 28/06/95	University of Iceland Reykjavik, Iceland
Prof. Ramesh Hariharan	12/07/96 – 15/07/96	Indian Institute of Science Bangalore, India
Jaap-Henk Hoepman	30/05/95 – 09/06/95	CWI, Amsterdam, The Netherlands
Piotr Indyk	25/02/96 – 09/03/96 24/06/96 – 24/08/96	Stanford University Stanford, California, USA
Prof. Giuseppe Italiano	09/11/96	University of Venice, Venice, Italy
Sanjiv Kapoor	18/09/95 – 15/12/95 15/05/96 – 31/07/96	Indian Institute of Technology, New Delhi, India
Jyrki Katajainen	12/04/96 – 16/04/96	University of Copenhagen, Copenhagen, Denmark
Prof. Naoki Katoh	09/05/96 – 12/05/96	Kobe University of Commerce Kobe, Japan
Prof. Michael Kaufmann	01/11/96 – 31/01/97	Universität Tübingen Tübingen, Germany
Prof. John Kececioglu	20/11/95 – 23/11/95 24/06/96 – 20/07/96	University of Georgia Athens, Georgia, USA
Prof. Danny Krizanc	30/09/96 – 08/11/96	Carleton University Ottawa, Canada
Dietmar Kühl	09/05/96 – 10/05/96	Universität Konstanz Konstanz, Germany
Prof. Larry Larmore	15/07/96 – 22/07/96	University of Las Vegas Las Vegas, Nevada, USA
Hoong Chuin Lau	15/07/96 – 07/08/96	Tokyo Institute of Technology Tokyo, Japan
Prof. Ulrich Lauther	27/07/95 12/02/97	Siemens AG München, Siemens
Sebastian Leipert	17/02/97 – 21/02/97	Universität zu Köln Köln, Germany
Maciej Liskiewicz	05/06/95 – 10/06/95	International Computer Science Institute Berkeley, California, USA

Prof. Heikki Mannila	07/09/95 – 31/08/96	University of Helsinki Helsinki, Finland
Marios Mavronikolas	16/11/95 – 23/11/95	University of Cyprus, Cyprus
Dr. Michael Mitzenmacher	27/05/96 – 31/05/96	DEC SRC, Palo Alto, USA
Eric Monfroy	27/11/95	CRIN - CNRS, INRIA Vandoeuvre-Les-Nancy, France
Prof. Maharaj Mukherjee	22/04/96 – 31/07/96	Indian Institute of Technology Kharagpur, India
Prof. Eugene Myers	14/07/95	University of Arizona Tucson, Arizona, USA
Prof. Umberto Nanni	09/11/96	Università degli Studi di Roma Rome, Italy
Dr. Joseph Naor	05/02/96 – 17/02/96	Technion, Haifa, Israel
Prof. Frank Nielsen	12/10/95 – 28/10/95	Technical University of Denmark Lyngby, Denmark
Thomas Odenthal	14/01/97 – 15/01/97	Columbia University New York, USA
Giulio Pasqualore	09/11/96	Università degli Studi di Roma Rome, Italy
Dr. Marco Pelegrini	23/07/95 – 23/08/95	King's College London, London, United Kingdom
Andrzej Pelc	23/02/97 – 26/02/97	Université de Quebec à Hull Hull, Canada
Dr. Rajeev Raman	07/08/95 – 04/09/95 30/09/96 – 31/10/96	King's College London, London, United Kingdom
Prof. Pandu Rangan	29/11/96 – 26/12/96	Indian Institute of Technology Madras, India
Dr. Desh Ranjan	01/06/95 – 15/08/95 01/06/96 – 15/08/96	New Mexico State University Las Cruces, New Mexico, USA
Dr. P. Srinivasa Rao	22/05/95 – 07/07/95	University of Hyderabat Hyderabat, India
Prof. R. Ravi	17/07/95 – 15/08/95 15/02/97 – 22/02/97	Carnegie Mellon University Pittsburgh, USA
Joachim Rieger	01/08/95 – 29/02/96	Universidade de Sao Paulo Sao Carlos, Brasil
Suleyman Sahinalp	21/04/95	UMIACS, University of Maryland, USA
Dr. Peter Sanders	17/03/97	Universität Karlsruhe Karlsruhe, Germany
Sunita Sarawagi	02/06/96 – 08/07/96	University of California Berkeley, California, USA
Réné Schott	27/11/95 – 29/11/95	CRIN - CNRS, INRIA Vandoeuvre-Les-Nancy, France

Joachim Schrod	06/12/95	Technische Universität Darmstadt Darmstadt, Germany
Assaf Schuster	04/12/95 – 06/12/95	The Hebrew University Jerusalem, Israel
Benno Schwikowski	16/12/96 – 17/12/96	Universität Bonn Bonn, Germany
Prof. Micha Sharir	30/08/96	Tel Aviv University Tel Aviv, Israel
Prof. Hans-Ulrich Simon	25/09/95 – 29/09/95	Universität Dortmund Dortmund, Germany
Prof. Vaclav Skala	08/06/96 – 23/06/96	University of West Bohemia Plzen, Czech Republic
Prof. Michiel Smid	03/04/96 – 14/04/96 01/07/96 – 02/08/96	Otto-Von-Guericke-Universität Magdeburg, Germany
Dr. Aravind Srinivasan	22/03/95 – 30/04/95	National University of Singapore Singapore, Republic of Singapore
Yannis Stamatiou	09/12/96 – 18/12/96	University of Patras Patras, Greece
Dr. Boris Steipe	17/05/96	München, Germany
Dr. Torsten Suel	14/12/96 – 18/12/96	Bell Laboratories Murray Hill, New Jersey, USA
Dr. Stefan Thienel	11/03/96 – 12/03/96	Universität zu Köln Köln, Germany
Dr. Dimitrios Thilikos	09/12/96 – 11/12/96	Utrecht University Utrecht, The Netherlands
Prof. Vijay Vazirani	17/07/95 – 01/08/95	Georgia Institute of Technology Atlanta, Georgia, USA
Dr. Martin Vingron	18/03/96	DKFZ, Heidelberg, Germany
Ralf Wastl	30/08/96	Rottenburg, Germany
Karsten Weihe	09/05/96 – 10/05/96	Universität Konstanz Konstanz, Germany
Prof. Sue Whitesides	15/09/95	McGill University Montreal, Canada
Prof. Alexander Zelikowsky	09/12/96 – 14/12/96	University of Virginia Charlottesville, Virginia, USA

11 Journal and Conference Activities

11.1 Editorial Positions

Kurt Mehlhorn has been an editor of *Algorithmica* (since 1985), *Computational Geometry: Theory and Applications* (since 1990), *Information and Computation* (since 1985), *Int'l Journal of Computational Geometry* (since 1990), *Discrete and Computational Geometry* (since 1988), and *SIAM Journal on Computing* (since 1988).

11.2 Conference Positions

11.2.1 Membership in program committees

Susanne Albers: 5th Scandinavian Workshop on Algorithm Theory (SWAT 1996), Reykjavik, Iceland.

Shiva Chaudhuri: 16th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'96), Bangalore, India.

Torben Hagerup: 3rd Italian Conference on Algorithms and Complexity (CIAC'97), Rome, Italy; and 13th Symp. on Theoretical Aspects of Computer Science (STACS'96), Grenoble, France.

Kurt Mehlhorn: 5th Annual European Symposium on Algorithms (ESA'97), Graz, Austria; and Workshop on Algorithm Engineering (WAE'97), Venice, Italy.

Petra Mutzel: Graph Drawing 1997 Symposium (GD'97), Rome, Italy.

Philippas Tsigas: 11th International Workshop on Distributed Algorithms (WDAG'97), Saarbrücken, Germany.

11.2.2 Membership in organizing committees

Marina Papatriantafilou, Philippas Tsigas: 11th International Workshop on Distributed Algorithms (WDAG'97). More information about the workshop can be found in `http://www.mpi-sb.mpg.de/~wdag`.

11.3 Workshops/Schools

11.3.1 Teaching

Stefan Schirra: Advanced School on "Algorithmic Foundations of Geographical Information Systems", CISM, Udine, Italy, September 16–20, 1996. He gave talks on *Precision and Robustness in Geometric Computation* and on *A Computational Geometry Algorithms Library.*

Kurt Mehlhorn, Petra Mutzel: Summer School on "Discrete Optimization", Hamburg, September 9–13, 1996, in the series of "Hamburger Sommerschulen" organized by the German Mathematical Society (DMV). The school was supported by the Volkswagen-Foundation. K. Mehlhorn gave a talk on *Exact Geometric Computations with Algebraic Numbers*; P. Mutzel gave a talk on *Automatic Graph Drawing.*

Petra Mutzel: Seventh Australian Workshop on "Combinatorial Algorithms" (AWOCA), Magnetic Island, Queensland, Australia, July 15-19, 1996. She gave a series of lectures on *Polyhedral Combinatorics.*

11.3.2 Organization

Marina Papatriantafilou, Philippas Tsigas: School on “Distributed Computing”, to be held at the Max-Planck-Institut für Informatik, Saarbrücken, September 1997. The school is supported by the ALCOM-IT project.

Jop Sibeyn: 7th Workshop on Algorithms for Future Technologies (ALTEC–VII), to be held at the Max-Planck-Institut für Informatik, Saarbrücken, on May 21-24 1997. More information can be found in `http://www.mpi-sb.mpg.de/~jopsi/altec.html`.

12 Teaching Activities

The group contributes intensively to the curriculum of the Department of Computer Science at the Universität des Saarlandes. We teach core courses (like “Praxis des Programmierens”, “Datenstrukturen und Algorithmen”) and specialized courses. The details follow.

Summer Semester 1995

LECTURES:
Praxis des Programmierens (C. Rüb)
Optimierung (K. Mehlhorn, P. Mutzel)
Geometrische Algorithmen und ihre Implementation (M. Smid, S. Schirra)
SEMINARS:
Pattern Matching (T. Hagerup, H.P. Lenhof)
PROJECT CLASSES:
On-Line-Algorithmen (K. Mehlhorn, R. Fleischer)
Effiziente Algorithmen unter LEDA (K. Mehlhorn, C. Uhrig)

Winter Semester 1995/96

LECTURES:
C-Blockkurs (P. Müller, M. Seel)
Praxis des Programmierens (H.P. Lenhof)
Komplexitätstheorie (T. Hagerup)
Online Algorithmen (S. Albers)
Approximation Algorithms (N. Garg)
Graph Drawing (P. Mutzel, S. Saluja)
Graph and Network Algorithms (C. Zaroliagis)
PROJECT CLASSES:
Geometrische Algorithmen in LEDA (K. Mehlhorn, S. Schirra, C. Uhrig)

Summer Semester 1996

LECTURES:
C-Blockkurs (M. Seel)
Praxis des Programmierens (H.P. Lenhof)
Informatik IV (G. Hotz, K. Mehlhorn, R. Fleischer)
Parallele Algorithmen (J. Sibeyn, J.L. Träff)
Geometrische Algorithmen und ihre Implementation (S. Schirra)

SEMINARS:
Das Problem des Handlungsreisenden (S. Albers)
Dynamic Graph Algorithms (C. Zaroliagis)
PROJECT CLASSES:
Automatisches Zeichnen von Graphen (P. Mutzel)

Winter Semester 1996/97

LECTURES:
C-Blockkurs (P. Müller)
Praxis des Programmierens (R. Fleischer, P. Mutzel)
Datenstrukturen und Algorithmen (T. Hagerup)
Algorithmen für sehr große Datenmengen (K. Mehlhorn, A. Crauser)
Distributed Algorithms (M. Papatriantafilou, P. Tsigas)
PROJECT CLASSES:
Dynamic Graph Algorithms (C. Zaroliagis)
Visualisierung von Online-Algorithmen (S. Albers, R. Fleischer)
Geometrische Algorithmen (K. Mehlhorn, S. Schirra)

Summer Semester 1997

LECTURES:
C-Blockkurs (M. Seel)
Praxis des Programmierens (J. Sibeyn, J.L. Träff)
Optimierung (T. Hagerup)
Der Handlungsreisende und seine Verwandten (P. Mutzel)
Parallele Algorithmen (C. Zaroliagis)
SEMINARS:
Algorithmen für große Datenmengen (K. Mehlhorn, A. Crauser)
Online-Algorithmen (S. Albers, R. Fleischer)
PROJECT CLASSES:
Algorithmen für große Datenmengen (K. Mehlhorn, A. Crauser)
Geometrische Algorithmen (C. Burnikel, S. Schirra)
Implementation, Visualisation, and Evaluation of Distributed Algorithms using Simulation Platforms (M. Papatriantafilou, P. Tsigas)
Implementation of a Job Scheduler for a Parallel Mesh-Connected Cluster System K. Mehlhorn, J. Gergov, O. Kohlbacher)
Visualisierung von Online-Algorithmen (R. Fleischer)

Our group offers continually an advance course, called “Selected Topics in Algorithms”, on various (advanced) topics in algorithms and complexity. This course is actually a sequence of mini-courses on graduate-level; each mini-course is taught by a set (usually singleton) of instructors which are group members and/or visitors. The advance course is mainly intented to our PhD students, but it is also attended by many members of the group. The course is organized by Jesper Träff and Christos Zaroliagis. Topics treated from March 1995 until today were:

Algorithmic problems in parametric CAD systems (V. Capoyleas)
Computing with weak sources of randomness (A. Srinivasan)

Matrix Searching (P. Bradford)
Fault Tolerant Distributed Computing (M. Papatriantafilou, P. Tsigas)
Approximating Independent Sets (M. Halldorsson)
Data Structures for Edge-connectivity Problems (S. Arikati)
Quantum Computing (R. Hariharan, S. Mahajan)
Applications of Computational Geometry to Orientation Problems in CAM (P. Gupta)
Super-fast and efficient string matching (L. Gąsieniec)
Agnostic PAC-Learning of decision trees (D. Gunopulos)
Probabilistic Aspects of Online Algorithms (S. Albers)
Computing Homology Groups of Simplicial Complexes (K.V. Subrahmanyam)
Combinatorial Optimization (N. Garg)
Randomized Algorithms (C. Pandu Rangan)

Furthermore, our group contributes to the curricula of other Universities. Members of our computational geometry group have given a series of lectures in the frame of a computational geometry graduate course for PhD students at the Department of Computer Science in the University of Nancy (LORIA-CRIN Nancy, France). The lectures given were:

Exact Geometric Computation and Implementation (C. Burnikel)
Geometric Program Checking (M. Seel)
Parametric Search and Geometric Applications (S. Schirra)
Divide-and-Conquer Algorithms based on Geometric Sampling (E. Ramos)

Susanne Albers taught a mini-course on *Competitive Online Algorithms* at BRICS, Aarhus University, Denmark, in August 1996. Detailed lecture notes are available as BRICS Lecture Series, LS-96-2, September 1996.

13 Dissertations and Habilitations

13.1 Dissertations

Completed:
Burnikel, Ch.: Precision and Degeneracy in Geometric Computations. April 1996.
Schwarzenecker, E.: Ein NP–vollständiges Problem aus der Kartographie. September 1995.
Thiel, Ch.: Schnitt von Polyedern in höheren Dimensionen. April 1996.

On-going:
Bast, H.: Provably Optimal Scheduling on Uniform Processors. December 1997[4].
Crauser, A.: External Memory Algorithms in Theory and Practice.
Finkler, U.: Design of Correct and Efficient Algorithms: Theoretical Results and Runtime Prediction of Implementations in Practice. June 1997[4].
Gergov, J.: Approximation Algorithms for Dynamic Storage Allocation.
Hundack, C.: Structuring Graphs and Algorithmic Applications.
Kohlbacher, O.: Simulation of Protein Reactions.
Meyer, U.: Parallel algorithms for large data sets.
Müller, P.: Parallel Algorithms for MD-Simulations of Synthetic Polymers.
Priebe, V.: Probabilistic analysis of combinatorial algorithms. October 1997[4].

---

[4] Expected completion date.

Reinert, K.: Combinatorial optimization for sequence alignment.

Schilz, T.: Effiziente Algorithmen für das verteilte Rechnen auf Workstationclustern. September 1997[4].

Seel, M.: Intersection of Polyhedra in 3-Space.

Ziegler, T.: Graph Drawing Using Planarization.

13.2 Habilitations

The habilitation procedure of Shiva Chaudhuri is on-going.

14 Organization of our Group

The group meets two to four times a week at 1.30 pm.

On Monday and Wednesday (1.30 - 2.15) we have our noon seminar. It lasts about 45 minutes and is reserved for presentations of new results and ongoing research. We also ask our guests to give presentations in the noon seminar.

On Tuesday and Thursday (1.30 - 3.00) we run the "Selected Topics in Algorithms" advance course. This course is reserved for two to four week intensive treatments of subjects of current interest and it is organized by Jesper Träff and Christos Zaroliagis. More information can be found in Section 12.

There also three other meetings:

(a) The "executive committee meeting" that runs on a weekly basis and makes the day to day decisions concerning the group. Elected members are: A. Crauser, L. Gąsieniec, E. Haak, J. Könemann, K. Mehlhorn, J. Sibeyn and C. Storb.

(b) The "group-meeting" that runs on a monthly basis and in which all the members of the group participate to discuss various topics regarding the group and to be informed about several other activities.

(c) The "research-associates' meeting" that runs on a 2 to 3 weeks basis and makes the basic decisions concerning the scientific strategy of the group.

15 Cooperations

At an informal level we cooperate with researchers from the other research group (AG2) of the Max-Planck-Institut für Informatik and from the Computer Science Department of the Universität des Saarlandes. With AG2 we cooperate on two problems: specification of abstract data types and identification of partial orders. In the computer science department our main contacts are Prof. Hotz (in VLSI–design), Prof. Paul (in parallel algorithms), Prof. Seidel (in computational geometry) and Prof. Wilhelm (in parallel programming languages).

At an institutional level we are involved in several research projects with various sources of funds. The projects are funded by the European Union (ALCOM-IT, ALTEC, CGAL), the German government through DFG and BMBF, and by Bilateral Agreements. The details are given in the rest of this section.

15.1 Projects funded by the European Union

15.1.1 ALCOM-IT

ALCOM-IT (ALgorithms and COMplexity in Information Technology) is an ESPRIT IV Long Term Research project involving 12 partners from 9 different countries. ALCOM-IT and CGAL are off-shoots of the ALCOM I and II projects. ALCOM-IT differs form its predecessors in that there is a stronger focus and greater emphasis on the applied part of research. The aim is to bridge the gap between research in the field of algorithms and applications in information technology, and to play for European industry the role that the strong algorithm groups within IBM Research and Bell Labs play for their companies.

The project runs from January 1, 1996 to December 31, 1998 and is coordinated by Prof. Giorgio Ausiello. Naveen Garg is our local contact person.

The ALCOM-IT project brings together researchers from major European institutions. The partners and group leaders are:

University of Århus, Århus, Denmark (Prof. E.M. Schmidt)
Universitat Politecnica Catalunya, Barcelona, Spain (Prof. J. Diaz)
Universität zu Köln, Köln, Germany (Prof. M. Jünger)
EHESS, Paris, France (Directeur P. Rosenstiehl)
INRIA-Paris, Rocquencourt, France (Dr. P. Flajolet)
Universität-GH Paderborn, Paderborn, Germany (Prof. B. Monien)
Computer Technology Institute, Patras, Greece (Prof. P. Spirakis)
Universitá di Roma "La Sapienza", Roma, Italy (Prof. G. Ausiello)
Utrecht University, Utrecht, The Nederlands (Prof. J. van Leeuwen)
University of Warwick, Coventry, United Kingdom (Prof. M. Paterson)
ETH Zürich, Zürich, Switzerland (Prof. G. Gonnet)
Max-Planck-Institut für Informatik, Saarbrücken, Germany (Prof. K. Mehlhorn)

15.1.2 ALTEC

The ALTEC (ALgorithms for future TECnologies) project is financed by the EU, and is intended to join the expertise of a number of Western and Central/Eastern European research groups establishing a network of research and cooperation and to carry out a cooperative program of coordinated research aimed at the further development of algorithm-based techniques for future information processing technologies. The project is a pilot for future projects with Western and Central/Eastern European partners.

The first period of the project (Cooperative Action IC 1000) has expired on March 1996. In the fall of 1996, we have entered a second period. This is a so-called "Keep In Touch" project. It builds further on the achievements of ALTEC: its main objective is to support the existence of the ALTEC co-operative network. On the technical level, the goal is to stimulate joint research and cooperation in the areas of parallel architectures, parallel algorithms and their implementation, and the underlying communication networks of parallel computers. More generally, it stimulates research on the effective exploitation of novel programming techniques and programming support environments in the area of high performance computing and networking (HPCN). This should lead to the development of HPCN, particularly also in the Central European countries. Jop Sibeyn is our local contact person.

The next yearly meeting of ALTEC will be organized from May 21st until May 24th at the Max-Planck-Institut für Informatik. The ALTEC web-page is maintained by our institute, and

provides more information about the project. It can be found at
`http://www.mpi-sb.mpg.de/~jopsi/altec.html`.

15.1.3 CGAL

CGAL (Constructing a Geometric Algorithms Library) is an ESPRIT IV Long Term Research project. It is a joint effort of seven sites in six (mainly) European countries and has its roots in the computational geometry groups of the ALCOM projects. Inspired by the success of LEDA, the goal of the project is to design and implement robust, easy to use, efficient geometric software and libraries of geometric algorithms in C++. The geometric software is developed in close collaboration with 10 companies in Europe. Via the Saarbrücken site, Prof. R. Seidel and his group at the Universität des Saarlandes are contributing to the project. The project started at October 1996 and will end in March 1998. The project is coordinated by Utrecht University; our local contact persons are Kurt Mehlhorn and Stefan Schirra.

Partners and group leaders of the project are:

Utrecht University, Utrecht, The Netherlands (Prof. M. Overmars, Dr. M. de Berg, Dr. R. Veltkamp)
ETH, Zürich, Switzerland, (Prof. E. Welzl, Prof. P. Widmayer, Prof. J. Nievergelt)
Freie Universität Berlin, Germany (Prof. H. Alt)
INRIA Sophia-Antipolis, Sophia-Antipolis, France (Dr. J.-D. Boissonnat, Dr. A. Fabri)
RISC, Linz, Austria (Dr. S. Stifter)
Tel Aviv University, Israel (Prof. D. Halperin, Prof. M. Sharir)
Max-Planck-Institut für Informatik, Saarbrücken, Germany (Prof. K. Mehlhorn, Dr. S. Schirra)

15.2 Projects funded by DFG

Three projects are funded by the German National Science Foundation (DFG – Deutsche Forschungsgemeinschaft). The first project is supported by the special research area program (SFB – Sonderforschungsbereich) of DFG, while the other two are supported by a special program of DFG (DFG-Schwerpunktprogramm).

15.2.1 SFB 124 VLSI–Entwurfsmethoden und Parallelität

The SFB 124 is a special research effort on VLSI design methods and parallelism. The part of the project in which our group has been involved has been centered around more practical aspects of parallel systems: design, programming (language and implementation aspects) and applications. The SFB 124 was initiated in 1983, and will end by December 1997.

Currently, two members of our group, Jop F. Sibeyn (problems and algorithms for interconnection networks) and Jesper L. Träff (construction of the PAD library of PRAM algorithms and data structures), are sponsored by SFB 124.

For the last 3 year period of the project the partners are as follows:

University of Kaiserslautern (Prof. Zimmermann, Dr. Schuermann)
Universität des Saarlandes (Professors: Buchmann, Hotz, Loeckx, Paul, Wilhelm; Dr. Ruenger)
Max-Planck-Institut für Informatik (Prof. Mehlhorn)

15.2.2 Molecular Dynamics Simulations of Synthetic Polymers

The project is part of the DFG-Schwerpunktprogramm "Efficient Algorithms for Discrete Problems and their Applications". The goal of the project is the development and implementation of efficient parallel algorithms for MD-simulations of synthetic polymers. It started in 1995 and funds one research assistant position for three years.

Partners and group leaders of the project are:

Max-Planck-Institut für Polymerforschung, Mainz, (Dr. B. Jung)
Max-Planck-Institut für Informatik, Saarbrücken, (Dr. C. Rüb, Dr. H.P. Lenhof)

15.2.3 Graph Drawing

The project "Design, Analysis, Implementation, and Evaluation of New Algorithms for Graph Drawing" is also part of the DFG-Schwerpunktprogramm "Efficient Algorithms for Discrete Problems and their Applications". The aim of this project is to develop new techniques for drawing graphs. Moreover, we are designing a software library (AGD library), which will be independent of the graph editor used (e.g., GraphWin or Graphlet) and will contain modules of graph drawing algorithms. Among others, our software library will be distributed as part of the new LEDA Extension Packages. Our local contact person is Petra Mutzel.

The project started in 1995, and will be running until the end of 1997. Hopefully, we will get a prolongation for three more years this fall.

Partners and group leaders of the project are:

Universität Halle (Prof. S. Näher)
Universität Köln (Prof. M. Jünger)
Universität Passau (Prof. F. Brandenburg)
Max-Planck-Institut für Informatik (Prof. K. Mehlhorn)

15.3 Projects funded by BMBF

The German Ministry of Education and Research (BMBF), in the frame of the program "Mathematical Methods for Problem Solving in Trade and Industry", funds the project "Labeling in Graph Drawing". In this project we will investigate the (map) labelling problem in connection with the drawing problem, since for the drawing of the finite state machines for SIEMENS, the relatively big labels of the states and the state transitions lead to serious problems. The project starts at July 1997 and funds one research associate position for a duration of three years. Our industrial partner in this project is SIEMENS AG. The project leader is Petra Mutzel.

15.4 Projects funded by Bilateral Agreements

15.4.1 Joint project with Tel-Aviv University, Israel

The German-Israeli Foundation for Scientific Research and Development (GIF) is a bi-national science foundation. It was created by the two governments in order to promote and fund joint civil research and development projects in basic and applied research.

We have been successfully working with our Israeli partner Micha Sharir from Tel Aviv University and our German partner Raimund Seidel from the Universität des Saarlandes. Emo Welzl gave up his position within the foundation when he moved to ETH Zürich.

In the GIF II project we study various basic problems in computational and discrete geometry involving arrangements of curves and surfaces, and investigating the use of randomized techniques to solve many of these problems.

The GIF I project, *Randomized Techniques and Related Studies of Arrangements in Computational Geometry*, ran from 1-10-1990 until 30-9-1994, (including an extension of one year without additional funding). The GIF II project, *Study of Arrangements and Randomized Techniques in Discrete and Computational Geometry*, spans the period from 1-1-1995 until 31-12-97. Our local contact person is Michael Seel.

Partners and group leaders of the project are:

Universität des Saarlandes, Saarbrücken, Germany (Prof. R. Seidel)
Tel-Aviv University, Tel-Aviv, Israel (Prof. M. Sharir)
Max-Planck-Institut für Informatik, Saarbrücken, Germany (Prof. K. Mehlhorn)

15.4.2 Joint project with IIT Madras, India

In 1974 the governments of India and Germany concluded an “Agreement on Cooperation in the Field of Scientific Research and Technological Development”. Subsequently, special agreements were concluded between research organizations and institutes in the two countries. Our cooperation project with the Indian Institute of Technology Madras entitled *Studies in Computational Geometry and Relevant Data Structures* was implemented by a special agreement between the Kernforschungsanlage Jülich (KFA) in Germany, and the Department of Education (DoEd) and the Department of Science and Technology (DST) in India. The project provided funding for the exchange of scientists between MPI and IIT Madras for the period from June 1993 til March 1996. Our Indian partners were Kamala Krithivasan and C. Pandu Rangan; our local contact person was Stefan Schirra.

Partners and group leaders of the project are:

Indian Institute of Technology Madras, Madras, India (Prof. K. Krithivasan)
Max-Planck-Institut für Informatik, Saarbrücken, Germany (Prof. K. Mehlhorn)

15.5 Cooperations with Industry

We are cooperating with SIEMENS on a project on automatic drawing of finite state machines that describe the control of computer integrated manufacturing processes. Here, we show that our graph drawing techniques based on planarization methods are indeed practical. The SIEMENS AG is sponsoring a PhD fellowship to us for two to three years. The project started in 1996, and the project leader is Petra Mutzel. Our contact person at SIEMENS is Dr. U. Lauther.

For the LEDA project the main industrial partner is LEDA Software GmbH. This partnership is “legalized” by a contract between the Max-Planck-Gesellschaft and LEDA GmbH which gives LEDA GmbH the right to market LEDA and regulates the royalties that have to be paid by LEDA GmbH in return.

16 Recent Publications

In Journals and Book Chapters

[1] S. Albers. A competitive analysis of list update algorithms with lookahead. *Theoretical Computer Science*, to appear 1997.

[2] S. Albers. Improved randomized on-line algorithms for the list update problem. *SIAM Journal on Computing*, to appear.

[3] S. Albers. On the influence of lookahead in competitive paging algorithms. *Algorithmica*, to appear 1997.

[4] S. Albers and T. Hagerup. Improved parallel integer sorting without concurrent writing. *Information and Computation*, to appear.

[5] S. Albers and M. Mitzenmacher. Average case analyses of list update algorithms, with applications to data compression. *Algorithmica*, to appear.

[6] S. Albers, B. von Stengel, and R. Werchner. A combined BIT and TIMESTAMP algorithm for the list update problem. *Information Processing Letters*, 56:135–139, 1995.

[7] S. Albers and J. Westbrook. A survey of self-organizing data structures. In A. Fiat and G. Woeginger, editors, *Online Algorithms*. Springer Verlag, 1997. To appear.

[8] N. Alon, P.G. Bradford, and R. Fleischer. Matching nuts and bolts faster. *Information Processing Letters*, 59:123–127, 1996.

[9] H. Alt, L. Guibas, K. Mehlhorn, and R. Karp. A method for obtaining randomized algorithms with small tail probabilities. *Algorithmica*, 16(4/5):543–547, 1996.

[10] S. Arikati and A. Maheshwari. Realizing degree sequences in parallel. *SIAM Journal on Discrete Mathematics*, 9(2):317–338, 1996.

[11] S. Arikati, A. Maheshwari, and C. Zaroliagis. Efficient computation of implicit representations of sparse graphs. *Discrete Applied Mathematics*, 1997, in print.

[12] S. Arya, D. M. Mount, and O. Narayan. Accounting for boundary effects in nearest neighbor searching. *Discrete and Computational Geometry*, 16(2):155–176, 1996.

[13] S. Arya and M. Smid. Efficient construction of a bounded degree spanner with low weight. *Algorithmica*, 17(1):33–54, 1997.

[14] H. Bast and T. Hagerup. Fast parallel space allocation, estimation, and integer sorting. *Information and Computation*, 123:72–110, 1995.

[15] Y. Ben-Asher, A. Schuster, and J.F. Sibeyn. Load balancing: A programmer's approach or the impact of task-length parameters on the load balancing performance of parallel programs. *International Journal on High-Speed Computing*, 7(2):303–326, 1995.

[16] H.L. Bodlaender and T. Hagerup. Parallel algorithms with optimal speedup for bounded treewidth. *SIAM Journal on Computing*, to appear.

[17] P.G. Bradford and V. Capoyleas. Weak ϵ-nets for points on a hypersphere. *Discrete and Computational Geometry*, to appear.

[18] P.G. Bradford, R. Fleischer, and M. Smid. More efficient parallel totally monotone matrix searching. *Journal of Algorithms*, to appear.

[19] G.S. Brodal, S. Chaudhuri, and J. Radhakrishnan. The randomized complexity of maintaining the minimum. *Nordic Journal of Computing*, 3(4):337–351, 1996. Special issue on SWAT'96.

[20] G. Călinescu, C. Fernandes, U. Finkler, and H. Karloff. A better approximation algorithm for finding planar subgraphs. *Journal of Algorithms*, to appear.

[21] S. Chaudhuri. Sensitive functions and approximate problems. *Information and Computation*, 126(2):161–168, 1996.

[22] S. Chaudhuri, Y. Dimopoulos, and C. Zaroliagis. On the parallel complexity of acyclic logic programs. *Parallel Processing Letters*, 6(2):223–230, 1996.

[23] S. Chaudhuri and D. Dubhashi. Probabilistic recurrence relations revisited. *Theoretical Computer Science*, 1997, to appear.

[24] S. Chaudhuri and J. Radhakrishnan. The complexity of parallel prefix problems on small domains. *Information and Computation*, 1997, to appear.

[25] S. Chaudhuri and C. Zaroliagis. Shortest paths in digraphs of small treewidth. Part I: Sequential algorithms. *Algorithmica*, 1997, to appear. Special issue on Treewidth, Graph Minors, and Algorithms.

[26] S. Chaudhuri and C. Zaroliagis. Shortest paths in digraphs of small treewidth. Part II: Optimal parallel algorithms. *Theoretical Computer Science*, 1997, to appear. Special issue on ESA'95.

[27] J. Cheriyan and T. Hagerup. A randomized maximum-flow algorithm. *SIAM Journal on Computing*, 24:203–226, 1995.

[28] J. Cheriyan, T. Hagerup, and K. Mehlhorn. An $o(n^3)$-time maximim-flow algorithm. *SIAM Journal on Computing*, 25:1144–1170, 1996.

[29] J. Cheriyan and K. Mehlhorn. Algorithms for dense graphs and networks on the random access computer. *Algorithmica*, 15(5):521–549, 1996.

[30] A. Datta, H.P. Lenhof, C. Schwarz, and M. Smid. Static and dynamic algorithms for the k-point clustering problems. *Journal of Algorithms*, 19:474–503, 1995.

[31] C. De Simone, M. Diehl, M. Jünger, P. Mutzel, G. Reinelt, and G. Rinaldi. Exact ground states of Ising spin glasses: New experimental results with a branch and cut algorithm. *Journal of Statistical Physics*, 80:487–496, 1995.

[32] C. De Simone, M. Diehl, M. Jünger, P. Mutzel, G. Reinelt, and G. Rinaldi. Exact ground states of two-dimensional $\pm J$ Ising spin glasses. *Journal of Statistical Physics*, 84(5/6):1363–1371, 1996.

[33] O. Devillers, M.J. Golin, K. Kedem, and S. Schirra. Queries on Voronoi diagrams of moving points. *Computational Geometry: Theory and Applications*, 6:315–327, 1996.

[34] P. Dietz, K. Mehlhorn, R. Raman, and C. Uhrig. Lower bounds for set intersection queries. *Algorithmica*, 14(2):154–168, 1995.

[35] M. Dietzfelbinger, T. Hagerup, J. Katajainen, and M. Penttonen. A reliable randomized algorithm for the closest-pair problem. *Journal of Algorithms*, to appear.

[36] D. Dubhashi, G. Pantziou, P. Spirakis, and C. Zaroliagis. The fourth moment in Luby's distribution. *Theoretical Computer Science*, 148(1):133–140, 1995.

[37] P. Eades and P. Mutzel. *Graph Drawing Algorithms*, CRC Handbook of Algorithms and Theory of Computation, Chapter 23, M. Atallah (Ed.). CRC Press, 1997. To appear.

[38] R. Fleischer. A simple balanced search tree with $O(1)$ worst-case update time. *International Journal of Foundations of Computer Science*, 7(2):137–149, 1996.

[39] R. Fleischer, H. Jung, and K. Mehlhorn. A communication-randomness tradeoff for two-processor systems. *Information and Computation*, 116(2):155–161, 1995.

[40] N. Garg and V. Vazirani. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM Journal on Computing*, 25(2):235–251, 1996.

[41] M.T. Goodrich and E.A. Ramos. Bounded-independence derandomization of geometric partitioning with applications to parallel fixed-dimensional linear programming. *Discrete and Computational Geometry*, 1997, to appear.

[42] P. Gupta, R. Janardan, and M. Smid. Further results on generalized intersection searching problems: counting, reporting, and dynamization. *Journal of Algorithms*, 19(2):282–317, 1995.

[43] P. Gupta, R. Janardan, and M. Smid. Algorithms for generalized halfspace range searching and other intersection searching problems. *Computational Geometry: Theory and Applications*, 5:321–340, 1996.

[44] P. Gupta, R. Janardan, and M. Smid. Fast algorithms for collision and proximity problems involving moving geometric objects. *Computational Geometry: Theory and Applications*, 6:371–391, 1996.

[45] T. Hagerup. Fast deterministic processor allocation. *Journal of Algorithms*, 18:629–649, 1995.

[46] T. Hagerup. A lower bound for the emulation of PRAM memories on processor networks. *Information and Computation*, 119:124–128, 1995.

[47] T. Hagerup, J. Katajainen, N. Nishimura, and P. Ragde. Characterizations of k-terminal flow networks and computing network flows in partial k-trees. *Journal of Computer and System Sciences*, to appear.

[48] T. Hagerup and J. Keller. Fast parallel permutation algorithms. *Parallel Processing Letters*, 5:139–148, 1995.

[49] T. Hagerup and M. Kutyłowski. Fast integer merging on the EREW PRAM. *Algorithmica*, 17:55–66, 1997.

[50] B. Jung, H.P. Lenhof, P. Müller, and C. Rüb. Langevin dynamics simulations of macromolecules on parallel computers. *Macromolecular Theory and Simulation*, 6:507–521, 1997.

[51] M. Jünger and P. Mutzel. Exact and heuristic algorithms for 2-layer straightline crossing minimization. *Journal of Graph Algorithms and Applications*, to appear.

[52] M. Jünger and P. Mutzel. Maximum planar subgraphs and nice embeddings: Practical layout tools. *Algorithmica*, 16(1):33–59, 1996. Special Issue on Graph Drawing.

[53] M. Jünger, P. Mutzel, T. Odenthal, and M. Scharbrodt. The thickness of a minor-excluded class of graphs. *Discrete Mathematics*, to appear.

[54] D. Kagaris, G. Pantziou, S. Tragoudas, and C. Zaroliagis. Transmissions in networks with capacities and delays. *Networks*, 1997, to appear.

[55] S. Kapoor and M. Smid. New techniques for exact and approximate dynamic closest-point problems. *SIAM Journal on Computing*, 25(4):775–796, 1996.

[56] M. Kaufmann, U. Meyer, and J.F. Sibeyn. Matrix transpose on meshes: Theory and practice. *Computers and Artificial Intelligence*, to appear.

[57] M. Kaufmann, H. Schröder, and J.F. Sibeyn. Asymptotically optimal and practical routing on the reconfigurable mesh. *Parallel Processing Letters*, 5(1):81–95, 1995.

[58] M. Kaufmann and J.F. Sibeyn. Randomized multipacket routing and sorting on meshes. *Algorithmica*, 17:224–244, 1997.

[59] D. Kavvadias, G. Pantziou, P. Spirakis, and C. Zaroliagis. Hammock-on-ears decomposition: A technique for the efficient parallel solution of shortest paths and other problems. *Theoretical Computer Science*, 168(1):121–154, 1996. Special issue on MFCS'94.

[60] L. Kirousis, P. Spirakis, and Ph. Tsigas. Simple atomic snapshots: A linear complexity solution with unbounded time-stamps. *Information Processing Letters*, 58:47–53, 1996.

[61] H.P. Lenhof and M. Smid. Sequential and parallel algorithms for the k closest pairs problem. *International Journal of Computational Geometry & Applications*, 5(3):273–288, 1995.

[62] K. Mehlhorn and P. Mutzel. On the embedding phase of the Hopcroft and Tarjan planarity testing algorithm. *Algorithmica*, 16(2):233–242, 1996.

[63] K. Mehlhorn and V. Priebe. On the all-pairs shortest-path algorithm of Moffat and Takaoka. *Random Structures & Algorithms*, 10:205–220, 1997.

[64] K. Mehlhorn, R. Sundar, and C. Uhrig. Maintaining dynamic sequences under equality tests in polylogarithmic time. *Algorithmica*, 17(2):183–198, 1997.

[65] M. Müller, C. Rüb, and W. Rülling. A circuit for exact summation of floating-point numbers. *Information Processing Letters*, 57(3):159–163, 1996.

[66] P. Mutzel, T. Odenthal, and M. Scharbrodt. The thickness of graphs: A survey. *Graphs and Combinatorics*, to appear.

[67] C. Rüb. On the average running time of odd–even merge sort. *Journal of Algorithms*, 22(2):329–346, 1997.

[68] J.F. Sibeyn, B.S. Chlebus, and M. Kaufmann. Deterministic permutation routing on meshes. *Journal of Algorithms*, 22:111–141, 1997.

[69] M. Smid. Closest point problems in computational geometry. In J.-R. Sack and J. Urrutia, editors, *Handbook on Computational Geometry.* North Holland, 1997. To appear.

[70] M. Vingron, H.-P. Lenhof, and P. Mutzel. Computational molecular biology. To appear in: *Annotated Bibliographies in Combinatorial Optimization.* Eds. M. Dell'Amico, F. Maffioli and S. Martello.

[71] C. Zaroliagis. Simple and work-efficient parallel algorithms for the minimum spanning tree problem. *Parallel Processing Letters*, 1997. In print.

In Conference Proceedings

[1] S. Albers. Better bounds for online scheduling. In *Proceedings 29th Annual ACM Symposium on Theory of Computing (STOC)*, 1997.

[2] S. Albers and M. Henzinger. Exploring unknown environments. In *Proceedings 29th Annual ACM Symposium on Theory of Computing (STOC)*, 1997.

[3] S. Albers and H. Koga. Page migration with limited local memory capacity. In *Proceedings 4th International Workshop on Algorithms and Data structures (WADS), Springer Lecture Notes in Computer Science, Volume 955*, pages 514–525, 1995.

[4] S. Albers and M. Mitzenmacher. Average case analyses of list update algorithms, with applications to data compression. In *Proceedings 23rd International Colloquium on Automata, Languages and Programming (ICALP), Springer Lecture Notes in Computer Science, Volume 1099*, pages 147–158, 1996.

[5] A. Andersson, T. Hagerup, S. Nilsson, and R. Raman. Sorting in linear time? In *Proc. 27th Annual ACM Symposium on Theory of Computing (STOC)*, pages 427–436, 1995.

[6] S. Arikati, S. Chaudhuri, and C. Zaroliagis. All-pairs min-cut in sparse networks. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS'95)*, pages 363–376. Lect. Notes in Comp. Sc. 1026, Springer-Verlag, 1995.

[7] S. Arikati, D.Z. Chen, L.P. Chew, G. Das, M. Smid, and C. Zaroliagis. Planar spanners and approximate shortest path queries among obstacles in the plane. In *Algorithms (ESA'96)*, pages 514–528. Lect. Notes in Comp. Sc. 1136, Springer-Verlag, 1996.

[8] S. Arikati, A. Dessmark, A. Lingas, and M.V. Marathe. Approximation algorithms for maximum two-dimensional pattern matching. In *Proc. Combinatorial Pattern Matching, Lecture Notes in Computer Science, Vol. 1075*, pages 348–360, 1996.

[9] S. Arya and D. M. Mount. Approximate range searching. In *Proc. 11th Ann. ACM Symp. Comput. Geom.*, pages 172–181, June 1995.

[10] S. Arya, D. M. Mount, and O. Narayan. Accounting for boundary effects in nearest neighbor searching. In *Proc. 11th Ann. ACM Symp. Comput. Geom.*, pages 336–344, 1995.

[11] R. Bacik and S. Mahajan. Semidefinite programming and its applications to np problems. In *Computing and Combinatorics (COCOON'95)*, volume 959 of *Lecture Notes in Computer Science*, pages 566–575. Springer, 1997.

[12] P. Berthome, T. Duboux, T. Hagerup, I. Newman, and A. Schuster. Self-simulation for the passive optical star model. In *Algorithms-ESA'95*, volume 979 of *Lecture Notes in Computer Science*, pages 268–380. Springer, 1995.

[13] H.L. Bodlaender and T. Hagerup. Parallel algorithms with optimal speedup for bounded treewidth. In *Proc. 22nd International Colloquium on Automata, Languages and Programming (ICALP), Lecture Notes in Computer Science, Vol. 944*, pages 268–279. Springer-Verlag, 1995.

[14] P.G. Bradford and R. Fleischer. Matching nuts and bolts faster. In *Proceedings of the 6th International Symposium on Algorithms and Computation (ISAAC'95), Springer Lecture Notes in Computer Science 1004*, pages 402–408, 1995.

[15] P.G. Bradford and K. Reinert. Lower bounds for row minima searching. In *Proc. 23rd International Colloqium on Automata, Languages and Programming (ICALP'96), Springer Lecture Notes in Computer Science 1099*, pages 454–465, 1996.

[16] G.S. Brodal. Priority queues on parallel machines. In *Proc. 5th Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 1097 of *Lecture Notes in Computer Science*, pages 416–427. Springer Verlag, Berlin, 1996.

[17] G.S. Brodal and L. Gąsieniec. Approximate dictionary queries. In *Proc. 7th Symposium on Combinatorial Pattern Matching (CPM'96)*, pages 65–74, 1996.

[18] G.S. Brodal, J.L. Träff, and C. Zaroliagis. A parallel priority data structure with applications. In *Proceedings of the 11th IEEE International Parallel Processing Symposium (IPPS'97)*, pages 689–693, 1997.

[19] C. Burnikel, R. Fleischer, K. Mehlhorn, and S. Schirra. A Strong and Easily Computable Separation Bound for Arithmetic Expressions Involving Square Roots. In *Proc. 8th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 702–709, 1997.

[20] G. Călinescu, C. Fernandes, U. Finkler, and H. Karloff. A better approximation algorithm for finding planar subgraphs. In *Proc. 7th ACM-SIAM Symposium on Discrete Algorithms*, pages 16–25, 1996.

[21] S. Chaudhuri and J. Radhakrishnan. Deterministic restrictions in circuit complexity. In *Proc. 28th ACM Symp. on Theory of Computing (STOC)*, pages 30–37, 1996.

[22] S. Chaudhuri and C. Zaroliagis. Optimal parallel shortest paths in small treewidth digraphs. In *Algorithms (ESA'95)*, pages 31–45. Lect. Notes in Comp. Sc. 979, Springer-Verlag, 1995.

[23] S. Chaudhuri and C. Zaroliagis. Shortest path queries in digraphs of small treewidth. In *Automata, Languages and Programming (ICALP'95)*, pages 244–255. Lect. Notes in Comp. Sc. 944, Springer-Verlag, 1995.

[24] D.Z. Chen, G. Das, and M. Smid. Lower bounds for computing geometric spanners and approximate shortest paths. In *Proc. 8th Canad. Conf. Comput. Geom.*, pages 155–160. Carleton University Press, Ottawa, Canada, 1996.

[25] B.S. Chlebus, A. Czumaj, L. Gąsieniec, M. Kowaluk, and W. Plandowski. Parallel alternating-direction access machine. In *Proceedings 21th International Symposium on Mathematical Foundations of Computer Science (MFCS'96)*, pages 255–266, 1996.

[26] K.W. Chong. Finding minimum spanning trees on the EREW PRAM. In *Proc. Int'l Computer Symp, Taiwan*, pages 7–14, 1996.

[27] K.W. Chong and T.W. Lam. Towards more precise parallel biconnectivity approximation. In *Proc. 7th Annual International Symposium on Algorithms and Computation*, pages 223–232, 1996.

[28] T. Christof, M. Jünger, J. Kececioglu, P. Mutzel, and G. Reinelt. A branch-and-cut approach to physical mapping with end-probes. In *Proc. 1st Intern. Conf. on Computational Molecular Biology (RECOMB'97)*, pages 84–92, 1997.

[29] R. Cole and R. Hariharan. An $O(n \log n)$ algorithm for the maximum agreement subtree problem for binary trees. In *Proc. 4th ACM-SIAM Symposium on Discrete Algorithms*, pages 323–332, 1996.

[30] C. Cooper, A. Frieze, K. Mehlhorn, and V. Priebe. On the average-case complexity of shortest-paths problems in the vertex-potential model. *Proc. 1st International Symposium on Randomization and Approximation Techniques in Computer Science*, 1997, to appear.

[31] M. Crochemore, L. Gąsieniec, and W. Rytter. Constant-space string-matching in sublinear average time. In *Proc. Compression and Complexity of SEQUENCES 1997*, 1997. To appear.

[32] G. Das, R. Fleischer, L. Gąsieniec, D. Gunopulos, and J. Karkkainen. Episode matching. In *Proceedings 8th Symposium on Combinatorial Pattern Matching (CPM'97)*, 1997. To appear.

[33] G. Das, S. Kapoor, and M. Smid. On the complexity of approximating Euclidean traveling salesman tours and minimum spanning trees. In *Proc. 16th Conf. Foundations of Software Technology and Theoretical Comput. Sci.*, volume 1180 of *Lecture Notes in Computer Science*, pages 64–75. Springer-Verlag, 1996.

[34] M. de Berg, M. van Kreveld, and S. Schirra. A new approach to subdivision simplification. In *Proceedings ACSM/ASPRS Annual Convention, volume 4: Auto-Carto 12*, pages 79–88, 1995.

[35] J. Dedorath, J. Gergov, and T. Hagerup. More efficient parallel flow algorithms. In *Proc. 6th International Symposium on Algorithms and Computation*, pages 234–243, 1995.

[36] K. Diks and T. Hagerup. More general parallel tree contraction: Register allocation and broadcasting in a tree. In *Proc. 22nd International Workshop on Graph-Theoretic Concepts in Computer Science (WG) Lecture Notes in Computer Science, Vol. 1197*, pages 126–140, 1996.

[37] H. Djidjev, G. Pantziou, and C. Zaroliagis. Fast algorithms for maintaining shortest paths in outerplanar and planar digraphs. In *Fundamentals of Computation Theory (FCT'95)*, pages 191–200. Lect. Notes in Comp. Sc. 965, Springer-Verlag, 1995.

[38] H. Djidjev, G. Pantziou, and C. Zaroliagis. On-line and dynamic algorithms for shortest path problems. In *Theoretical Aspects of Computer Science (STACS'95)*, pages 193–204. Lect. Notes in Comp. Sc. 900, Springer-Verlag, 1995.

[39] D. Dobkin and D. Gunopulos. Geometric problems in machine learning. In Ming C. Lin and Dinesh Manocha, editors, *Applied Computational Geometry : Towards Geometric Engineering (WACG96)*, pages 121–132. Springer LNCS 1148, 1996.

[40] C.A. Duncan, M.T. Goodrich, and E.A. Ramos. Efficient approximation and optimization algorithms for computational metrology. In *Proceedings 8th ACM-SIAM Symposium on Discrete Algorithms (SODA'97)*, pages 121–130, 1997.

[41] A. Fabri, G.-J. Giezeman, L. Kettner, S. Schirra, and S. Schönherr. The CGAL kernel : a basis for geometric computation. In Ming C. Lin and Dinesh Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering (WACG96)*, pages 191–202. Springer LNCS 1148, 1996.

[42] U. Finkler and K. Mehlhorn. Runtime prediction of real programs on real machines. In *Proc. 8th ACM-SIAM Symposium on Discrete Algorithms (SODA'97)*, 1997.

[43] F. Follert, E. Schömer, J. Sellen, M. Smid, and C. Thiel. Computing a largest empty anchored cylinder, and related problems. In *Proc. 15th Conf. Foundations of Software Technology and Theoretical Comput. Sci.*, volume 1026 of *Lecture Notes in Computer Science*, pages 428–442. Springer-Verlag, 1995.

[44] N. Garg. A 3-approximation for the minimum tree spanning k vertices. In *Proc. IEEE Symposium on Foundations of Computer Science*, pages 302–309, 1996.

[45] N. Garg, M. Papatriantafilou, and Ph. Tsigas. Distributed list coloring: How to dynamically allocate frequencies to mobile base stations. In *Proceedings of the Eighth IEEE Symposium on Parallel and Distributed Processing*, pages 18–25, 1996.

[46] L. Gąsieniec, P. Indyk, and P. Krysta. External inverse pattern matching. In *Proceedings 8th Symposium on Combinatorial Pattern Matching (CPM'97)*, July 1997. To appear.

[47] L. Gąsieniec, M. Karpiński, W. Plandowski, and W. Rytter. Efficient algorithms for lempel-ziv encoding. In *Proceedings 5th Scandinavian Workshop on Algorithm Theory (SWAT'96)*, pages 392–403, 1996.

[48] L. Gąsieniec, M. Karpiński, W. Plandowski, and W. Rytter. Randomized efficient algorithms for compressed strings: the finger-print approach. In *Proceedings 7th Symposium on Combinatorial Pattern Matching (CPM'96)*, pages 39–49, June 1996.

[49] L. Gąsieniec, E. Kranakis, D. Krizanc, and A. Pelc. Minimizing congestion of layouts for ATM networks with faulty links. In *Proceedings 21th International Symposium on Mathematical Foundations of Computer Science (MFCS'96)*, pages 372–381, September 1996.

[50] J. Gergov. Approximation algorithms for dynamic storage allocation. In *Proc. Europ. Symp. Alg. (ESA'96)*, pages 52–61. Lect. Notes in Comp. Sc. 1136, Springer-Verlag, 1996.

[51] D. Gunopulos, H. Mannila, and S. Saluja. Discovering all most specific sentences by randomized algorithms. In *Proc. 6th Intern. Conf. on Database Theory (ICDT'97)*, pages 215–229. Lect. Notes in Comp. Sc. 1186, Srpinger-Verlag, 1997.

[52] A. Gupta, D. Kaller, S. Mahajan, and T. Shermer. Vertex partitioning problems on partial k-trees. In *Algorithm Theory - SWAT'96*, volume 1097 of *Lecture Notes in Computer Science*, pages 161–172. Springer, 1996.

[53] P. Gupta, R. Janardan, and M. Smid. Efficient algorithms for counting and reporting pairwise intersections between convex polygons. In *Proc. 8th Canad. Conf. Comput. Geom.*, pages 8–13. Carleton University Press, Ottawa, Canada, 1996.

[54] P. Gupta, R. Janardan, M. Smid, and B. Dasgupta. The rectangle enclosure and point-dominance problems revisited. In *Proc. 11th Ann. ACM Symp. Comput. Geom.*, pages 162–171, 1995.

[55] T. Hagerup. Allocating independent tasks to parallel processors: an experimental study. In *Proc. Parallel Algorithms for Irregularly Structured Problems (IRREGULAR'96)*, volume 1117 of *Lecture Notes in Computer Science*, pages 1–33, 1996.

[56] T. Hagerup. Dynamic algorithms for graphs of bounded treewidth. In *Proc. 24th International Colloquium on Automata, Languages and Programming (ICALP), Lecture Notes in Computer Science.* Springer-Verlag, 1997. To appear.

[57] R. Hariharan, P. Kelsen, and S. Mahajan. Coloring 2-colorable hypergraphs with a sublinear number of colors. In *Proc. Scandinavian Workshop on Algorithm Theory, Springer Lecture Notes in Computer Science, Volume 1097*, pages 41–52, 1996.

[58] R. Hariharan and S. Mahajan. Derandomizing semidefinite programming based approximation algorithms. In *Proc. 36th IEEE Symposium on Foundations of Computer Science*, pages 162–169, 1995.

[59] J-H. Hoepman, M. Papatriantafilou, and Ph. Tsigas. Self-stabilization in wait-free shared memory objects. In *Proceedings of the Ninth International Workshop on Distributed Algorithms*, pages 273–287, 1995.

[60] B. Jung, H.P. Lenhof, P. Müller, and C. Rüb. Parallel MD-simulations of synthetic polymers. In *Proc. 8th SIAM Conference on Parallel Processing for Scientific Computing*, 1997.

[61] M. Jünger and P. Mutzel. The polyhedral approach to the maximum planar subgraph problem: New chances for related problems. In R. Tamassia and I.G. Tollis, editors, *Graph Drawing (Proc. GD '94)*, volume 894 of *Lecture Notes in Computer Science*, pages 119–130. Springer-Verlag, 1995.

[62] M. Jünger and P. Mutzel. Exact and heuristic algorithms for 2-layer straightline crossing minimization. In *Graph Drawing (Proc. GD '95)*, volume 1027 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996.

[63] B. Juurlink, P.S. Rao, and J.F. Sibeyn. Worm-hole gossiping on meshes and tori. In *Proc. 2nd Euro-Par Conference*, volume 1123 of *Lecture Notes in Computer Science*, pages 361–369. Springer-Verlag, 1996.

[64] B. Juurlink and J.F. Sibeyn. Fast routing on weak hypercubes: Making the Valiant-Brebner algorithm more practical. In *Proc. 2nd Conference on High-Performance Computing*, pages 755–760. IEEE Press, 1995.

[65] D. Kagaris, G. Pantziou, S. Tragoudas, and C. Zaroliagis. On the computation of fast data transmissions in networks with capacities and delays. In *Algorithms and Data Structures (WADS'95)*, pages 291–302. Lect. Notes in Comp. Sc. 955, Springer-Verlag, 1995.

[66] J. Katajainen and J.L. Träff. A meticulous analysis of mergesort programs. In *Algorithms and Complexity. Third Italian Conference, CIAC'97*, volume 1203 of *LNCS*, pages 217–228, 1997.

[67] M. Kaufmann, U. Meyer, and J.F. Sibeyn. Matrix transpose on meshes: Theory and practice. In *Proc. 11th IEEE International Parallel Processing Symposium*, pages 315–319. IEEE, 1997.

[68] M. Kaufmann, J.F. Sibeyn, and T. Suel. Beyond the bisection bound: Fast ranking and counting on meshes. In *Proc. 3rd European Symposium on Algorithms*, volume 979 of *Lecture Notes in Computer Science*, pages 75–88, 1995.

[69] C.W. Keßler and J.L. Träff. A library of basic PRAM algorithms and its implementation in FORK. In *8th Annual ACM Symposium on Parallel Algorithms and Architechtures (SPAA)*, pages 193–195, 1996.

[70] C.W. Keßler and J.L. Träff. Language and library support for practical PRAM programming. In *5th Euromicro Workshop on Parallel and Distributed Processing (PDP'97)*, pages 216–221, 1997.

[71] K. Krithivasan, A. Rema, S. Schirra, and P.I. Vijaykumar. Velocity planning for a robot moving along the shortest straight line path among moving obstacles. In *Proc. of 8th Canadian Conference on Computational Geometry*, pages 149–154, 1996.

[72] H.P. Lenhof. An algorithm for the protein docking problem. In D. Schomburg and U. Lessel, editors, *Bioinformatics: From nucleic acids and proteins to cell metabolism: GBF Monographs Volume 18*, pages 125–139, 1995.

[73] H.P. Lenhof. New contact measures for the protein docking problem. In *Proc. 1st Annual International Conference on Computational Molecular Biology (RECOMB'97)*, pages 182–191, 1997.

[74] M. Mavronicolas, M. Papatriantafilou, and Ph. Tsigas. The impact of timing on linearizability in counting networks. In *Proc. 11th IEEE International Parallel Processing Symposium*, pages 684–688, 1997.

[75] K. Mehlhorn, M. Müller, S. Näher, S. Schirra, M. Seel, C. Uhrig, and J. Ziegler. Higher-dimensional computational geometry and applications. In *Proceedings of 13th Annual ACM Symp. on Computational Geometry*, 1997. (To appear).

[76] K. Mehlhorn, S. Näher, T. Schilz, S. Schirra, M. Seel, R. Seidel, and C. Uhrig. Checking Geometric Programs or Verification of Geometric Structures. In *Proc. of the 12th Annual Symposium on Computational Geometry*, pages 159–165, 1996.

[77] K. Mehlhorn, T.C. Shermer, and C.K. Yap. A complete roundness classification procedure. In *Proceedings 13th ACM Symposium on Computational Geometry (SoCG'97)*, June 1997. To appear.

[78] U. Meyer and J.F. Sibeyn. Simulating the simulator: Deterministic PRAM simulation on a mesh simulator. In F. Breitenecker and I. Husinsky (Eds), editors, *Proc. Eurosim'95*, pages 285–290, 1995.

[79] P. Mutzel. A polyhedral approach to planar augmentation and related problems. In *Algorithms – ESA '95, Third Annual European Symposium*, volume 979 of *Lecture Notes in Computer Science*, pages 494–507. Springer-Verlag, 1995.

[80] P. Mutzel. An alternative approach for drawing hierarchical graphs. In *Graph Drawing (Proc. GD '96)*, Lecture Notes in Computer Science. Springer-Verlag, 1997. To appear.

[81] M. Papatriantafilou and Ph. Tsigas. Wait-free consensus in in-phase multiprocessor systems. In *Proc. 7th IEEE Symposium on Parallel and Distributed Processing*, pages 312–319, 1995.

[82] A.J. Preetham, K. Krithivasan, C.P. Rangan, and J.F. Sibeyn. Parallel algorithm for maximum empty L-shaped polygon. In *Proc. 3rd Conference on High-Performance Computing.* IEEE, 1996.

[83] E.A. Ramos. Construction of 1-d lower envelopes and applications. In *Proceedings 13th ACM Symposium on Computational Geometry (SoCG'97)*, June 1997. To appear.

[84] K. Reinert, H.P. Lenhof, P. Mutzel, K.Mehlhorn, and J. Kececioglu. A branch-and-cut algorithm for multiple sequence alignment. In *Proc. 1st Annual International Conference on Computational Molecular Biology (RECOMB'97)*, pages 241–250, 1997.

[85] Ch. Rüb. On the average running time of odd-even merge sort. In *12th Annual Symposium on Theoretical Aspects of Computer Science (STACS'95)*, volume 900 of *LNCS*, pages 491–502, Munich, Germany, 1995. Springer.

[86] E. Schömer and C. Thiel. Efficient collision detection for moving polyhedra. In *Proc. 11th Ann. ACM Symp. Comput. Geom.*, pages 51–60, 1995.

[87] J. Schwerdt, M. Smid, and S. Schirra. Computing the minimum diameter for moving points: an exact implementation using parametric search. In *Proc. 13th Ann. ACM Sympos. Comput. Geom.*, 1997. To appear.

[88] J.F. Sibeyn. Sample sort on meshes. In *Proc. Computing Science in the Netherlands*, pages 217–228, 1995.

[89] J.F. Sibeyn. List ranking on interconnection networks. In *Proc. 2nd Euro-Par Conference*, volume 1123 of *Lecture Notes in Computer Science*, pages 799–808. Springer-Verlag, 1996.

[90] J.F. Sibeyn. Routing on triangles, tori and honeycombs. In *Proc. 21st Symposium on the Mathematical Foundations of Computer Science*, volume 1113 of *Lecture Notes in Computer Science*, pages 529–541. Springer-Verlag, 1996.

[91] J.F. Sibeyn. Better trade-offs for parallel list ranking. In *Proc. 9th Symposium on Parallel Algorithms and Architectures.* ACM, 1997. To appear.

[92] J.F. Sibeyn and M. Kaufmann. BSP-like external-memory computation. In *Proc. 3rd Italian Conference on Algorithms and Complexity*, volume 1203 of *Lecture Notes in Computer Science*, pages 229–240. Springer-Verlag, 1997.

[93] M. Smid and R. Janardan. On the width and roundness of a set of points in the plane. In *Proc. 7th Canad. Conf. Comput. Geom.*, pages 193–198, 1995.

[94] J.L. Träff and C. Zaroliagis. A simple parallel algorithm for the single-source shortest path problem on planar digraphs. In *Parallel Algorithms for Irregularly Structured Problems (IRREGULAR'96)*, volume 1117 of *Lecture Notes in Computer Science*, pages 183–194, 1996.

Part III

The Programming Logics Group

1 Personnel

Director:
Harald Ganzinger

Researchers:
Peter Barth (–February 1996)
David Basin (–February 1997; now (C4) professor of software engineering at the University of Freiburg)
Alexander Bockmayr
Manfred Jaeger (–October 1996)
Seán Matthews
Andreas Nonnengart (–May 1996)
Hans Jürgen Ohlbach (–April 1996; now collaborating as a lecturer at Imperial College, London)
Andreas Podelski
Christoph Weidenbach
Emil Weydert

Post-doctoral fellows and long term guests:
Penny Anderson (–August 1996; previously at INRIA Sophia Antipolis)
Thomas Burch (March 1996–; previously at the Universität des Saarlandes)
Witold Charatonik (October 1995–; previously at the University of Wroclaw)
Florent Jaquemard (January 1996–; previously at the LRI, Orsay)
R.K. Krishna Rao Madala (May 1994-August 1994, August 1995–January 1997; previously at the Tata Institut, Bombay)
Peter Madden (–January 1996; previously at the University of Edinburgh)
Ralamboniaina Ramangalahy (–January 1996; previously at the EDF, Clamart)
Leon van der Torre (March 1997–; previously at the Erasmus University Rotterdam)
Sergei Vorobyov (previously at INRIA-Lorraine & CRIN)
Yongge Wang (October 1996-December 1996; previously University of Heidelberg)

PhD students:
Abdelwaheb Ayari (–February 1997)
Hubert Baumeister
Peter Graf (–January 1996)
Jörn Hopf
Ullrich Hustadt
Thomas Kasper
Christoph Meyer
Renate Schmidt
Georg Struth
Jürgen Stuber
Andreas Tönne (–March 1997)
Miroslava Tzakova
Uwe Waldmann
Luca Viganò
Frank Zartmann (–January 95)

Secretaries:
Ellen Fries
Christine Kiesel

2 Visitors

Since January 1995, the following researchers (71 in total) have visited, or are visiting, our group.

1995		
Dov Gabbay	01.07.91–01.07.95	Imperial College, London
Shahid Rahman	01.01.94–01.04.95	Universität des Saarlandes
Michael Maher	16.01.95–17.01.95	IBM at Yorktown Heights
Bernhard Steffen	16.01.95–17.01.95	Universität Passau
Tiziana Margaria	16.01.95–17.01.95	Universität Passau
Tim Fernando	07.02.95	Universität Stuttgart
Thomas Santen	25.02.95–26.02.95	GMD-First, Berlin
Christopher Lynch	06.03.95–08.03.95	INRIA Lorraine
Mark Wallace	07.03.95–08.03.95	Imperial Colleg, London
P.D.A. de Maine	24.04.95–26.04.95	Auburn University
Burkhart Wolff	02.05.95	Universität Bremen
Julian Richardson	07.05.95–13.05.95	University of Edinburgh
Rajeev Goré	09.05.95–12.05.95	Australian National University
Leo Bachmair	15.05.95–30.06.95	SUNY at Stony Brook
Andreas Herzig	15.05.95–15.07.95	IRIT at Toulouse
David Plaisted	15.05.95–21.08.95	University of North Carolina
Thorsten Altenkirch	23.05.95–24.05.95	Chalmers University of Technology
Christoph Kreitz	14.06.95	TH Darmstadt
Wolfgang Küchlin	16.06.95	Universität Tübingen
Wayne Snyder	21.06.95	Boston University
Hans Rott	13.07.95–15.07.95	Universität Konstanz
John Slaney	16.07.95–18.07.95	Australian National University
Murdoch Gabbay	17.07.95–30.06.95	Trinity College, Cambridge
Jürgen Avenhaus	21.07.95	Universität Kaiserslautern
Chris Brink	03.08.95–12.08.95	University of Cape Town
Deepak Kapur	07.08.95–01.09.95	SUNY Albany
Evan Tick	14.07.95–28.08.95	University of Oregon
Andrei Voronkov	10.08.95	Uppsala University
Lutz Priese	09.10.95–10.10.95	TH Darmstadt

1996

Francois Bourdoncle	09.01.96–11.01.96	E.N.S. des Mines de Paris
Detlef Plump	24.01.96–26.01.96	Universität Bremen
Nils Klarlund	07.02.96–09.02.96	AT&T Laboratories
Hans Rott	15.02.96–15.04.96	Universität Konstanz
Piotr Krysta	15.02.96–15.08.96	University of Wroclaw
Morten Biehl Christiansen	22.02.96–02.03.96	University of Arhus
Paliath Narendran	02.03.96–08.03.96	SUNY at Albany
Aart Middledorp	05.03.96–09.03.96	Tsukuba
Frank Pfenning	06.03.96–07.03.96	TH Darmstadt
Nachum Dershowitz	06.03.96–08.03.96	University at Urbana
Friedrich Otto	06.03.96–08.03.96	Universität Kassel
Ursula Martin	06.03.96–08.03.96	University of St. Andrews
Hubert Comon	06.03.96–08.03.96	LRI Orsay
Robert Nieuwenhuis	06.03.96–09.03.96	University de Cataluna
David Plaisted	06.03.96–11.03.96	University of North Carolina
Wayne Snyder	07.03.96–08.03.96	University of Boston
Pierre Lescanne	07.03.96–08.03.96	CRIN at Nancy
Tobias Nipkow	08.03.96	TU München
Jürgen Avenhaus	08.03.96	Universität Kaiserslautern
Andrew Ireland	12.03.96	University of Edinburgh
Maurice Nivat	20.03.96–22.03.96	Université Paris
Bertrand le Saec	24.04.96–27.04.96	Université Bordeaux II
Maarten de Rijke	05.05.96–10.05.96	University of Warwick
Jean-Raymond Abrial	27.05.96–29.05.96	Digilog, Aix-en-Provence
Rajeev Goré	31.05.96	Australian National University
Tomasz Zajac	02.06.96–29.08.96	University of Wroclaw
A. Lukaszewski	02.06.96–30.06.96	University of Wroclaw
Alain Griffault	10.06.96–12.06.96	Université Bordeaux
Frank Pfenning	04.07.96–05.07.96	CMU
Susanne Graf	10.07.96–11.07.96	IMAG Grenoble
Arnaud Venet	05.08.96–08.08.96	Ecole Polytechnique, Paris
Yde Venema	08.08.96	Free University of Amsterdam
I.V. Ramakrishnan	29.08.96–30.08.96	SUNY at Stony Brook
Maurice Nivat	18.09.96–21.09.96	Université Paris
Leon van der Torre	12.11.96–13.11.96	Erasmus University Rotterdam
Tim Fernando	12.11.96–13.11.96	Universität Stuttgart
Frank Morawitz	25.11.96	Universität Tübingen
Tom Cornell	25.11.96	Universität Tübingen

1997

Marco Schaerf	19.02.97–22.02.97	University of Rome
Tomasz Zajac	10.03.97–28.03.97	University of Wroclaw
Danièle Beauquier	10.03.97–31.03.97	Université Paris
Tom Henzinger	20.03.97–22.03.97	University at Berkeley

3 Executive Summary

The research unit ‘Programming Logics’ applies methods of mathematical logic to a variety of problems in computer science. Computation is deduction, a principle that is taken literally in the area of Logic Programming. Formal specifications of software and hardware are formulae in logical systems. Program development and verification is based on proving theorems about specifications and programs. Solving problems in specific application domains such as mathematical optimization or program analysis can be seen as deduction with respect to specific constraint structures such as the integers, reals or Herbrand terms.

Computation often means to simulate some model of the real world. While in logic programming and in program synthesis and verification one applies, to a large extent, the classical logics known from mathematics, simulation of the real world requires logics that allow to treat incomplete and changing knowledge and to reason about beliefs, wishes, knowledge, and the like, of their agents. In that context non-classical logics have to be designed, investigated and applied.

Our work is both theoretical and practical in nature. A large fraction of it is essentially concerned with searching for new and better methods for finding proofs with the support of a computer. As the practical worth of results in this area can often not be judged from the theory alone, we are engaged in various implementation projects in which we try to obtain experimental evidence of the practical potential of our results.

We briefly survey some of the highlights in our main research areas.

Automated theorem proving for predicate logic

Coordinators: Harald Ganzinger, Christoph Weidenbach

A first production version of the SPASS theorem prover, an implementation of superposition with automatically inferred soft types (sorts), has been completed by Weidenbach and Meyer. SPASS is unique in many ways. At the CADE’96 competition of ATP systems, it won in one of the system categories and scored second in another. SPASS solves about 50% of the problems of the TPTP problem library, a major benchmark in the field, automatically. According to the data available to us at present, this is the highest success score of any system that has run the benchmark. SPASS is also unique with respect to its performance on decidable fragments of first-order logic as they arise from the embedding of a large range of modal logics with the techniques of Ohlbach, Hustadt, Nonnengart, and Schmidt. SPASS outperforms, often by orders of magnitude, even systems that have been specifically designed for deciding these logics.

On the theoretical side, David Basin and Harald Ganzinger have found an interesting new application of saturation by ordered resolution. They have shown how to use this tool for an automated complexity analysis of (ground) entailment problems for Horn theories and/or general first-order theories. Basically it has been shown that if a presentation is order-local, that is, is saturated under ordered resolution, the complexity of the entailment problem can be bounded from above by a bound that depends on the ordering. Experiments with several non-trivial theories have been made with the Saturate system by which such automatic complexity bounds have been obtained. Basin and his students have integrated efficient decision procedures for order-local theories into Isabelle.

Higher-order logic and logical frameworks

Coordinators: David Basin and Seán Matthews

Under this general heading we continue to study the theory and applications of proof development systems that are intended to be interactive, rather than fully automatic.

David Basin, Seán Matthews and Luca Viganò have developed a general theory of labelled deductive systems for the analysis of non-standard systems such as modal or relevance logics, and their implementation in a standard $\Rightarrow$ framework such as Edinburgh LF or Isabelle. They have explored in detail the proof-theoretic properties of their approach, and its relationship to more traditional 'semantic embeddings'.

We have made both theoretical and practical progress in our investigation of how a theorem prover can be exploited to combine the verification and synthesis of hardware and software. Penny Anderson, David Basin and Stefan Friedrich have developed general tools which exploit higher order unification to solve logical 'equations' of the form $\Sigma \leftrightarrow P$ where Σ is a specification and P is a metavariable over, e.g., logic programs or electronic circuits. The idea being to use the theorem prover to develop a proof that P is 'meta' ground, i.e., a concrete object of the appropriate type. To complement this, David Basin and Seán Matthews have developed a theoretical analysis of the mechanism that is independent of the metatheoretic facilities of a theorem prover.

Modal logic and automated theorem proving

Coordinator: Hans Jürgen Ohlbach

Work in this area is based on the hypothesis that semantic embedding into first-order logic provides a uniform approach for automated theorem proving in modal logic. The basic idea is to use one efficient resolution prover such as SPASS as a uniform decision procedure for many logics. Saturation, if powerful redundancy elimination techniques are exploited, often terminates on consistent formulae. This is in contrast to the behavior of the backward reasoning approaches in automated theorem proving including semantic tableau and model elimination.

Renate Schmidt has shown that for a large class of these logics, by semantic embedding into a first-order fragment called path logic, any refutationally complete resolution theorem strategy will always be terminating, and hence become a decision procedure, provided condensement of clauses (preventing these from becoming unboundedly wide) is performed eaglerly. The significance of this result is that any resolution prover can be used as a decision procedure, and that no specific, 'intelligent' simplification is required.

Ullrich Hustadt has shown how to embed many modal logics into Maslov's class K, one of the largest known classes of decidable first-order formulas. He has also been able to identify, in detail, a resolution-based decision procedure for the class K. That solves a longstanding open problem for which in the previous literature only partial solutions can be found.

Logic and uncertainty

Coordinator: Emil Weydert

Manfred Jaeger and Emil Weydert have investigated probabilistic and quasi-probabilistic approaches to nonmonotonic inference and default reasoning and have started to look at applications for probabilistic logic in Bayesian networks and datamining. They have succeeded in developing a new family of default formalisms which have turned out to be qualitative approximations of robust entropy maximization. The tool set of knowledge representation has also been enriched by the introduction of a very general representation independence concept and a natural semantics and proof theory for default quantification solving major problems of previous attempts.

Constraint solving

Coordinators: Alexander Bockmayr, Andreas Podelski

Under this headline we study both numerical constraints as they arise in mathematical optimization as well as constraints over symbolic structures (finite and infinite trees) as the are dealt with in program analysis.

Peter Barth, Alexander Bockmayr, and Thomas Kasper have continued to work on methods and tools for solving combinatorial problems in 0-1 variables. In contrast to existing finite domain constraint solvers, which are all based on domain reasoning, the distinguishing feature of their approach is to use more powerful inference systems, which are based on cutting planes, in order to prune the enumeration tree. Two software systems, a logic-based and a polyhedral branch-and-cut solver for pseudo-Boolean constraints, OPBDP and COUPE, are now available. Furthermore, to support model building in practical problem solving, an algebraic modeling system based on logic programming, PLAM, has been developed. The group has started a cooperation with COSYTEC SA, one of the leading companies in constraint technology, in order to apply their techniques to real world industrial problems and to find out how they might be combined with COSYTEC's CHIP system.

Witold Charatonik and Andreas Podelski have identified practically useful fragments of set constraints for which constraint solving is less complex than in the general case where it is NEXPTIME-complete. For instance they have shown that set constraints with intersection are the same as the so-called definite set constraints and have obtained DEXPTIME completeness for the complexity of the problem. They also have identified a class of set constraints that arise naturally in the context of analysing reactive (nonterminating) programs. The complexity of this class has also turned out to be DEXPTIME-complete.

Christoph Weidenbach has investigated the decidability and complexity of the unification and emptiness problem for a variety of so-called sort theories. These may be viewed as Horn theories for monadic predicates obeying certain syntactic restrictions. The most expressive among the decidable ones of his sort theories allow for certain non-linear variable dependencies. Sort theories are related to set constraints and tree automata. We use them successfully to cut the search space of saturation-based theorem proving in SPASS˙

Sergei Vorobyov has identified several natural problems in λ-calculus and higher-order logic that require non-elementary complexity of the 'worst' kind. For instance, he showed that a certain decidable higher-order theory of finite sets due to Henkin requires time to decide that is bounded from below by a tower of exponentials that is of exponential height in the length of the input. Previously only theoretical constructions by diagonalization were known to yield languages of similar bad complexity. He has also refuted a conjecture of Kunen about the complexity of the decision problem for the first-order theory of finite trees and related constraint structures. They turn out to be of non-elementary complexity (with a linearly growing tower of exponentials) rather than 'just' being PSPACE-complete as conjectured.

4 First-order theorem proving and term rewriting

Work in this area on theoretical and experimental questions has continued. On the methodological level we have given a systematic account of the principal resolution-based methods and described them as special cases of a general non-clausal resolution scheme (§4.1). We have continued our work on combining algebra and universal algebra in chaining- and superposition-based theorem proving (§4.3). A related issue is that of dealing with first-order theories of non-symmetric binary relations by techniques based on commutation of rewrite systems (§4.4). We have started to look at methods of soft typing, as known from program analysis, as a way to reduce proof search (§4.1). The concept of saturation up to redundancy by ordered resolution has been applied as a method of obtaining new decision procedures for certain fragments of first-order and modal logic (§4.2), and as a tool for obtaining complexity bounds for decidable theories (§4.1).

On the experimental side we have finished a first production version of the prover SPASS. SPASS has shown itself to be one of the leading three systems world-wide for automated theorem proving in first-order logic (see §9.1).

4.1 Refinements of resolution

Leo Bachmair, David Basin, Harald Ganzinger, Christoph Meyer and Christoph Weidenbach

A unified theory of resolution There is an abundance of resolution calculi and related theorem proving methods that have been described in the literature, e.g., binary resolution, hyper-resolution, semantic resolution, theory resolution, Boolean ring superposition, just to name a few. We have given a systematic account of most of these methods by showing how they can be derived as special cases of a very general scheme of ordered non-clausal resolution with selection and with simplification [5]. Simplification techniques are justified within our general concept of redundancy [1]. This work is the first uniform treatment of resolution-like techniques in which the avoidance of redundant computations receives the attention it deserves as something indispensable to the acceptable performance of any practical theorem prover. In many cases our presentation of a known resolution method indicates new potential for an improvement of the method. For some methods (Boolean ring superposition, Maslov's inverse method) the previously unknown relation to resolution has been clarified. For others we provide a simple means for proving completeness of more efficient variants (Boolean ring resolution, ordered theory resolution). We also give answers to several open technical problems in the area. For instance we show why tautologies cannot be eliminated in lock resolution. We identify ways of mixing clausal and non-clausal forms of resolution. We give a completeness proof for positive and negative variants of Boolean ring resolution. We analyse the effect of simplification on intermediate results in hyper-resolution inferences. We show how to apply this theory of resolution to semantic tableau, obtaining refinements which, for instance, include ordering restrictions and selection.

Soft typing for ordered resolution The semantic theory of resolution laid down in [5] allows for semantic refinements of resolution in the spirit of [9]. In [7] we propose a variant of ordered resolution with semantic restrictions based on interpretations which are identified by the given atom ordering and selection function. That is, the ordering, together with the set of reductive clauses for that ordering, is used to identify a partial interpretation that serves as a model hypothesis for the given set of clauses. Inferences then are selected with respect to the truth value of the premises in this interpretation. For instance, one of the premises must be false in the interpretation for an inference to be a selectable candidate. These additional semantic restrictions (they are compatible with the usual restrictions from orderings, selection, and redundancy) substantially further reduce the search tree. Techniques for effectively approximating validity in these interpretations, related to methods of soft typing for programming languages, have been described. The framework has been shown strictly more general than certain related methods, including original semantic resolution [9] and the method applied in the Satchmo prover [3]. Implementation in the SPASS prover of a subset of the described techniques has lead to encouraging experimental results (see §9.1).

SPASS implements the static variant (the general theory accommodates dynamically changing model hypotheses) of semantic resolution with soft types. Here a sort theory (see §8.1) is extracted from the initial clause set and used thereafter for inference selection and clause simplification. If $P(x)$ is a predicate that occurs in the input clauses, a type $T_P(x)$ will be automatically extracted for it such that $T_P(x)$ is an approximation, in fact a superset, of the extension of $P(x)$ in any model of the input clauses. The sort theory is such that unification problems $\forall x_1 : T_{P_1}(x_1) \ldots \forall x_k : T_{P_k}(x_k).\, s(\vec{x}) = t(\vec{x})$, as they arise from resolution and paramodulation inferences, are decidable. Apart from an analysis of such unification problems, in [8] we give proofs of refutational completeness for refinements of semantic tableau, ordered resolution, and superposition with static sort theories. In particular it is shown that superposition inferences with a clause in which the sort constraint for its variables is not solved yet can be blocked. More importantly, superposition inferences into the sort constraint part of a clause turn out to be redundant (which extends a result already known for basic superposition).

Complexity analysis based on ordered resolution We have found out that saturation up to redundancy by ordered resolution can be used as a tool for constructing presentations of a first-order theory for which ground entailment becomes decidable in time depending on the ordering. Given a first-order theory N (in clausal normal form), ground entailment for N is the problem of deciding logical implication $N \models C$ for ground clauses C. For instance, reachability r in a directed graph is an entailment problem of the form

$$\forall x, y, z : r(x, y) \wedge r(y, z) \supset r(x, z) \models r(v_{i_1}, v_{i_2}) \wedge \ldots \wedge r(v_{i_n}, v_{i_n}) \supset r(v, v')$$

in which the antecedent atoms $r(v_{i_j}, v_{i_j})$ represent the (edges of the) graph, $r(v, v')$ is the instance of the reachability problem that one wants to answer, and where the theory simply expresses that reachability is a transitive relation. Our research was motivated by the work of McAllester and Givan [4, 2] who gave procedures for recognizing sets of Horn clauses which generate polynomial time decidable entailment relations. McAllester's procedures test if Horn clauses have a property, which he calls locality, which roughly corresponds to a kind of subformula property for Horn clause proofs; when a clause set has this property, then there are a bounded number of terms (dependent only on the query) which need appear in proofs and this is used to solve ground queries in polynomial time.

We have extended this work by (i) identifying a close relationship between locality and saturation up to redundancy under ordered resolution, and by (ii) extending automatic complexity analysis to other complexity classes other than P [6]. We define *order locality* to be a property of clauses relative to a term ordering. This property is a generalization of the subformula property for proofs where terms arising in proofs are bounded, under the given ordering, by terms appearing in the goal clause. We have shown that when a clause set is order local, then the complexity of its ground entailment problem is a function of its structure (e.g., full versus Horn clauses), and the ordering used. We have proven that, in many cases, order locality is equivalent to a clause set being saturated under ordered resolution. This provides a means of using standard resolution theorem provers for testing order locality and transforming non-local clause sets into local ones. We have used the Saturate system (§9.2) to automatically establish complexity bounds for a number of nontrivial entailment problems relative to complexity classes which include Polynomial and Exponential Time and co-NP.

References

[1] L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994. Revised version of Technical Report MPI-I-91-208, 1991.

[2] R. Givan and D. McAllester. New results on local inference relations. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR-92)*, pages 403–412. Morgan Kaufmann Press, October 1992.

[3] R. Manthey and F. Bry. Satchmo: A theorem prover implemented in prolog. In E. Lusk and R. Overbeek, editors, *9th International Conference on Automated Deduction,CADE-9*, volume 310 of *Lecture Notes in Computer Science*, pages 415–434. Springer, 1988.

[4] D. A. McAllester. Automatic recognition of tractability in inference relations. *J. ACM*, 40(2):284–303, April 1993.

- [5] L. Bachmair and H. Ganzinger. A theory of resolution. In J. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier, Amsterdam, 1997. To appear.

- [6] D. A. Basin and H. Ganzinger. Complexity analysis based on ordered resolution. In *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS'96)*, pages 456–465, New Brunswick, New Jersey, USA, 1996. IEEE Comput. Soc. Tech. Committee on Math. Found. Comput.; ACM Special Interest Group on Automata & Comput. Theory; Assoc. Symbolic Logic; Eur. Assoc. Theoret. Comput. Sci, IEEE Comput. Soc. Press.

- [7] H. Ganzinger, C. Meyer, and C. Weidenbach. Soft typing for ordered resolution. In *Proceedings of the 14th International Conference on Automated Deduction (CADE-14)*, Lecture Notes in Computer Science, Townsville, Australia, 1997. Springer. to appear.

- [8] C. Weidenbach. *Computational Aspects of a First-Order Logic with Sorts.* PhD thesis, Universität des Saarlandes, 1996.

[9] J. R. Slagle. Automatic theorem-proving with renamable and semantic resolution. *Journal of the ACM*, 14:687–697, 1967.

4.2 Resolution as a decision procedure

Ullrich Hustadt and Renate Schmidt

Since the late sixties decision procedures based on resolution have been a topic of research on automated theorem proving. So as to ensure a resolution procedure terminates for a decidable first-order class two parameters must be controlled: (i) the growth of nesting of functional terms in resolvents and (ii) the growth in size of the resolvents. For most solvable classes considered in the literature, unrestricted resolution is only a semi-decision procedure, since either terms grow without bounds or resolvents have increasing numbers of literals.

In [2] we present a new solvable class, namely basic path logic, that is as far as we know the only non-trivial solvable class that is decidable by unrestricted resolution. Basic path logic arises by the optimized functional translation from basic propositional modal logic. It is a clausal logic of clauses built from unary predicate symbols and linear terms that are strings of variables and constants required to satisfy an ordering condition of the variables. Unrestricted resolution together with condensing decides satisfiability for this logic. The increase of nesting of functional terms is not a problem in basic path logic as terms do not expand during deduction. However, the number of variables in clauses may increase, which makes exhibiting a bound for the second parameter, the maximal clause size, difficult. The proof involves an embedding of variables into a finite Boolean algebra by a technique of nested partitioning.

In general, special refinements need to be devised that render resolution decision procedures for solvable classes. Classes often considered in this context have particularly simple clausal forms, for example, clausal forms in which all variables that occur in a clause occur in all literals of that clause. Resolution preserves this property and ordering refinements ensure that the maximal depth of variable occurrences does not increase.

More challenging is the problem of finding a resolution procedure that decides satisfiability for Maslov's class K, and many attempts have fallen short of solving the problem. The class is very general and contains the monadic class, the initially extended Skolem class and the Gödel class. Maslov states that the inverse method decides the validity of disjunctions of formulae in the class K. The question whether his result can be transferred to resolution-based approaches has been open since 1968. Only in 1993 did Zamov describe a decision procedure for a subclass of the class K that uses lock resolution. We have shown that there is a resolution decision procedure for the entire class K, and the proof will appear as part of U. Hustadt's dissertation. His procedure utilizes an ordering restriction such that for all clauses in a derivation the nesting of functional terms and the number of literals is bounded, even though both the nesting of functional terms and the maximal depth of variable occurrences can increase. The decidability proof makes use of the concept of redundancy due to Bachmair and Ganzinger [1].

Both, basic path logic and Maslov's class K are of particular interest due to their close relation to modal logic (see §6).

References

[1] L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994. Revised version of Technical Report MPI-I-91-208, 1991.

• [2] R. A. Schmidt. Resolution is a decision procedure for many propositional modal logics: Extended abstract. To appear in Kracht, M., de Rijke, M., Wansing, H. and Zakharyaschev, M. (eds), *Advances in Modal Logic '96.* CSLI Publications. The long version is available as Research Report MPI-I-97-2-002, 1997.

4.3 Refinements of superposition

Leo Bachmair, Harald Ganzinger, Georg Struth, Jürgen Stuber and Uwe Waldmann

Superposition is a refutationally complete refinement of the paramodulation calculus parameterized by a syntactic ordering on terms (Bachmair and Ganzinger [1]). Its inference rules are restricted to overlaps of maximal sides of maximal literals. Together with global simplification techniques, these local ordering conditions give rise to a significant reduction of the search space.

Most applications of theorem proving include uninterpreted function and predicate symbols that are specific for a particular domain of application, as well as standard algebraic structures and theories, such as the natural numbers, or abelian groups, or orderings. Unfortunately, axioms like associativity, commutativity or transitivity are difficult for a superposition theorem prover, since they allow a huge number of inferences and tend to generate many equivalent formulae. A sophisticated treatment of the standard theories is therefore crucial to the performance of the prover. To this end mathematical and meta-mathematical techniques need to be combined.

It is rarely possible to link a decision procedure for a decidable theory to a prover as a black box: The requirement of sufficient completeness [2] practically excludes uninterpreted function symbols; and even in applications where sufficient completeness is not too restrictive a requirement, insufficient communication between the external decision procedure and the general prover hampers the performance [3]. Consequently, the integration has to be achieved on the level of the inference system and simplification techniques.

References

[1] L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994. Revised version of Technical Report MPI-I-91-208, 1991.

[2] L. Bachmair, H. Ganzinger, and U. Waldmann. Refutational theorem proving for hierarchic first-order theories. *Applicable Algebra in Engineering, Communication and Computing (AAECC)*, 5(3-4):193–212, April 1994. Earlier Version: Theorem Proving for Hierarchic First-Order Theories, in Giorgio Levi and Hélène Kirchner, editors, *Algebraic and Logic Programming, Third International Conference*, LNCS 632, pages 420–434, Volterra, Italy, September 2–4, 1992, Springer-Verlag.

[3] R. S. Boyer and J. S. Moore. Integrating decision procedures into heuristic theorem provers: A case study of linear arithmetic. In J. E. Hayes, D. Michie, and J. Richards, editors, *Machine Intelligence 11: Logic and the acquisition of knowledge*, chapter 5, pages 83–124. Oxford University Press, 1988.

Abelian Groups

Bachmair and Ganzinger [1] and Wertz [11] have integrated the associativity and commutativity axioms into superposition using AC-unification and extended clauses, developed for the equational case by Peterson and Stickel [8]. In this way, the prolific inferences with the theory axioms become superfluous, but new sources of inefficiency emerge, since a minimal finite set of AC-unifiers may have doubly exponential size. Under these circumstances, the possibility to delay unifications by using constraints becomes particularly valuable: the enumeration of unifiers can be avoided and the number of generated conclusions is drastically reduced. Only if the empty clause with some constraint T has been derived, is it necessary to solve T, that is, to decide the unifiability problem. This takes still simply exponential time, though (Nieuwenhuis and Rubio [7], Vigneron [9]).

The problem can be mitigated by integrating more axioms. If our theory also contains the identity law, then AC-unification can be replaced by ACU-unification (unification modulo AC and identity), which is only simply exponential [2, 3]. We observe a much more radical improvement when we switch over from the abelian semigroup axioms AC or the abelian monoid axioms ACU to the axioms of abelian groups. To see the operational difference between these theories in a rewrite or superposition-based calculus, consider the two unit clauses $u_1 + \cdots + u_k \approx s$ and $v_1 + \cdots + v_l \approx t$. In AC-superposition, there is an inference between these two clauses (via extended clauses), whenever *some* u_i is unifiable with *some* v_j. In the presence of the inverse axiom, extended rules become superfluous, and a superposition inference is required only if the *maximal* u_i is unifiable with the *maximal* v_j. This technique can be found for instance in normalized rewriting (Marché [4]) and in our extension of the superposition calculus [6], where we have used integer modules to represent abelian groups. That is, multiple occurrences of the same term in a sum, say $t_1 + \cdots + t_n$ where $t_1 =_{AC} \cdots =_{AC} t_n$, are replaced by a product $n \cdot t_1$, and the scalars are manipulated by constraints.

The restriction to overlaps of maximal summands in maximal sides of maximal literals implies in particular that there are no superpositions with shielded variables. While inferences with unshielded variables cannot generally be avoided, we have presented several simplification techniques that allow us to reduce the number of literals with unshielded variables.

The group axioms yield both the existence of the difference of any two elements and its uniqueness. Operationally, the second property is much more important than the first one. We can thus employ similar techniques for cancellative abelian monoids, that is, for abelian monoids in which the cancellation axiom $x + y \approx x + z \rightarrow y \approx z$ holds. Cancellative abelian monoids are in some sense the most general algebraic structure where such an "abelian-group-like" reasoning is possible. They comprise not only abelian groups, but also such ubiquitous structures as the natural numbers or multisets. In [5], we have presented a refined superposition calculus for sets of clauses including the axioms of cancellative abelian monoids. As in the abelian group calculi above, ordering restrictions can be strengthened and explicit inferences with the theory axioms and extended clauses are superfluous. In [10] we have demonstrated that our calculus offers also the possibility to integrate the torsion-freeness axioms $n \cdot x \approx n \cdot y \rightarrow x \approx y$ for positive integers n. While torsion-freeness complicates negative superposition inferences, it also makes new variable elimination techniques applicable. Since an abelian monoid is cancellative and torsion-free if and only if it can be totally ordered, our calculus allows us to solve equational problems in ordered abelian monoids or groups without having to derive intermediate clauses containing ordering literals.

References

[1] L. Bachmair and H. Ganzinger. Associative-commutative superposition. In N. Dershowitz and N. Lindenstrauss, editors, *Proceedings of the 4th International Workshop on Conditional and Typed Rewrite Systems*, volume 968 of *Lecture Notes in Computer Science*, pages 1–14, Jerusalem, Israel, 1995. Springer. Revised version of TR MPI-I-93-267, 1993.

[2] A. Boudet, E. Contejean, and C. Marché. AC-complete unification and its application to theorem proving. In H. Ganzinger, editor, *Rewriting Techniques and Applications, 7th International Conference, RTA-96*, LNCS 1103, pages 18–32, New Brunswick, NJ, USA, July 27–30, 1996. Springer.

[3] J.-P. Jouannaud and C. Marché. Termination and completion modulo associativity, commutativity and identity. *Theoretical Computer Science*, 104(1):29–51, Oct. 1992.

[4] C. Marché. Normalized rewriting: an alternative to rewriting modulo a set of equations. *Journal of Symbolic Computation*, 21(3):253–288, Mar. 1996.

• [5] H. Ganzinger and U. Waldmann. Theorem proving in cancellative abelian monoids (extended abstract). In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 388–402, New Brunswick, USA, 1996. Springer. Full version: Technical Report MPI-I-96-2-001, Max-Planck-Institut für Informatik, Saarbrücken, Germany, January 1996.

• [6] J. Stuber. Superposition theorem proving for abelian groups represented as integer modules. In H. Ganzinger, editor, *Rewriting Techniques and Applications, 7th International Conference, RTA-96*, volume 1103 of *Lecture Notes in Computer Science*, pages 33–47, New Brunswick, USA, 1996. Springer.

[7] R. Nieuwenhuis and A. Rubio. AC-superposition with constraints: no AC-unifiers needed. In A. Bundy, editor, *Twelfth International Conference on Automated Deduction*, LNAI 814, pages 545–559, Nancy, France, June 26–July 1, 1994. Springer.

[8] G. E. Peterson and M. E. Stickel. Complete sets of reductions for some equational theories. *Journal of the ACM*, 28(2):233–264, Apr. 1981.

[9] L. Vigneron. Associative-commutative deduction with constraints. In A. Bundy, editor, *Twelfth International Conference on Automated Deduction*, LNAI 814, pages 530–544, Nancy, France, June 26–July 1, 1994. Springer.

[10] U. Waldmann. *Cancellative Abelian Monoids in Refutational Theorem Proving.* Dissertation, Universität des Saarlandes, 1997. Submitted.

[11] U. Wertz. First-order theorem proving modulo equations. Research Report MPI-I-92-216, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, April 1992.

Chaining calculi for binary relations

We have further extended and improved our chaining calculi for first-order theories of transitive relations that we have introduced in [1, 2]. The chaining calculi are based on rewriting techniques for commuting rewrite systems (§4.4). In [3] we describe, and prove complete, a chaining calculus in which the concept of rewrite proofs is extended to (certain classes of) sets of composition laws of the form $R \circ S \subseteq T$ for binary relations R, S, and T. For instance with the standard definitions for the composition relations such as strict and non-strict ordering $<$, $\leq$ together with equality $\approx$, ordinary resolution and superposition inferences turn out to be extremely prolific. Within our chaining calculi they are dealt with by specifically refined chaining inferences.

The improvements with respect to our previous work are mainly on the methodological level. We show how specialized systems for orderings, partial congruences (that is, congruences without reflexivity as they arise in certain formalizations of partial functions), and equality can be easily derived from the general case with the help specifically refined syntactic orderings for atoms and clauses. These specialization in particular enjoy compatibility with specific variable elimination techniques as they come with the specific theories. To that end, many technical problems needed to be solved.

We also have made progress towards a chaining inference system for fragments of set theory with subset inclusion and projective geometry. The systems are being developed systematically applying techniques from non-symmetric completion [4] and proof theory.

References

[1] L. Bachmair and H. Ganzinger. Ordered chaining for total orderings. In A. Bundy, editor, *Proceedings of the 12th International Conference on Automated Deduction (CADE-12)*, volume 814 of *Lecture Notes in Artificial Intelligence*, pages 435–450, Nancy, France, 1994. Springer-Verlag. Full version available as Research Report MPI-I-93-250, 1993.

[2] L. Bachmair and H. Ganzinger. Rewrite techniques for transitive relations. In *Proceedings of the 9th IEEE Symposium on Logic in Computer Science*, pages 384–393. IEEE Computer Society Press, 1994. Full version available as Technical Report MPI-I-93-249.

• [3] L. Bachmair and H. Ganzinger. Ordered chaining calculi for first-order theories of binary relations. Research Report MPI-I-95-2-009, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, October 1995.

• [4] G. Struth. Non-symmetric rewriting. Research Report MPI-I-96-2-004, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, June 1996.

4.4 Term rewriting

M. R. K. Krishna Rao, Georg Struth and Sergei Vorobyov

During the period relevant to this report, research in term rewriting concentrated on graph rewriting and non-symmetric rewriting. In addition we have investigated the decidability problem of one step term rewriting.

Non-copying implementations of term rewriting

Graph rewriting is a compact implementation of term rewrite systems, where terms, which are naïvely perceived as trees, are represented as directed acyclic graphs by (partially) collapsing the shared structure of the associated tree. The reduction of evaluation costs due to this implementation is remarkable: While in tree-based implementations terms can grow exponentially with the length of computations, non-copying implementations only lead to linear space complexity. Graph rewriting is sound in the sense that all computations of normal forms of graphs yield normal forms for the associated terms. It is also complete in the sense that a graph can be reduced if and only if the term corresponding to the redex can be reduced to a term corresponding to the reduct. However, the relation between term and graph rewriting is in general quite subtle and certain properties, as, for instance, confluence, weak normalization, termination and modularity of confluence, are not preserved under the transformation from trees to graphs. Conversely, termination is modular for the graph implementation (with respect to the direct sum), but not for tree rewriting. As an undesired side effect of these non-preservation properties, graph rewriting fails to compute all existing term normal forms. These properties make a rigorous study of study various aspects of graph-based implementations important.

In [4], we state sufficient conditions for preservation of weak normalization. We call a term rewrite system graph reducible if every collapsed tree represents a weakly normalizable term, i.e., a term possessing some normal form, is weakly normalizable with respect to graph rewriting. We call a term rewrite system weakly graph reducible, if some collapsed tree representing a weakly normalizable term is weakly normalizable with respect to graph rewriting. Our main result shows that every weakly innermost normalizing term rewrite system is graph reducible. We compare our result to previous ones [7], which are specialized to a variant of parallel rewriting, by establishing various relations between parallel normalizing and innermost normalizing classes. Moreover we show that semi-completeness, i.e., confluence together with weak normalization, is modular for weakly innermost normalizing systems with disjoint signatures. Finally we state sufficient syntactic criteria for graph reducibility. On the negative side, non-overlapping as well as outermost normalizing systems need not be graph reducible. On the positive side, we can go beyond orthogonality by showing that right-linear systems are a weakly graph reducible class, and by identifying two classes of term rewrite systems with normal forms entirely computable via graph rewriting. One class is that of right-linear systems, the other that of strongly innermost normalizing overlay systems.

In [6] we investigate modularity of termination in term graph rewriting and consider hierarchical combinations, where operators defined in one system are used as built-ins in the other one. We generalize previous results on modular systems [7, 2] to a larger class. Modularity of termination in term graph rewriting is in a certain sense orthogonal to the standard case, where modularity holds, provided the two systems under consideration are either both confluent and terminating, or simply terminating, i.e., their termination can be proven by simplification orderings. For term graph rewrite systems we show that neither confluence nor simple termination is needed. Our results are important for incremental synthesis of rewrite systems and an divide-and-conquer approach to their analysis.

In [5] we study completeness of basic narrowing in term graph rewriting, that is the ability to compute all solutions of an equation under that strategy. Our results show that term graph rewriting not only improves efficiency, but even provides more general completeness results that the usual tree-based techniques. More precisely, basic narrowing is complete with respect to term graph rewriting for the following classes of term rewrite systems with confluent and normalizing graph rewriting relations: First, overlay systems containing the classes of weakly orthogonal systems and inductively sequential systems studied in [1, 3], second weakly innermost normalizing systems and third right-linear systems with weakly normalizing and confluent graph rewrite relations. The importance of our results lies in the fact that most recent proposals for an integration of functional and logic programming languages based on a sound and complete operational semantics were based on narrowing.

References

[1] S. Antoy, R. Echahed, and M. Hanus. A needed narrowing strategy. In *Proceedings of the 21st ACM Symposium on Principles of Programming Languages (POPL'94)*, pages 268–279, Portland, USA, 1994. ACM Press. Extended version available as Research Report MPI-I-93-243, Max-Planck-Institut für Informatik, Saarbrücken, Germany.

[2] K. Kurihara and A. Ohuchi. Modularity in noncopying term rewriting. *TCS*, 152:139–169, 1995.

• [3] M. Hanus. On extra variables in (equational) logic programming. In *Proceedings of the Twelfth International Conference on Logic Programming*, pages 665–679, Kanagawa, Japan, 1995. MIT Press.

• [4] M. R. K. Krishna Rao. Graph reducibility of term rewriting systems. In J. Wiedermann and P. Hájek, editors, *Proceedings of Mathematical Foundations of Computer Science*, volume 969 of *Lecture Notes in Computer Science*, pages 371–381, Prague, Czech Republic, August 1995. Springer.

• [5] M. R. K. Krishna Rao. Completeness results for basic narrowing in non-copying implementations. In M. Maher, editor, *Logic Programming*, JICSLP, pages 393–407, Bonn, Germany, 1996. The Association for Logic Programming, MIT Press.

• [6] M. R. K. Krishna Rao. Modularity of termination in term graph rewriting. In H. Ganzinger, editor, *Rewriting Techniques and Applications*, volume 1103 of *Lecture Notes in Computer Science*, pages 230–244, New Brunswick, July 1996. Springer.

[7] D. Plump. Graph–reducible term rewriting systems. In H. Ehrig, H.-J. Kreowski, and G. Rozenberg, editors, *Proc. of Graph Grammars and Their Application to Computer Science*, volume 532 of *Lecture Notes in Computer Science*, pages 622–636, Bremen, GER, March 1990. Springer.

Non-symmetric rewriting

For non-symmetric rewriting, starting from ideas of Levy and Agustí [2], and Bachmair and Ganzinger [1], we have generalized some of the theory of term rewriting from the equational case to possibly non-symmetric transitive binary relations modulo a congruence [4]. Such a generalization is needed when one wants to apply rewriting methods beyond equational specifications in relational structures, such as Kripke structures for modal logics, subset inclusion, or partial and total orderings.

The approach is based on the simple observation that appropriate syntactic term orderings partition transitive relations into two sets of rewrite rules. As a simple example consider the relations $p(x) < x$ and $x < s(x)$ for the predecessor and the successor in arithmetics. Obviously in the first inequality, the left-hand side is syntactically more complex than the right-hand side, whereas for the second inequality the converse is true. Intersection with the syntactic ordering and its inverse, respectively, yields two sets R and S of rules which rewrite terms to syntactically smaller, respectively, larger, terms. (In both systems, rewriting produces semantically larger terms with respect to $<$.) In our example, we obtain $p(x) \rightarrow_R x$ and $x \rightarrow_S s(x)$. This partition turns sequences of $<$-inequalities into "mountain ranges" reminiscent of the peaks and valleys in equational rewriting. To prove $p(p(0)) < s(0)$, for instance, one rewrites $p(p(0)) \rightarrow_R p(0) \rightarrow_R 0 \leftarrow_S s(0)$. This proof has the form of a rewrite proof. In analogy to the equational case, non-symmetric rewriting provides effective techniques to rearrange mountain ranges as rewrite proofs, and to obtain completion procedures, by generalizing confluence properties to properties of commutation and by modifying the concept of a rewrite proof accordingly. The decision procedure associated with non-symmetric rewriting, however, is weaker than in equational rewriting in that search cannot be entirely avoided since the concept of unique normalforms is meaningless here. Nevertheless the procedure still yields acceptable results where equational rewriting fails, as we show by proving completeness of a solution to the word problem in the variety of lattices proposed by Levy and Agustí [5].

We have extended the approach not only to a non-symmetric version of term rewriting in presence of monotonic operators, but also to non-symmetric rewriting modulo a congruence and to normalized non-symmetric rewriting à la Marché [3]. However, as a drawback of the generalization, also critical pairs corresponding to overlaps below non-linear variables have to be considered. These cannot be finitely represented in the realm of first-order logic and therefore are severe obstacles to the automatization of completion procedures.

Our results comprise rewriting with quasi-orderings, as investigated by Levy and Agustí, and liberate their approach from severe restrictions. We are able to localize "cliffs" and we do not need any linearity restrictions in bi-rewriting modulo a congruence. Our results are presented such that the equational case is (strictly) included. In that setting we provide a combinatorial foundation of term rewriting in which commutation and reduction are the central concepts, whereas term normal forms only arise as a side effect of the special equational case. In the area of automated deduction, our work is meant to contribute a conceptual basis for efficient chaining calculi for relational theories (§4.3).

References

[1] L. Bachmair and H. Ganzinger. Rewrite techniques for transitive relations. In *Proceedings of the 9th IEEE Symposium on Logic in Computer Science*, pages 384–393. IEEE Computer Society Press, 1994. Full version available as Technical Report MPI-I-93-249.

[2] J. Levy and J. Agustí. Bi-rewriting, a term rewriting technique for monotonic order relations. In C. Kirchner, editor, *Proc. of Rewriting Techniques and Applications*, volume 690 of *Lecture Notes in Computer Science*, pages 17–31. Springer, 1993.

[3] C. Marché. Normalised rewriting: an alternative to rewriting modulo a set of equations. *J. Symbolic Computation*, 21:253–288, 1996.

- [4] G. Struth. Non-symmetric rewriting. Research Report MPI-I-96-2-004, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, June 1996.

- [5] G. Struth. On the word problem for free lattices. In *Rewriting Techniques and Applications, 8th International Conference (RTA-97), Proceedings*, Lecture Notes in Computer Science, Sitges, Spain, June 1997. Springer. to appear.

Theories of one step term rewriting

A finite term rewriting system R generates the binary one step ground reducibility relation R on the set of closed terms of underlying signature. A theory of one step rewriting in R is the first-order theory of this binary relation R. The problem whether all first-order theories of one step rewriting in finite systems are decidable was put forward in [1]. Positive solutions would have interesting application. For example, the ground reducibility of a term $t(\overline{x})$ (a central concept for "inductionless" inductive proofs) and the strong confluence of a system are expressible by the formulae $\forall \overline{x} \exists y R(t(\overline{x}), y)$ and $\forall x, y, z \exists w (R(x,y) \wedge R(x,z) \rightarrow R(y,w) \wedge R(z,w))$ respectively. Note that both properties are known to be decidable. Similarly, the decidability of properties like encompassment [1] would follow from the general decidability of one step rewriting. Also the first-order theories of one step rewriting in finite ground systems are decidable [2].

Unfortunately, the problem was settled in the negative in [5, 6]. The proof in [6] even exhibited a particularly simple *fixed* rewrite rule system with an undecidable theory. The proof in [5] demonstrated that there exist rewrite systems with undecidable $\exists\forall$-theories, without giving any fixed examples of such systems.

Both proofs [5, 6] essentially used divergent and non-linear rewrite rules. Moreover, the proof [5] directly used rules of the form $t \to t$, which one never encounters in practice. This somehow diminished the practical relevance of the obtained results and left the hope that the theories of one step rewriting should be decidable for finitely terminating systems. This conjecture, however, we have also settled in the negative in [3], where a specific finite, finitely terminating, and linear system with an undecidable theory of one step rewriting was constructed. The proof of [3], as well as [6], are obtained by the reduction from the theory of concatenation (finitely generated free semigroups), known to be undecidable from [4]. While the constructed rewrite system is simple, the quantifier alternation of the sentences forming the reduction class in [3] is quite high. Work in progress seems to indicate that already the $\exists\forall\exists$-theories of one step rewriting for the same class of rewrite systems is undecidable.

References

[1] A.-C. Caron, J.-L. Coquidé, and M. Dauchet. Encompassment properties and automata with constraints. In C. Kirchner, editor, *Rewriting Techniques and Applications'93*, volume 690 of *Lecture Notes in Computer Science*, pages 328–342. Springer, 1993.

[2] M. Dauchet and S. Tison. The theory of ground rewrite systems is decidable. In *Proc 5th IEEE Symp Logic in Computer Science*, pages 242–256, 1990.

• [3] S. Vorobyov. The first-order theory of one step rewriting in linear noetherian systems is undecidable. In *Proceedings of the 8th International Conference on Rewriting Techniques and Applications (RTA'97)*, Lecture Notes in Computer Science, Sitges, Spain, 1997. Springer. To appear.

[4] W. V. O. Quine. Concatenation as a basis for arithmetic. *J. Symb. Logic*, 11(4):105–114, 1946.

[5] R. Treinen. The first-order theory of one step rewriting is undecidable. In *Rewriting Techniques and Applications'96*, Lecture Notes in Computer Science, pages 276–285. Springer, 1996.

[6] S. Vorobyov. The elementary theory of one-step rewriting is undecidable (note). Unpublished draft, see `http://www.mpi-sb.mpg/~sv/`, Section "On Trees and Rewriting", 1995.

4.5 Optimized and strong skolemization

Andreas Nonnengart

One major preparatory step in clause-based theorem proving is the so-called skolemization. Its purpose is to eliminate existential quantifiers and to replace them by suitable skolem function applications.

In [2] and [1] we introduced two skolemization approaches, called Optimized Skolemization and Strong Skolemization respectively, which allow us to, first, reduce the number of clauses in the final clause normal form transformation result and, second, to abstract from some of the arguments of the newly introduced skolem functions by universally quantified variables not yet involved. The Optimized Skolemization does so by introducing suitable lemmas to be proved during the skolemization process. In this sense it is a non-local method which takes into account not only the variables visited so-far, but also all the other formulae of the given problem that are usually not considered by skolemization. The Strong Skolemization does not involve intermediate lemmas to be proved. It is thus fully syntactic and just utilizes certain knowledge about the positions of variables. Strong Skolemization produces a result which is almost identical to the one of standard Skolemization; only some of the skolem function arguments are replaced by new universally quantified variables.

Both skolemization methods turned out to be superior to the approaches mentioned in the standard literature in the sense that the ultimately resulting clause set contains either fewer clauses with fewer literals (in case the Optimized Skolemization applies) or, just the same amount of clauses and literals as in the classical case, however, with a certain abstraction what the arguments of skolem functions is concerned. The two methods don't interfere with each other if Strong Skolemization is applied after Optimized Skolemization and it thus makes sense to include both in a clause normal form transformation system.

Optimized skolemization is implemented in the clause normal form transformation system FLOTTER (see §9.1). The next version of FLOTTER will be extended to cover strong skolemization as well.

References

- [1] A. Nonnengart. Strong skolemization. Research Report MPI-I-96-2-010, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, December 1996.
- [2] H. J. Ohlbach and C. Weidenbach. A note on assumptions about skolem functions. *Journal of Automated Reasoning*, 15(2):267–275, 1995.

4.6 Data structures and implementation techniques

Peter Graf, Christoph Meyer and Christoph Weidenbach

Since the last progress report we have continued our work on indexing techniques supporting automated theorem proving. In the presence of associative and commutative function symbols indexing has to be carefully adopted due to the exponential explosion for possible answers. We developed Path-Indexing for AC-theories which is able to retrieve AC-unifiable terms, AC-generalizations, AC-instances and AC-variants [3].

In addition, we extended the theory of substitution trees [2, 4] in two directions. First, we generalized their data format from terms to sets of substitutions. This was motivated by theorem proving in Horn-clause fragments where it is possible to base the overall theorem proving process on the exchange of substitution sets between complementary literals. A well-known class of such a fragment are condensed detachment problems [1]. We developed the distributed theorem prover PURR [6] which uses the advanced substitution tree indexing techniques. In our experiments PURR shows superior behavior over other systems on this problem class. The second extension to substitution trees concerns the generalization of single term queries to operations on several indexes [5], similar to the join operator known from database technology. These operations are in particular useful in the context of hyper-resolution-like inference rules.

Since most of the time in a theorem proving process is spent by the check of redundancy criteria, it is necessary to support the applicability tests for such criteria. Therefore, we added incremental query retrieval in the implementation of substitution tree indexing. Given some query term, it is possible to incrementally ask for hits from the index. An example for the application of incremental retrieval is the subsumption test between clauses. Typically, given a test clause, one asks the index for generalizations of some literal of this clause to derive candidate clauses that probably subsume the test clause. If eventually a candidate is found that subsumes the test clause, the retrieval process can be stopped and no time is wasted in searching for other candidates.

The second implementation technique is the combination of substitution trees with a sharing structure. Although the substitution tree itself is able to perform sharing of the input terms, filtering the inputs with an appropriate sharing structure reduces the number of input terms, hence speeds up retrieval. Since sharing is used anyway in a prover to support the storage of huge clause sets, we get this speed up for free.

Both implementation techniques are incorporated in our theorem prover SPASS and our clause normal form translator FLOTTER (see §9.1).

References

[1] W. McCune and L. Wos. Experiments in automated deduction with condensed detachment. In *11th International Conference on Automated Deduction, CADE-11*, volume 607 of *LNCS*, pages 209–223. Springer, 1992.

- [2] P. Graf. *Term Indexing.* PhD thesis, Universität des Saarlandes, 1995.
- [3] P. Graf. Path indexing for AC-theories. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 718–732, New Brunswick, New Jersey, USA, 1996. Springer.
- [4] P. Graf. *Term Indexing*, volume 1053 of *Lecture Notes in Artificial Intelligence.* Springer, Berlin, 1996.
- [5] P. Graf and C. Meyer. Advanced indexing operations on substitution trees. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 553–567, New Brunswick, New Jersey, USA, 1996. AT&T, Bell Labs, Lucent, IEEE, CADE inc, Springer.
- [6] C. Meyer. Parallel unit resulting resolution. Master's thesis, Universität des Saarlandes, February 1996.

5 Logical frameworks

We have continued to investigate both the applications and the theory of *logical frameworks* (we define a logical framework to be an abstraction of some style of deduction system which can be used as the foundation of a generic proof system) in the widest sense; i.e. our research has not been restricted to the systems based on minimal implicational logic (what we call '⇒ frameworks') that are most commonly identified with the phrase, e.g. Isabelle [4] or the Edinburgh LF [2, 5], but has also included those based on theories of inductive definition (ID frameworks), e.g. FS_0 [1].

One part of our research has been into applications: we have been investigating how the level of abstraction provided by an ⇒ framework can be exploited, using higher-order unification, as a general framework in which to combine program verification and synthesis. In particular we have been exploring techniques for synthesising logic programs and electronic circuits (we have also adopted the same methods in providing a useable implementation of the inductive definitions framework FS_0). We have also explored the theoretical foundations of this sort of mechanism, and tried to develop a precise formal characterization of the necessary machinery. Along side this we have been looking at how decision procedures for decidable logics (e.g. the Mona system [3] for Monadic second-order logic) might be exploited as development tools in a proof development system; specifically, how they might integrated into, and exploited by implementations of larger and, in general, undecidable logics.

A second area of research has concentrated on how ⇒ frameworks can be used to provide effective implementations of 'non-standard' systems like modal or relevance logic which do not fit 'naturally' with a standard ⇒ framework. We have conducted an extensive investigation of the properties of *labelled deductive systems* (LDS)—proof systems for dealing with non-classical logics which are able to make controlled use of a semantics, without resorting to a complete semantic embedding. As a complement to the LDS approach, we have also investigated the possibility of conservative extensions to a ⇒ framework with a modal connective which allow a direct, proof-theoretic, encoding of non-standard logics.

As a contrast to ⇒ frameworks, we have also been exploring the implications and applications of inductive definition (ID) as an alternative style of logical framework. The advantage of ID is that with it we are able to build the sort of inductive arguments we need for general proof theory, something that is not possible with an ⇒ framework. The large disadvantage is that the basic notation is much more primitive. We have implemented a system of inductive definitions and used it both to explore how to make such a framework more practical, and the issues that arise when we try to combine software-engineering principles of structured development with ambitious metatheoretic facilities.

Finally, and in contrast to this proof-theoretic view of how logics behave, we have also been exploring more abstract semantic tools. In particular, institutions, which we have been using to investigate how the more abstract properties of the style of model based specification languages that is often proposed as suitable for the specification of large state-based systems (e.g. Z or VDM) can be described in a logic independent way.

References

[1] S. Feferman. Finitary inductive systems. In *Logic Colloquium '88*. North-Holland, Amsterdam, 1990.

[2] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. *J. Assoc. Comput. Mach.*, 40:143–184, 1993.

[3] J. Henriksen, J. Jensen, M. Jørgensen, N. Klarlund, B. Paige, T. Rauhe, and A. Sandholm. Mona: Monadic second-order logic in practice. In B. Steffen, editor, *Tools and Algorithms for the Construction and Analysis of Systems, First International Workshop*. Springer, Berlin, 1996.

[4] L. C. Paulson. *Isabelle: A Generic Theorem Prover*. Springer, Berlin, 1994.

[5] F. Pfenning. Elf: A language for logic definition and verified meta-programming. In *Proc. 4th IEEE Ann. Symp. Logic in Comp. Sci.* IEEE Computer Society Press, 1989.

5.1 General theory

David Basin and Sean Matthews

If we can see the machinery of a natural deduction system purely in terms of logical concepts, as we do in a ⇒ framework, then we can use various powerful, purely logical, tools in the work of searching for, and building, proofs. And this in turn can suggest new applications for the the theorem-proving system itself, for instance higher-order unification has been adopted by the Isabelle and ELF systems as a basic component, and been used by, among others, ourselves (see §5.2, §5.6, §5.3) to extend an original idea from Green [1] and provide a general framework for the synthesis of formally verified programs, by using a framework theorem prover to find a program by 'solving' a logical equation; i.e., given a specification Σ we try to find a program π such that

$$\Sigma \leftrightarrow \pi. \tag{1}$$

Using a theorem-prover based on ⇒ framework and provided with unification, all the machinery we need to do this is already available: if we take π to be a *meta*variable, and enter (1) as the 'theorem' we want to prove, then by a process of controlled backwards chaining through the rules π is gradually reified as required, so that at the end we have a *proof* that

$$\vdash \Sigma \leftrightarrow P. \tag{2}$$

where P is a ground term representing a program, and a solution for π in the theory. The idea is simple but powerful, and various applications are documented elsewhere here (e.g. §5.3, §5.2). Beyond the particular applications though, is the question of the general principle that is being exploited. We would like an explanation of how this machinery functions that is simpler than just 'higher-order unification in the metatheory'. The problem is that some parts of the machinery seem to be intrinsically metatheoretic: in particular the behavior of the unification procedure, which can have non-local effects on a proof tree (if a variable occurs in more than one part of an unfinished derivation).

We have developed a theory based on extension of a arbitrary first-order theory with second-order quantifiers, [2] so that it is possible to write (1) as a formula

$$\exists_2 \pi (\Sigma \leftrightarrow \pi)$$

unlike more usual higher-order extensions, however, our proposal is conservative, equally effective for intuitionistic and classical first-order logics and permits the recovery of witnesses for $\exists_2$, unlike more standard higher-order extensions which may be non-constructive, and are certainly non-conservative. We are able then to provide a analysis of the non-local behavior of the metavariable/unification mechanism, and some possible variations, for different subclasses of theories.

Another theoretical issue we have been addressing is the limitations of a $\Rightarrow$ framework for intentional, or modal, systems. Standard frameworks are closely tied to the the basic logics of mathematics and the implication connective $\rightarrow$, which encodes (via the deduction theorem) *truth consequence* (i.e. 'if A is true then B is true' is modeled as $T(A) \Rightarrow T(B)$). Many 'philosophical' logics, such as modal logics, however, are based not just on truth consequence but also *validity* ('A is a theorem'), and it is not possible to capture this directly in a $\Rightarrow$ framework. We have thus developed a conservative extension of a standard logical framework which includes a 'modal' or intentional connective $\sharp$ for validity which stands in the same relation to intensional connectives as $\Rightarrow$ does to $\rightarrow$ (we can, e.g., encode 'A is a theorem' as simply $\sharp T(A)$). [3] This provides a intuitive language, an extension of natural deduction, in which to work with a range of intensional logics; we have used it to provide encodings of a range of modal logics. Further, this purely 'proof-theoretic' approach to treating intensional connectives is in contrast to the semantically motivated approach that we have also studied (cf. §5.5).

References

[1] C. Green. Application of theorem proving to problem solving. In *Proc. Int. Joint Conf Artificial Intelligence*, pages 219–239, 1969.

- [2] D. A. Basin and S. Matthews. Adding metatheoretic facilities to first-order theories. *Journal of Logic and Computation*, 6(6):835–849, 1996.

- [3] S. Matthews. Extending a logical framework with a modal connective for validity. In *Proceedings of TACS'97*, Lecture Notes in Computer Science, Sendai, Japan, September 1997. Tohoku University, ACM SIGACT, the Information Processing Society of Japan, the Japan Society of Software Science, Association for Symbolic Logic, Springer.

5.2 Logic program derivation

Penny Anderson

Methodologies for deriving correct functional programs have been studied extensively. The development of similar techniques for programs has, in comparison, lagged behind. Preliminary work with David Basin [1] has investigated the adaptation of methods from functional program development to logic program development, using the Isabelle proof assistant. Using this method, we derived logic programs by successive instantiations of Isabelle metavariables during the development of their correctness proofs. This work gave several methods of faithfully representing logic programs in Isabelle/HOL: by inductively defined sets, well-founded sets, and logical equivalences involving inductive datatypes. We have extended of this approach to essentially nondeterministic logic programs. The major difficulty to be solved is the development of better rewrite rules for dealing with the predicates we use to represent logic programs.

As it stands, the work demonstrates several methodological points of interest. Since functions are easier to reason about in our framework than predicates, we can exploit this uniformity by freely mixing functions and predicates in our programs. We restrict only the end product of the synthesis to eliminate functions. Our previous work concerned only pure program transformation rules, that is, we did not include verification steps. This is not always a natural way to proceed. Our new work shows how to implement both transformation and verification steps, which fit together quite naturally in the theorem-proving framework of Isabelle. Typically, verification steps were used to 'condition' a program so that a given transformation could be applied to it. Another improvement over our previous work is the representation of specifications and programs as axioms, rather than assumptions. This removes some of the inessential complexity from the proofs. We are also able to explore some interesting aspects of the relationship between tail recursion and incrementality in logic programs.

References

- [1] P. Anderson and D. A. Basin. Deriving and applying logic program transformers. In *Algorithms, Concurrency and Knowledge (1995 Asian Computing Science Conference)*, volume 1023 of *Lecture Notes in Computer Science*, pages 301–318, Pathumthani, Thailand, December 1995. Springer.

5.3 Formal synthesis of hardware using higher-order unification

David Basin and Stefan Friedrich

Hanna, Daeche and Longley, in [3], propose an approach to circuit design, verification and combination, which they call *formal synthesis* where the concrete design of a circuit is developed in parallel with its verification relative to a specification. In [4] we have reconstructed and generalized this methodology using a combination of Isabelle's theory of higher-order logic and the higher-order unification provided there. This idea goes back to Green's use of first-order resolution to synthesize programs automatically during proofs [2]. Our interpretation simplifies and extends Formal Synthesis both conceptually and in implementation. It also supports integration of this development style with other proof-based synthesis methodologies and leads to techniques for developing new classes of circuits, e.g., recursive descriptions of parametric designs. We also exhibit that the kind of rules used for resolution-based development in the LAMBDA system [1] are compatible with our reinterpretation.

References

[1] S. Finn, M. P. Fourman, M. Francis, and R. Harris. Formal system design — interactive synthesis based on computer-assisted formal reasoning. In D. Claesen, editor, *IMEC-IFIP International Workshop on Applied Formal Methods for Correct VLSI Design*, volume 1, pages 97–110, Amsterdam, 1989. Elsevier Science publishers B.V. (North-Holland).

[2] C. Green. Application of theorem proving to problem solving. In *Proc. Int. Joint Conf Artificial Intelligence*, pages 219–239, 1969.

[3] F. K. Hanna, N. Daeche, and M. Longley. Formal synthesis of digital systems. In *IMEC-IFIP International Workshop on: Applied Formal Methods For Correct VLSI Design*, volume 2, pages 532–548, Leuven, Belgium, 1989.

• [4] D. A. Basin and S. Friedrich. Modeling a hardware synthesis methodology in isabelle. In J. von Wright, J. Grundy, and J. Harrison, editors, *Theorem Proving in Higher Order Logics. 9th International Conference, TPHOLs'96*, volume 1125 of *Lecture Notes in Computer Science*, pages 33–50, Turku, Finland, 1996. Turku Centre for Comput. Sci.; Res. Inst. Found. Abo Acad.; Acad. Finland, Springer.

5.4 Integrating decision procedures with interactive theorem provers

David Basin, Stefan Friedrich and Peter Leven

Although most nontrivial logics are undecidable, they typically have decidable sublogics. The efficiency of theorem proving can often be improved by integrating specialized proof procedures for decidable sublogics. There are two ways to do this: either the theorem prover can be extended by additional proof procedures (tactics) implementing a decision procedure, or the decision procedure can be implemented as an *external* theorem prover invoked as an oracle. We have investigated both approaches, exploiting, in the second, the facility for integrating oracles into the Isabelle system.

Integrating a locality-based proof-tactic in Isabelle

Our work is based on the research of Basin and Ganzinger [3] (see §4.1), who define a notion of order locality, characterizing a class of decidable first order theories, for which the entailment problem of ground clauses is decidable. A clause set N when a proof of $N \models A$ for ground A can be built using clauses from N instantiated only with terms from some finite set $(A)*$.

In our work we have develop proof tactics for entailment problems in local Horn theories. Basin and Ganzinger, in their theoretical analysis, give upper complexity bounds for entailment proofs in local theories. We have improved the (average-case) efficiency of the procedure in practice by exploiting work done in the areas of deductive databases and logic programming, where efficient evaluation procedures have been developed. We use goal-oriented forward-chaining inference, motivated by the analysis of definite Horn clause program evaluation in [1]. We restrict resolution by selecting antecedents atoms and forbidding resolution on non selected atoms. We are able to use 'generators' for lazy enumerating partial instantiated clause instances involved in the decision procedure.

Our tactics are implemented in Isabelle's meta-logic, so we can benefit of Isabelle's generic support of object logics: the tactics can be used for theorem-proving in the different object logics without change. Finally, the tactics we have developed are not restricted to local theories: even in the non-local case the tactic can be used to simplify the proof goal before a stronger but less efficient tactic is applied.

Integration of a decision procedure for second-order monadic logic in Isabelle/HOL

Monadic second-order theories with finite number of successors has been proved decidable in the 1960s. Although the complexity of the decision procedure is non-elementary it behaves well in many cases. The Mona tool [2], which has been successfully applied to hardware verification and arithmetic, provides a fast BDD-based implementation of the method. In our work we have attached it, as an 'oracle', to an implementation of higher-order logic (HOL) in Isabelle. Our interest in this work was twofold. We wanted to investigate the way such a tool should be attached to an interactive theorem prover and, secondly how a the user can take advantage of the benefits it offers. We have constructed an interface which translates suitable formulae from the language of Isabelle's HOL into the language of Mona and and then uses Mona to prove (or disprove) the result. To this end we formalized the semantics of the Mona system as a theory of Isabelle's HOL. The interface is packaged as an Isabelle tactic.

The other part of our research investigates how the decision procedure can be used in the HOL logic. We have developed techniques which allow a user to decide the validity of sentences of Isabelle's standard theories of arithmetic within the natural numbers and the integers. Further we have implemented examples taken from hardware verification, where the specifications to be verified are beyond the expressiveness of second-order monadic logic. We showed that it is possible to decompose these specifications in subparts in which we can can exploit our decision procedure.

References

[1] R. Bol and L. Degerstedt. The underlying search for magic templates and tabulation. In D. S. Warren, editor, *Proc. 10th Int Conf. Logic Programming.* MIT Press, Boston, 1993.

[2] J. Henriksen, J. Jensen, M. Jørgensen, N. Klarlund, B. Paige, T. Rauhe, and A. Sandholm. Mona: Monadic second-order logic in practice. In B. Steffen, editor, *Tools and Algorithms for the Construction and Analysis of Systems, First International Workshop.* Springer, Berlin, 1996.

• [3] D. A. Basin and H. Ganzinger. Complexity analysis based on ordered resolution. In *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS'96)*, pages 456–465, New Brunswick, New Jersey, USA, 1996. IEEE Comput. Soc. Tech. Committee on Math. Found. Comput.; ACM Special Interest Group on Automata & Comput. Theory; Assoc. Symbolic Logic; Eur. Assoc. Theoret. Comput. Sci, IEEE Comput. Soc. Press.

5.5 Labelled proof systems for non-classical logics

David Basin, Seán Matthews and Luca Viganò

So long as we are interested only in implementing classical or intuitionistic, first-order or higher-order logic, we should have no problems giving a direct encoding in a $\Rightarrow$ framework. However there are many 'badly behaved' logics that we might also want to implement; e.g. if we are interested in modal logics we find that because the deduction theorem fails, a naïve natural deduction presentation is impossible; if we are interested in substructural logics, then thinning or contraction fail, so we have to modify the nature of natural deduction substantially. We might even be interested in non-monotonic logic. The standard logical frameworks have difficulty dealing with all these.

We have given uniform presentations of various non-classical logics (including modal, relevance and other substructural logics) as uniform natural deduction proof systems for which we can prove general metatheorems (such as soundness and completeness with respect to the intended semantics or proof normalization). To this end, we have taken the view of a logic as a *Labelled deductive system* (LDS), as proposed by Gabbay [9]. In this approach, logics are presented by associating formulae with labels generated by some algebra, possibly the Kripke semantics of the logic, and formal deductions are performed on the pair of the formula and the label together. The information in the labels can be used in the application of rules, and turns out to be enough to enforce good behavior on previously badly behaved logics, which can now be implemented in standard frameworks like Isabelle (the 'bad' behavior of the logic is captured in the algebra defined over the labels, but derivations in the algebra itself are carried out using ordinary logic, which there is no problem implementing). Further, related logics are formalized only by changes in the algebra of the labels, so that metatheorems can be proved in a modular way.

We have been looking at how practical it is to use labelled natural deduction to implement various non-classical logics in the Isabelle Logical Framework, and, as a first case study, we have focussed on modal logics [3, 4, 2, 6]. Our experience has been very positive: we have been able to implement many standard propositional modal logics in uniform manner, parameterized only over the properties of the labelling algebra, which captures the accessibility relation of the corresponding Kripke semantics. This allows us to present a large class of propositional modal logics in a structured manner where logics naturally inherit theorems from their ancestors. Equally importantly, we have found that labelled natural deduction implementations of logics are very intuitive to use. While we are not the first to provide a modal logic presentation in a logical framework (Avron et al. [1] and Coen, among others, precede us) ours is the first to provide and implement a general system, rather than a few special cases.

We have extended this prototypical theoretical and practical work in four directions. First, we have considered the more difficult case of quantified modal logics [5], which raise difficulties not present in the propositional case. The difficulties arise because of additional freedom in choosing a semantics, and therefore a logic, appropriate for an intended application; these choices can be made independently of each other and the result is a multi-dimensional landscape of logics. In the past, this landscape has been considered in a piecemeal fashion and there has been a lack of uniformity in the presentation of proof systems and how metatheoretic results are proved. Our approach, on the other hand, yields implementations of quantified modal logics that are modular in several dimensions, reflecting the degrees of freedom previously described.

Second, we have formalized both partial and complete fragments of a wide range of non-classical logics, including in particular relevance logics, which are an active area of research in artificial intelligence and knowledge representation [4, 7]. In this case, our approach is based on an abstract classification of classical and non-classical connectives, including modalities, substructural implication, and various types of negation. By appropriate combinations of these connectives, we obtain uniform labelled natural deduction proof systems for several classes of logics. As for modal logics, the result is modular in the way the systems are formalized and implemented, as well as how general metatheoretic results are proved.

Third, we have presented a new proof-theoretic approach to bounding the complexity of the decision problem for propositional modal logics [8]. Based on the corresponding labelled natural deduction systems, we have presented logics in a uniform way as cut-free labelled sequent systems, and showed that an analysis of the structural rules for particular systems, combined with an analysis for the accessibility relation of the corresponding Kripke structures, yields restricted systems and decision procedures with space requirements that are easily bounded. While the space bounds we arrive at are not always new (similar bounds for modal logics were recently discovered, using different, non-modular techniques, by Hudelmaier), this work provides a first step towards a general framework for applying proof theory to the analysis of complexity of families of non-classical logics and for implementing decision procedures for these logics.

Finally, we have investigated the tradeoffs and limits of our approach: we have shown that variations in the algebra allow us to formalize 'paraconsistent' versions of non-classical logics or to obtain proof systems that are essentially equivalent to the ones resulting from standard *semantic embeddings* (i.e. translation of non-classical logics into first-order logic).

We are currently investigating how well our approach applies to other logics from the theoretical and practical point of view. We are also looking at the issues in the second part of Gabbay's proposal: that the LDS approach is an effective foundation for developing technology for combining different logics together.

References

[1] A. Avron, F. Honsell, I. Mason, and R. Pollack. Using typed lambda calculus to implement formal systems on a machine. *J. Auto. Reas.*, 9:309–352, 1992.

• [2] D. Basin, S. Matthews, and L. Viganò. a topography of labelled modal logics. In F. Baader and K. U. Schulz, editors, *Proc. Frontiers of Combining Systems.* Kluwer, Dordrecht, 1996.

• [3] D. Basin, S. Matthews, and L. Viganò. a modular presentation of modal logics in a logical framework. In *Proc. Tbilisi Symp. on Language, Logic and Computation.* CSLI, Stanford, to appear.

• [4] D. A. Basin, S. Matthews, and L. Viganò. Implementing modal and relevance logics in a logical framework. In L. Carlucci-Aiello, J. Doyle, and S. Shapiro, editors, *Proceedings of the Fifth International Conference on Principles of Knowledge Representation and Reasoning (KR'96)*, pages 386–397, Cambridge, MA, USA, 1996. Morgan Kaufmann.

• [5] D. A. Basin, S. Matthews, and L. Viganò. Labelled modal logics: quantifiers. *Journal of Logic, Language and Information*, 1997.

• [6] D. A. Basin, S. Matthews, and L. Viganò. Labelled propositional modal logics: Theory and practice. *Journal of Logic and Computation*, 7, ? 1997.

• [7] D. A. Basin, S. Matthews, and L. Viganò. Natural deduction for non-classical logics. *Studia Logica*, 1997. To appear.

• [8] D. A. Basin, S. Matthews, and L. Viganò. A new method for bounding the complexity of modal logics. In A. Leitsch, editor, *Proceedings of the 5th Kurt Gödel Colloquium on Computational Logic and Proof Theory (KGC'97)*, Lecture Notes in Computer Science, Vienna, 1997. Springer.

• [9] D. Gabbay. *Labelled Deductive Systems, vol. 1.* Clarendon, Oxford, 1996.

5.6 Inductive definitions

David Basin and Sean Matthews

While ⇒ frameworks are demonstrably very effective for a wide range of applications, they have one drawback: they do not support general inductive metatheoretic extension, i.e. 'proof theory'; this is precisely the price we pay for an elegant and usable abstraction of natural deduction style proof. For instance, we can easily encode minimal implication as a natural deduction calculus or as a Hilbert calculus in an ⇒ framework. We cannot however go on to show that these two systems define the same sets of theorems (i.e. prove the deduction theorem). Thus theories of inductive definition have been proposed as a possible alternative to ⇒ frameworks, for circumstances where we might want to do metatheoretic analysis, or need metatheoretic extension. Specifically, Feferman has proposed FS_0 [1] a theory of finitary inductive definitions, as a possible candidate.

We have developed a full implementation of FS_0 in a (slightly modified version of) Isabelle [4] which we have used as a tool for exploring applications. The system is effective enough that we have been able to carry out the investigations we describe below, as well as some other large developments, including an implementation of Talcott's algebra of binding [5].

The first area we have explored is the problem of theory structuring in a framework. This is something that we seem to have to trade off directly against access to general metatheory. In an ⇒ framework it is not a problem precisely because we do not there have the possibility of general metatheory: any logic is implemented simply as a theory in ⇒. We can specify a new logic simply as an extension of an old one which automatically, and safely, inherits everything previously provable (in the same way that all the theorems of any theory remain theorems under new axioms). This makes for a simple hierarchical structuring that satisfies classical rules of software engineering.

For general metatheory, on the other hand, we must be able to assume that we have *precisely* such-and-such rules and such-and-such axioms; i.e. that theories are closed. This is precisely what ID frameworks provide, but means that we cannot expand the definition of a theory so naturally. At first sight it seems that any sort of hierarchical structuring mechanism is ruled out in an ID framework, however in practice not all inductive arguments depend on *all* these assumptions (the deduction theorem, for instance, does not assume that the class of axioms is closed, only that the class of rules is). In [2] we show that in a language like FS_0 we state explicitly the assumptions we need for a particular metatheorem, and thus get back the general structuring facilities that are available in a $\Rightarrow$ framework *where applicable* while also allowing general metatheory.

We have also been considering the problem of how to implement a proof system for a logic so that it is practically usable for both object theory and metatheory. The 'natural' style of proof system to implement as an inductive definition is a Hilbert system, but it is universally recognized that such a presentation is difficult to work with for many metatheoretic purposes, and practically impossible to use for proving theorems. On the other hand the traditional encodings of 'better' proof systems, e.g. natural deduction, that we find in proof theory text books involve enormous overhead compared to encodings of the same systems in $\Rightarrow$ frameworks. We have developed and implemented an alternative to this 'textbook' encoding [3] which exploits combination of metatheory and the insights of $\Rightarrow$ frameworks to provide an encoding which is more directly competitive with a $\Rightarrow$ framework.

References

[1] S. Feferman. Finitary inductive systems. In *Logic Colloquium '88*. North-Holland, Amsterdam, 1990.

• [2] D. A. Basin and S. Matthews. Adding metatheoretic facilities to first-order theories. *Journal of Logic and Computation*, 6(6):835–849, 1996.

• [3] S. Matthews. A practical implementation of simple consequence relations using inductive definitions. In W. McCune, editor, *Proceedings of CADE'14*, Lecture Notes in Artificial Intelligence, Townsville, Australia, July 1997. Association for Automated Reasoning, Springer.

• [4] Seán Matthews. Implementing FS_0 in Isabelle: adding structure at the metalevel. In J. Calmet and C. Limongelli, editors, *Proc. Disco'96*. Springer, Berlin, 1996.

[5] C. Talcott. A theory of binding structures, and applications to rewriting. *Theoret. Comp. Sci.*, 112:99–143, 1993.

5.7 Applying algebraic specification techniques to the specification of dynamic systems

Hubert Baumeister

One way to model the state space of a dynamic system is as an abstract datatype. Each model of the abstract datatype is an admissible state of the dynamic system. The theory of abstract datatypes is well developed, has a rigorous formal basis and is institution independent, i.e. independent of the particular logic used for defining abstract datatypes, like, e.g., many sorted equational logic, or order sorted first order logic with partial functions.

The problem is how to define the state transitions of a dynamic system. Several approaches are known in the literature, most of them fix the institution used for describing the state space then provide a separate logic for the state transition.

In [1] we have defined, based on a given institution describing the state space of a dynamic system, a new institution, where abstract datatypes are interpreted as relations on models of abstract datatypes from the base institution. The advantage of this approach is that the institution independent part of the theory of abstract datatypes can be used to define the behaviour of dynamic systems.

However, because of the lack of tools for the new institution, it is in practice not convenient to define abstract datatypes in this new institution. We have thus developed in [2] a way to represent abstract datatypes denoting relations again as abstract datatypes of the institution of the state space.

The key idea is to exploit the isomorphism between pairs of models (m_1, m_2), where m_1 is a model of abstract datatype A and m_2 is a model of abstract datatype B, such that m_1 and m_2 coincide on the intersection of the signatures of A and B, and models m in the union of A and B. This isomorphism exists in any institution that has amalgamation and permits the definition of relations $R \subseteq \texttt{Mod}(A) \times \texttt{Mod}(B)$ by abstract datatypes $(\texttt{Sig}(A) \cup \texttt{Sig}(B), M)$.

This leads to specifications of dynamic systems similar in style to Z specifications for the same kind of systems. However, Z specifications of dynamic systems are based on set theory for specifying the state space and state transitions, while the method we propose can be used with any institution that has amalgamation. In particular, we have used the Larch Shared Language, a specification language based on many sorted equational logic, to define dynamic systems. The Larch Shared Language Checker and the Larch Prover were then used to check for wellformedness of specifications and to prove properties about the behaviour of dynamic systems.

References

[1] H. Baumeister. Relations as abstract datatypes: An institution to specify relations between algebras. In P. D. Mosses, M. Nielsen, and M. I. Schwartzbach, editors, *Proceedings of the Sixth Joint Conference on Theory and Practice of Software Development (TAPSOFT '95)*, volume 915 of *Lecture Notes in Computer Science*, pages 756–771, rhus, Denmark, May 1995. Springer.

• [2] H. Baumeister. Using algebraic specification languages for model-oriented specifications. Research Report MPI-I-96-2-003, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, February 1996.

6 Extended modal logic and automated theorem proving

We aim at developing and studying modal logics, both propositional and quantified, as well as extensions with strong practical motivations, and making available first-order resolution theorem provers for automated modal reasoning and automated solving of the correspondence problem.

Propositional modal and temporal logics are increasingly being used in various fields of computer science, including knowledge representation, the field of logics of programs and computational linguistics. Their popularity can be attributed to their incredibly simple syntax, their natural possible worlds semantics and decidability. Despite the pleasant properties of modal logics a serious problem in real applications is the lack of sophisticated modal theorem provers. Some theorem provers are available that are based on Gentzen or tableaux calculi, of which none have the level of sophistication that first-order theorem provers like OTTER or SPASS (see §9.1) have. Many propositional modal logics are decidable, but unrestricted resolution is merely a semi-decision procedure, so an important and non-trivial problem is finding suitable translation methods and resolution calculi that are decision procedures. The standard semantics based translation approach of modal formulae and entire logics is based on the possible worlds semantics. It transforms modal formulae to restricted quantifier expressions involving binary relations. The problem with this relational translation approach is the inherent non-determinism in relations, manifested in the non-termination of standard resolution procedures (and complications in other theorem proving approaches).

We study the alternative translation approaches due to H. J. Ohlbach and A. Nonnengart, discussed in §6.1, that are based on a functional alternative view of the semantics of modal logic that solves the problem of non-determinism. In the functional semantics accessibility is defined by sets of functions replacing binary relations. This reformulation has computational advantages that will be reported on in §6.2 and §6.3 which are devoted to decidability by the resolution method and empirical performance analysis.

In automated theorem proving it is not only of importance to develop sophisticated reasoning techniques, also the systematic derivation of background theories and suitable simplifications is valuable. A tool that has become indispensable in our work is SCAN, which we employ mostly for automatically computing the relational and functional correspondence properties for modal axioms that form the modal background theory. SCAN is a general tool that computes for given second-order quantified formulae equivalent first-order formulae. §6.4 describes a non-trivial application for solving correspondence problems in conditional logics and discusses an alternative method using a fixpoint approach. §6.5 describes the work on simplifying modal background theories.

In computer science applications it is very often useful to enhance the expressive power of basic propositional modal logic, for example with temporal modalities like Until, Next, etc., numerical quantifiers like graded/counting modalities, or parameterized modalities of type available in propositional dynamic logic. Many of these enhancements can be accommodated in hybrid logics described in §6.6.

6.1 Translations from modal and temporal logic into first-order logic

Hans Jürgen Ohlbach, Renate A. Schmidt and Andreas Nonnengart

The functional translation method of quantified modal logic into first-order logic was put forward in the late eighties by H. J. Ohlbach [8, 9] and others. The original motivation still holds. We want to integrate modal and temporal logics into first-order logic so as to exploit first-order inference tools, while preserving the characteristics and advantages of the original logics.

In our recent work we have been focusing on propositional modal logics. The fundamental idea of the functional translation is that of decomposing any binary (accessibility) relation into a set of functions and a set identifying the dead ends of the relation. This allows a translation of any first-order definable modal logic into first-order logic. Some modal logics are inherently second-order and do not reduce to first-order logic by the relational translation. However, we have studied an optimization of the functional translation that was known to be applicable only to propositional formulae and found that some truly second-order modal logics have first-order functional characterizations [10, 4]. The optimized functional translation method uses an operation by which existential and universal quantifiers are swapped in a way that eliminates non-constant Skolem terms from the clausal forms. Semantically, this optimization is embodied in special maximal or patched functional models, with respect to which we prove the preservation of satisfiability from which the completeness of our method follows [4]. The practical implication is that first-order inference techniques become available for second-order logics, like K with McKinsey's axiom and the new modal logic $\overline{K}_E$.

$\overline{K}_E$ is a multi-modal logic introduced in [6] that approximates the graded modal logic $\overline{K}$, and serves as an intermediary logic for realizing reasoning about numerical quantifiers in a first-order resolution calculus enhanced with simple arithmetic. This work is an involved application of the translation techniques and the tools (namely SCAN) we use. It demonstrates how reasoning about the cardinality of sets can be facilitated by resolution in combination with additional routines for doing simple arithmetic and has applications in knowledge representation [5].

In joint work H. J. Ohlbach and J. Koehler have been exploring a different and completely new approach to reasoning about set cardinality information [3]. The approach uses a technique of separating a collection of sets into smallest mutually disjoint components (called atoms) such that the cardinality of the sets are just the sum of the cardinalities of their atoms. With this idea of atomic decomposition, it is possible to have languages combining arithmetic formulae with set terms and to translate the formulae of this combined logic into pure arithmetical formulae.

The functional translation captures extensions of the basic modal logic by a theory comprising clauses that involve equations. Inference requires theory resolution with which we have limited practical experience. Finding appropriate and efficient theory unification and resolution procedure deserves further investigation. The lack of implementations of theory resolution was one motivation for A. Nonnengart [7, 1, 2] proposing the semi-functional translation approach. It combines the relational and the functional translation approach in that the $\Diamond$ operator is translated functionally whereas the $\Box$ operator is translated relationally. When the relational correspondence properties do not involve equations then the defining semi-functional theories do not either.

A major advantage of the semi-functional translation is that the modal logic background theory is separated from the theorem to be proved. This makes saturation and thus simplification of the modal background theories possible. Moreover, it enables us to replace entire background theories by simple theory rules.

References

- [1] A. Nonnengart. *A Resolution-Based Calculus for Temporal Logics.* PhD thesis, Universität des Saarlandes, December 1995.
- [2] A. Nonnengart. Resolution-based calculi for modal and temporal logics. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 598–612, New Brunswick, USA, 1996. CADE Incorporated, Springer.
- [3] H. J. Ohlbach and J. Koehler. Reasoning about sets via atomic decomposition. Technical Report TR-96-31, International Computer Science Institute, Berkeley, CA, 1996. Submitted to the *AI Journal*.
- [4] H. J. Ohlbach and R. A. Schmidt. Functional translation and second-order frame properties of modal logics. To appear in *Journal of Logic and Computation*. Available as Research Report MPI-I-95-2-002, 1997.
- [5] H. J. Ohlbach, R. A. Schmidt, and U. Hustadt. Symbolic arithmetical reasoning with qualified number restrictions. In A. Borgida, M. Lenzerini, D. Nardi, and B. Nebel, editors, *Proceedings of International Workshop on Description Logics'95*, number 07.95 in Rap., pages 89–95, Rome, Italy, May 1995. Dipartimento di Informatica e Sistemistica, Univ. degli studia di Roma.
- [6] H. J. Ohlbach, R. A. Schmidt, and U. Hustadt. Translating graded modalities into predicate logic. In H. Wansing, editor, *Proof Theory of Modal Logic*, Studies in Applied Logic Series, pages 253–291. Kluwer, Dordrecht, The Netherlands, 1996. Also available as Research Report MPI-I-95-2-008.

[7] A. Nonnengart. First-order modal logic theorem proving and functional simulation. In R. Bajcsy, editor, *Proc. 13th Intern. Joint Conference on Artificial Intelligence (IJCAI '93)*, volume 1, pages 80–85, Chambery, France, 1993. Morgan Kaufmann.

[8] H. J. Ohlbach. *A Resolution Calculus for Modal Logics.* PhD thesis, Universität Kaiserslautern, 1988.

[9] H. J. Ohlbach. Semantics based translation methods for modal logics. *Journal of Logic and Computation*, 1(5):691–746, 1991.

[10] H. J. Ohlbach. Optimized translation of multi modal logic into predicate logic. In A. Voronkov, editor, *Proceedings of the 4th International Conference on Logic Programming and Automated Reasoning (LPAR'93)*, volume 698 of *Lecture Notes in Artificial Intelligence*, pages 253–264, St. Petersburg, Russia, 1993. Springer.

6.2 Decidable modal inference by resolution

Renate A. Schmidt and Ullrich Hustadt

Many attempts at using resolution theorem provers for modal reasoning have been frustrated by the fact that resolution is only a semi-decision procedure for first-order logic. Without any heuristics resolution continually produces more and more seemingly non-redundant clauses without terminating, not even for decidable fragments of first-order logic that correspond to decidable propositional modal logics.

R. A. Schmidt has studied the functional translation approach, and its optimization [1], and exhibits what has often been claimed for the method but never substantiated: For many propositional modal logics unrestricted resolution decides modal satisfiability and theoremhood for the optimized functional translation [2]. Important for decidability is that we use an optimized version of the functional translation approach. It eliminates in the clausal forms all Skolem functions other than Skolem constants, and embeds modal logics in a hierarchy of clausal logics, called path logics (described in §4.2). The result is very powerful for a number of reasons. It immediately follows that standard unrestricted resolution is a decision procedure for many non-transitive uni- or multi-modal logics, including K and arbitrary extensions with D, T and B, as well as their multi-modal versions, and also *S5*. The decision result provides an alternative decision proof for the relevant modal logics, and related logics used in applications, like description logics in knowledge representation. This alone is not very exciting. A more far-reaching consequence of the result has practical value, namely, any standard first-order theorem prover that is based on resolution can serve as a reasonable and efficient inference tool for modal reasoning. It is not our intention to pursue the use of unrestricted resolution, but the decision result gives us complete freedom to fine tune our theorem prover with any modifications and refinements that are compatible with resolution so as to increase efficiency. We have encouraging benchmark results with ordered resolution that will be discussed in the next section.

As elaborated in §4.2, ordered resolution is a refinement that has proved valuable in providing decision procedures for many subclasses of first-order logic. The work of U. Hustadt deals with ordered resolution for Maslov's class K and addresses the problem of whether there are ordered resolution procedures that decide satisfiability for the semi-functional translation approach. He shows there are ordering restrictions and optimizations that render decision procedures for the modal logics K, *KD*, *KT*, *KDB*, *S5*, *KD5* and *KD45*. In particular, the background theories of the first five modal logics belong to the class K. The definitional form of the semi-functional translation of any modal formula belongs to the class K as well. Consequently, any decision procedure for the class K is also a decision procedure for the semi-functional translation of the relevant modal logics. The background theories of *KD5* and *KD45* consist of unit clauses that do not belong to the class K. We can saturate the semi-functional translation of a modal formula by unit propagation with the background theories that gives a set of clauses that belongs to the class K.

References

- [1] H. J. Ohlbach and R. A. Schmidt. Functional translation and second-order frame properties of modal logics. To appear in *Journal of Logic and Computation.* Available as Research Report MPI-I-95-2-002, 1997.
- [2] R. A. Schmidt. Resolution is a decision procedure for many propositional modal logics: Extended abstract. To appear in Kracht, M., de Rijke, M., Wansing, H. and Zakharyaschev, M. (eds), *Advances in Modal Logic '96.* CSLI Publications. The long version is available as Research Report MPI-I-97-2-002, 1997.

6.3 Performance analysis of different modal theorem provers

Ullrich Hustadt and Renate A. Schmidt

The decidability result of R. A. Schmidt described above, and the strong claims of Giunchiglia and Sebastiani [1, 2] initiated the experimental comparison of the performances of SPASS (described in §9.1) with other modal theorem provers that are available publicly. Performance comparison of theorem provers for modal logics and related description logics arising in the field of knowledge representation are very rare. Usually, the literature on modal theorem provers confines itself to a description of the underlying calculus and methodology accompanied with a consideration of the worst-case complexity of the resulting algorithm. Sometimes a small collection of benchmarks is given. The first extensive performance analysis was done by Giunchiglia and Sebastiani, who compare the tableaux system KRIS and a new theorem prover, called KSAT, on a benchmark suite of randomly generated multi-modal formulae in a certain normal form. KSAT is an adaptation for the multi-modal logic $K_{(m)}$ of a SAT-procedure for checking satisfiability in propositional logic.

In [4] we show that the evaluation of Giunchiglia and Sebastiani has some shortcomings. The random generator used to set up a benchmark suite produces formulae containing a substantial amount of tautological and contradictory subformulae. It favors the SAT-procedure KSAT which utilizes a preprocessing routine that eliminates trivial tautologies and contradictions from the formulae. Since KRIS is a system primarily designed for knowledge representation, an application domain where simple tautological and contradictory formulae seldomly occur, we think it is fair to add preprocessing to KRIS Having done so our performance comparison renders a more accurate picture that is completely different from that of Giunchiglia and Sebastiani who claimed that KSAT outperforms KRIS by orders of magnitude. In addition, we have refuted the conjecture of an easy-hard-easy pattern on randomly generated modal logic formulae. The theorem prover SPASS does not show the peaking behavior of KSAT for the optimized functional translations nor for the semi-functional translations. Instead, the median CPU time grows monotonically with the size of modal formulae.

Let alone the fact that the randomly generated formulae contain a substantial amount of tautological and contradictory subformulae, we show that they do not provide challenging unsatisfiable problems. So as to obtain harder problems we develop guidelines for random generation of modal formulae [3]. We present a set of samples of modal formulae generated according to these guidelines and verify that they provide challenging problems for KSAT and the translation approach. We show that the translation approach in combination with the theorem prover SPASS has better computational behavior than KSAT, KRIS, and the Logics Workbench on all samples of randomly generated modal 3CNF formulae we have considered, except for the samples of very small formulae. The reason, we believe, is a fundamental difference in method. The translation approach adopts a global method, whereas KSAT and tableaux or sequent-based systems adopt a local method. Local approaches are based on the internal view of Kripke semantics that reduces truth to truth in worlds (on the element level) and the derivation tree closely resembles the underlying frame. The systems perform a depth-first search through the derivation tree. The global method of the translation approach transfers the underlying tree structure from the meta level to the object level in the first-order translation of the given modal formula. The theorem prover reasons about the structure of the tree on the object level, and is not restricted to a depth-first search.

References

[1] F. Giunchiglia and R. Sebastiani. Building decision procedures for modal logics from propositional decision procedures: Case study of modal K. In M. A. McRobbie and J. K. Slaney, editors, *Automated Deduction: Proc. CADE-13*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 583–597. Springer, 1996.

[2] F. Giunchiglia and R. Sebastiani. A SAT-based decision procedure for $\mathcal{ALC}$. In L. C. Aiello, J. Doyle, and S. Shapiro, editors, *Proceedings of the 5th International Conference on Principles of Knowledge Representation and Reasoning (KR'96)*, pages 304–314, Cambridge, MA, USA, 1996.

- [3] U. Hustadt and R. A. Schmidt. On evaluating decision procedures for modal logic. Research Report MPI-I-97-2-003, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, February 1997.

- [4] U. Hustadt and R. A. Schmidt. On evaluating decision procedures for modal logics. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, Nagoya, Japan, 1997. International Joint Conferences on Artificial Intelligence, Inc. (IJCAII) and Japanese Society for Artificial Intelligence (JSAI), Morgan Kaufmann. To appear.

6.4 Quantifier elimination

Hans Jürgen Ohlbach, Andreas Nonnengart and Thorsten Engel

There are many research areas in which the transformation of second-order formulae to logically equivalent first-order formulae (provided they exist) is essential. Typical examples are set theory, modal and temporal logic correspondence theory, modal frame characterizations, and circumscription. The elimination of second-order predicate quantifiers can be automated to a large extent. One approach uses the SCAN algorithm [2, 1, 7, 3] that T. Engel integrated into the theorem prover OTTER. In joint work [6] with A. Szałas, A. Nonnengart proposes a fixpoint approach based in part on the work of Ackermann. The paper [5] surveys both approaches.

A. Herzig [4] studies how SCAN can be applied to a large family of systems of conditional logic, which includes most of the systems known from the literature. The semantics of conditional systems is more involved than the semantics of normal modal logics. With SCAN we were able to confirm some of the most important correspondence properties, but for some axioms the algorithm loops. This cannot happen with the alternative fixpoint approach, as it is guaranteed to terminate. Its implementation is currently in progress.

References

[1] C. Brink, D. M. Gabbay, and H. J. Ohlbach. Towards automating duality. *Journal of Computers and Mathematics with Applications*, 29(2):73–90, 1994. Special Issue on Automated Reasoning. Also available as Research Report MPI-I-93-220, Saarbrücken.

[2] D. M. Gabbay and H. J. Ohlbach. Quantifier elimination in second-order predicate logic. *South African Computer Journal*, 7:35–43, 1992. Also appeared in Proc. 3rd Intern. Conference on Principles of Knowledge Representation and Reasoning, editors: B. Nebel, C. Rich and W. Swartout, pp. 425-435, Morgan Kaufmann, 1992. Also available as Research Report MPI-I-92-213.

- [3] T. Engel. Quantifier elimination in second-order predicate logic. Master's thesis, Universität des Saarlandes, 1996.
- [4] A. Herzig. Scan and systems of condtional logic. Research Report MPI-I-96-2-007, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, September 1996.
- [5] A. Nonnengart, H. J. Ohlbach, and A. Szalas. Quantifier elimination for second-order predicate logic. To appear in *Logic, Language and Reasoning. Essays in honour of Dov Gabbay, Part I, Kluwer Academic Press*, 1997.
- [6] A. Nonnengart and A. Szalas. A fixpoint approach to second-order quantifier elimination with applications to correspondence theory. To appear in *Essays in Memoriam of Helena Rasiowa*, 1997.
- [7] H. J. Ohlbach. SCAN-elimination of predicate quantifiers. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 161–165, New Brunswick, New Jersey, USA, 1996. AT&T, Bell Labs, Lucent, IEEE, CADE inc, Springer.

6.5 Modal frame characterizations

Andreas Nonnengart

For model-theoretic analysis or theorem proving it important to choose a semantic characterization that is most appropriate. This is usually rather complicated, however, and, up to now, possible only at pure model-theoretic level.

A. Nonnengart [1] has investigateed a new and semi-automatic approach to finding stronger characterizations. The key idea is based on certain conservative extensions of the logic under consideration such that formulae of the original language are valid in the extended logic if and only if they are valid in the original logic. The extension has to be defined in a way such that all new axioms are in a form characterized by Sahlqvist [2].

These formulae have the advantage that their second-order correspondences reflect exactly the corresponding frame properties (which is not always the case for other kinds of formulae). With these formulae we are able to exploit automated predicate quantifier elimination play (discussed in the previous section). This can be used, first, to find the first-order correspondence of the various axioms under consideration, and second, after all correspondences have been found, to eliminate all the information that was interesting only for the extension but not for the original logic. This often results in a characterization which is more general than the one that was already known. But it can also be used to simplify the task of finding characterizations. As an example consider the modal logic *S4.1*. Its frames are determined by reflexivity, transitivity, and the McKinsey-property which is a complicated second-order property. It is known that this logic is characterized by the class of reflexive, transitive and atomic frames, but showing this model-theoretically requires a very sophisticated machinery. With the techniques we have developed this result can be obtained quite easily in a semi-automatic way.

The approach not only enables us to confirm well-known results; we have also obtained some new ones. For example, for dense extensions of the logic *KD4.3* we show that we may assume that irreflexive worlds are immediately followed by clusters of worlds, a property which implies (but is not implied by) density. The quality of the results obtained with the help of our approach depends very much on the possibilities of eliminating predicate quantifiers. Improvements of quantifier elimination techniques are very likely to bear more and stronger characterizations for modal logics.

References

- [1] A. Nonnengart. Auxiliary modal operators and the characterization of modal frames. Research Report MPI-I-96-2-005, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, June 1996.

[2] H. Sahlqvist. Completeness and correspondence in the first and second order semantics for modal logic. In S. Kanger, editor, *Proceedings of the 3rd Scandinavian Logic Symposium*, pages 110–143. North Holland, 1975.

6.6 Hybrid logics

Miroslava Tzakova

The expressivity of the basic modal language is comparatively limited, and must typically be boosted by the addition of various (application dependent) new modalities, such as the Until operator. In joint work with P. Blackburn, we investigate an alternative strategy for boosting modal expressive power: hybridization.

Hybridization is an attempt to combine the key semantic idea of modal logic (namely Kripke semantics, the local evaluation of formulae at states inside models) with the key syntactic idea of first-order logic (namely the use of explicit variables and quantifiers which bind them). To this end, we enrich the basic modal language with a new sort of variables that range across states in Kripke models, as well as with quantifiers that allow us to bind variables of this new type. The resulting languages are called hybrid languages. They possess great expressive power, for example can define the Until operator (therefore capture the expressivity of the linear temporal logic) and counting modalities (like 'there are at least two successors'), and express irreflexivity (which is a first-order property that is not modally definable).

The literature (for example [5, 1, 4]) discusses mostly languages in which unrestricted existential quantification over states is possible. In our view, (local) quantification which is restricted to states reachable (via the relation in Kripke models) in finitely many steps blends naturally with the locality intuitions underlying Kripke semantics. In [2] we present complete axiomatizations of hybrid logics of various local quantifiers. We also show that while hybrid logics of arbitrary Kripke models are undecidable, by suitable restrictions of the class of models (for example, to initial fragments of Rabin's tree) they become decidable.

Moreover, by further hybridization we can also cope with branching time. For this, in [3] we consider the hybrid language of a single quantifier which binds state variables to unique successor states. We introduce into this language a third sort of variables that range over paths in the models and a quantifier which binds path variables to paths through the current state (of evaluation). This enriched language captures the expressivity of the computational tree logic and is decidable over the class of models that are initial fragments of Rabin's tree. We present a complete axiomatization of the hybrid logic of certain treelike models. Thus, hybridization gives us a uniform approach to increased expressivity.

References

[1] P. Blackburn and J. Seligman. Hybrid languages. *Journal of Logic, Language and Information*, 4:251–272, 1995.

[2] P. Blackburn and M. Tzakova. Hybrid logic. Submitted for publication, 1997.

[3] P. Blackburn and M. Tzakova. A hybrid logic for branching time. Submitted for publication, 1997.

[4] V. Goranko. Hierarchies of modal and temporal logics with reference pointers. *Journal of Logic, Language and Information*, 5:1–24, 1996.

[5] S. Passy and T. Tinchev. An essay in combinatory dynamic logic. *Information and Computation*, 93:263–332, 1991.

7 Logic and uncertainty

Emil Weydert, Manfred Jaeger and Leon van der Torre (since 97)

Reasoning under uncertainty is a central issue in artificial intelligence and mainstream computer science. In our research area, we investigate qualitative and quantitative approaches for dealing with uncertainty, and how they are linked, from a systematic, logical perspective.

The last decade has seen a renaissance of probability theory, arguably the most powerful and best understood uncertainty formalism, in AI. In particular, the development of more efficient algorithms to handle probabilistic information, e.g. Bayesian networks, and some progress in probabilistic logic have increased its role in knowledge representation and contributed to its popularity. In addition, probabilistic and quasi-probabilistic approaches to default reasoning (plausible inference exploiting default rules, i.e. normal case rules with possible exceptions like $\mathit{Bird}(x)$ *normally implies* $\mathit{Flies}(x)$) have turned out to be quite promising. Conversely, earlier qualitative approaches to nonmonotonic reasoning, i.e. where the addition of new information may enforce the withdrawal of previous conclusions, have provided a fresh perspective of probabilistic reasoning. Linking quantitative and qualitative approaches to uncertainty is important because in practical applications, knowledge is often heterogeneous, including quantitative as well as qualitative items. Our research has concentrated on three areas:

Quasi-probabilistic inference where we have introduced and investigated a new paradigm for default reasoning close to qualitative entropy maximization and exploiting order of magnitude probabilities.

Principles of nonmonotonic reasoning where we have defined and axiomatized a powerful default quantification semantics, and proposed a very general representation independence concept for nonmonotonic inference.

Probabilistic reasoning where we have answered some questions in the semantics of probabilistic logic, developed a more expressive Bayesian network concept, and thought about a general formal framework for datamining using probabilistic logic.

Some of this work has been done in the context of the DFG-project *Defaults and Probability.* With the collaboration of Leon van der Torre, we have also started research in qualitative decision theory, which supplements and continues our work in default reasoning.

7.1 Quasi-probabilistic default inference

The quasi-probabilistic paradigm for default reasoning interprets defaults as constraints on ranking measures, a kind of implausibility or probabilistic order-of-magnitude valuations [10]. Within this framework, a default inference relation can be defined from those conditional relationships validated by the most preferred models of a given default set Δ.

$$\Delta \mathrel{|\!\sim} \mathit{Bird}(a) \rightarrow \mathit{Flies}(a) \quad \text{iff} \quad \mathit{Pref}(\mathit{Mod}(\Delta)) \vDash R(\mathit{Bird}(a) \wedge \mathit{Flies}(a)) < R(\mathit{Bird}(a) \wedge \neg\mathit{Flies}(a))$$

The main task is therefore to find a suitable preference criterion compatible with the known desiderata for defaults and nonmonotonic inference. We have investigated several ways to implement the minimal information principle in the quasi-probabilistic context. The basic idea is to prefer those ranking measure models which are constructible by a minimal iterated revision process starting from the uniform valuation. This gives us a whole family of powerful and well-behaved default inference formalisms (JX-entailment) [5, 7, 8]. Quasi-probabilistic default reasoning is closely related to approaches anchored in nonstandard probability theory, i.e. which admits infinitesimally small probability values. In this framework, we have defined two more robust variants of entropy-maximization-based default inference [7, 8]. Surprisingly, it turns out that they are intimately linked to JX-entailment relations, which in fact may be seen as qualitative approximations of robust entropy maximization [8]. There is a reasonable hope that more far-reaching equivalence results can be obtained.

7.2 Principles of nonmonotonic reasoning

Most research on default conditionals has concentrated on formalisms which are essentially propositional. However, it is possible to show that there are representational tasks which require many-dimensional dyadic default quantifiers, e.g. $\mathit{Likes}(x,y) \rightarrow_{x,y} \mathit{Likes}(y,x)$. We have considered an extension of first-order predicate logic with a new generalized quantifier, called the rational default quantifier, which is interpreted by quasi-probabilistic ranking measures over all product domains. It subsumes and refines the original, well-known notion of a propositional default conditional, admits a sound and complete axiomatization (RDQ), and overcomes some basic problems of other first-order conditional approaches [6, 9].

An issue that originally has been raised in the context of probabilistic maximum entropy inference is that of representation independence. For maximum entropy inference representation dependence refers to the fact that two different encodings, using different propositional vocabularies, of seemingly equivalent probabilistic information usually lead to clearly non-equivalent conclusions. For example, the two statements *the probability of life on Mars is at least 0.4*, and *the probability of either plant life or animal life on Mars is at least 0.4*, formulated in propositional probabilistic logic, and evaluated with the maximum entropy principle, will lead to different conclusions as to the probability of *life on Mars*, respectively *plant life or animal life on Mars.* The basic logical structure of this phenomenon is not contingent on maximum entropy inference, or even probabilistic inference in general. Rather, it can be viewed as a logical property with respect to which we can investigate every nonmonotonic inference relation. In order to provide a uniform framework for such investigations, in [2] a general concept of representation independence for nonmonotonic logics has been developed. It first clarifies what it means that two different knowledge bases (usually using different vocabularies) encode the same information, and on that basis arrives at a formalization of representation (in)dependence. Applications of the abstract framework include the derivation of a rule for modifying existing probabilistic inference strategies so as to make them more representation independent [2], and a clarification of the representation independence properties of the logic $\mathcal{L}^{\beta}$ developed in [1].

7.3 Probabilistic reasoning

An interesting model-theoretic question left open in [1] concerned the possibility of constructing model-theoretic structures for the logic $\mathcal{L}^{\beta}$ there defined, by appealing to a suitable general extension theorem for finitely additive probability measures. Here it has turned out that the envisaged extension theorem does not hold, and, furthermore, that $\mathcal{L}^\beta$-models of a specific interesting form do not exist. These results will be included in a future publication based on [1].

Bayesian networks currently are the most widely used tool for reasoning under uncertainty. In their basic form being somewhat limited in their ability to model stochastic relations on complex, structured domains, numerous variants of Bayesian networks have been introduced for special purposes such as modeling temporal developments, or probabilistic relations on some domain of individuals. In the latter case our task can be understood as specifying a probability measure on the set of first-order structures for some given vocabulary and the domain at hand. In [3] a method is proposed that allows us to define quite complex probability distributions of this kind, using a very compact Bayesian network representation. Inference from these networks is not harder than in previously proposed less expressive formalisms, and it is hoped that the new representation paradigm will eventually even lead to an inference technique of lower complexity than obtained so far.

In joint work with Heikki Mannila [4] a general concept of the task of data mining based on probabilistic logic has been developed. The aim of this work is to identify more clearly the common structures in the great variety of methods for extracting complex information from databases that are investigated in the field of data mining. A more unified view of the problems addressed in this area can help to develop more general and flexible solution paradigms.

References

- [1] M. Jaeger. *Default Reasoning about Probabilities.* PhD thesis, Universität des Saarlandes, November 1995.
- [2] M. Jaeger. Representation independence of nonmonotonic inference relations. In L. Carlucci-Aiello, J. Doyle, and S. Shapiro, editors, *Principles of Knowledge Representation and Reasoning, Proceedings of the Fifth International Conference (KR-96)*, pages 461–472, Cambridge, Massachusetts, 1996. Morgan Kaufmann.
- [3] M. Jaeger. Relational bayesian networks. To appear in Proceedings of UAI-97, 1997.
- [4] M. Jaeger, H. Mannila, and E. Weydert. Data mining as selective theory extraction in probabilistic logic. In R. Ng, editor, *Proceedings of the 1996 SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, Technical Report 96-08, Montreal, Canada, May 1996. Department of Computer Science, British Columbia.
- [5] E. Weydert. Default entailment. a preferential construction semantics for defeasible inference. In I. Wachsmuth, C.-R. Rollinger, and W. Brauer, editors, *Advances in Artificial Intelligence: Proceedings of the 19th German Annual Conference on Artificial Intelligence*, volume 981 of *Lecture Notes in Artificial Intelligence*, pages 173–184, Bielefeld, Germany, September 1995. Springer.
- [6] E. Weydert. Default quantifier logic. In I. Niemelä, editor, *Proceedings of the ECAI'96 Workshop on Integrating Nonmonotonicity into Automated Reasoning Systems*, volume 18-96 of *Fachberichte Informatik*, pages –, Budapest, Hungary, September 1996. Universität Koblenz-Landau, Institut für Informatik.
- [7] E. Weydert. System J – revision entailment: Default reasoning through ranking measure updates. In D. Gabbay and H. J. Ohlbach, editors, *Practical Reasoning - International Conference on Formal and Applied Practical Reasoning, FAPR'96*, volume 1085 of *Lecture Notes in Computer Science*, pages 637–649, Bonn, Germany, June 1996. Springer.
- [8] E. Weydert. Qualitative entropy maximization - a preliminary report. In E. Weydert, G. Brewka, and C. Witteveen, editors, *DGNMR97 - Proceedings of the Third Dutch/German Workshop on Nonmonotonic Reasoning Techniques and their Applications*, pages 63–72, Saarbrücken, Germany, 1997. Max-Planck-Institut für Informatik.

- [9] E. Weydert. Rational default quantifier logic. In D. Gabbay, R. Kruse, and A. Nonnengart, editors, *Qualitative and Quantitative Practical Reasoning*, volume 1244 of *Lecture Notes in Artificial Intelligence*, pages 589–600, Bad Honnef, Germany, June 1997. Springer. to be published.

[10] E. Weydert. General belief measures. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 575–582, University of Washington, Seattle, WA, 1994. Morgan Kaufmann.

8 Constraint solving

The idea of constraint solving is a very old and very general one: try to solve a problem by formulating it as a constraint. This means that one states the problem as a formula in a specific logical formalism which comes with a specific constraint-solving algorithm. As an example, take systems of linear equations with Gaussian elimination. The algorithm computes the solutions of the formula and, hence, of the problem. The resulting general research goal is clear: Find good constraint domains with good constraint-solving algorithms for important and suitable problem areas. The art is to find the right balance between the expressiveness and universality of the constraint domain and the efficiency of the constraint-solving algorithm.

In §8.1 and §8.1 we deal with constraints that arise from the problem of analysing programs. In §8.1 we consider systems for *programming constraint solvers.*

8.1 Symbolic constraints

If one wants to reason about programs, one encounters symbolic constraints. By this we understand constraints that are interpreted over data structures carrying symbolic information. Such data structures are usually trees (algebraically, ground terms). Compilers employ trees to represent and manipulate the program syntax. In program analysis, roughly, information about the run-time behavior of a program is computed by executing the programs with symbolic values (hence, with trees). Trees are also the basic data structures in tools that are employed to reason about programs, i.e. theorem provers or, on a lower level, logical or functional programming languages. The execution traces of concurrent programs are modeled as strings (or general trees); a temporal-logic formula expressing the correctness of a concurrent program is nothing but a constraint on the 'admissible' execution traces of the program (we have work in progress on a system called LISA that exploits this point of view; see §9.4)

Set constraints

Witold Charatonik and Andreas Podelski

Coined in 1990, the notion of set constraints now refers to a wide range of classes of constraints over set-valued variables. The main application domain is set-based program analysis and type inference for functional, imperative and logic programming languages. The satisfiability problems for the various classes of set constraints have been widely studied, in part for their significance for applications, in part for their own sake.

Given the actual state of the art, two research directions seem possible. Either one tries to push the frontier of decidable sub-logics further by investigating more and more expressive classes of set constraints; this is the direction taken by, for example, McAllester et. al. in [8]. Or, given the NEXPTIME-completeness results of the general classes, one tries to single out practically useful and fundamentally interesting classes of set constraints which trade a restricted expressiveness with better algorithmic properties; this is the direction that we have taken.

Set constraints with intersection *Definite set constraints* were the first class of set constraints for which decidability was shown [6]. Bachmair, Ganzinger and Waldmann [2] showed that the satisfiability problem for this class is in NEXPTIME. The relation of the class to other classes of set constraints has not been investigated before. Due to the the apparent difference in syntax, the two classes of definite and of positive set constraints were often considered incomparable.

In [11], we show that definite set constraints can be seen as a proper subclass of positive set constraints, namely positive constraints without complement and union symbols. We call them *set constraints with intersection* (the only operators forming the expressions are constructors and intersection). We give an algorithm, in a concise presentation, which (1) performs the satisfiability test in exponential time, (2) computes which variables denote the empty set in the least solution, and (3) represents the least solution as a tree automaton. Points (2) and (3) are important for the application to program analysis. Our algorithm is the fixpoint iteration under the operator that adds immediate consequences under nine logical axioms. Each consequence is an inclusion between two terms from a set of bounded size; this is used to show the termination and to derive the upper bound. Using a reduction from a tree automata problem, we give an exponential lower bound and thus obtain that the satisfiability problem is DEXPTIME-complete. This is the first time that a class of set constraints (over a general constructor alphabet) has been shown to fall into this complexity class. Since the classes of definite set constraints and of set constraints with intersection are not only equivalent in expressive power but also linearly inter-reducible, we hereby settle also the complexity question for definite set constraints, the historically first class for which the decidability question has been solved.

Set constraints for greatest models In the conclusion section of [11], we point out the open problem for the class of set constraints with union. The solution of the problem has so far resisted all efforts invested by us and others. In [10] we have, however, found a way to accommodate the union operator in a useful way, namely by the use of the projection operator. We consider set constraints consisting of inclusions $x \subseteq \tau$ between a variable x and a *positive set expression*, i.e. a term τ built up with intersection, union and projection. We solve such a set constraint by computing a representation of its greatest solution, essentially as a tree automaton. We obtain that the problem of the emptiness of its greatest solution (and a related satisfiability problem) is DEXPTIME-complete. These set constraints cover the application that we have in mind, namely the set-based analysis of logic programs with respect to the greatest model semantics (hence, we call them *set constraints for greatest models*). As a novelty of [10], we use automata over *infinite* trees to represent the solutions. The values of the set constraints, if these are employed in our intended application, range over sets of infinite trees. This is because we analyse reactive, i.e. possibly non-terminating programs; roughly, we need infinite trees in order to reason with the limits of infinite executions.

Nonempty-set constraints A close look at the constraint-solving algorithms shows that a lot of overhead occurs when one has to take care of the special role played by the empty set. In fact, some implementations [1] are 'optimized' by ignoring this special role (i.e. they trade efficiency for incompleteness).

Thus, it seems natural to restrict the interpretation domain to nonempty sets only and to investigate the new classes of *nonempty-set constraints* that arise. This is what we have started to do in [12]. We have shown that these classes are interesting from an algorithmic as well as from a fundamental point of view. They are related with the Herbrand constraint domain (equalities over finite trees), and they yield new insights and new algorithms for solving set constraints with the standard interpretation (of possibly empty sets of trees.)

We show in [12] that the satisfiability problem is linear if we take equalities between Herbrand terms interpreted over nonempty sets. This is due to a connection with the constraint domain of equalities over finite trees that we establish: the two first-order theories are equivalent. If we take *atomic nonempty-set constraints*, i.e. inclusions between Herbrand terms interpreted over nonempty sets, we can show that the satisfiability problem is cubic (also in [12]). If we admit all set operators, it lies in NEXPTIME; this follows from the fact (established in [3]) that set constraints with negation can express nonemptiness.

Independence property The *independence property* is a fundamental property for constraint domains. One well-known example of a constraint domain with the independence property is formed by linear equations over the real numbers. For comparison (and to give some intuition), in conjunction with the *non-linear* equation $(x-1)\cdot(x-2) = 0$ the two negated equations $x \neq 1$ and $x \neq 2$ are *not* independent from each other. Various other constraint domains have been shown to own the independence property, sometimes with difficult proofs. Colmerauer has introduced the notion for the manipulation of inequations in the context of constraint logic programming (CLP) [4]. A general study of the property [7] shows its importance in various symbolic computation areas.

In [9] we prove the independence property for atomic nonempty-set constraints. For comparison, any class of set constraints with the interpretation of possibly empty sets of trees does not have the independence property: In the constraint $f(x,y) \subseteq 0 \wedge x \not\subseteq 0 \wedge y \not\subseteq 0$, the conjuncts $x \not\subseteq 0$ and $y \not\subseteq 0$ are *not* independent from each other. In [11] we strengthen this result by establishing that nonempty-set constraints with intersection enjoy the independence property too. This is the first time, besides [5], that a constraint domain with high expressiveness and complexity is shown to have the independence property.

Set constraints with negation In [11], we use the above results for nonempty-set constraints in order to give an exponential satisfiability algorithm for the class of *negative set constraints with intersection*. The idea that underlies the general schema of this algorithm is based on the following observation: $x \not\subseteq y$ is satisfiable only of x is a nonempty set. The first part of the algorithm infers the set of nonempty variables in a constraint γ. Then we can split γ into one part which is trivially satisfiable, and another part γ_{ne} where all variables are nonempty. We have thus reduced the satisfiability problem for γ to the one for γ_{ne} over the interpretation domain of nonempty sets of trees. There, the independence property allows us to handle each negated inclusion separately (by the entailment algorithm that we devise). The correctness of the theorem depends crucially on the assumption that the constructor alphabet is infinite. The interpretation of this assumption might be that the constructor alphabet is never fully known, or is always extensible (which fits with the modularity in program analysis).

References

[1] A. Aiken and E. Wimmers. Type inclusion constraints and type inference. In *Proceedings of the 1993 Conference on Functional Programming Languages and Computer Architecture*, pages 31–41, Copenhagen, Denmark, June 1993.

[2] L. Bachmair, H. Ganzinger, and U. Waldmann. Set constraints are the monadic class. In *Eighth Annual IEEE Symposium on Logic in Computer Science*, pages 75–83, 1993.

[3] W. Charatonik and L. Pacholski. Negative set constraints with equality. In *Ninth Annual IEEE Symposium on Logic in Computer Science*, pages 128–136, 1994.

[4] A. Colmerauer. Equations and inequations on finite and infinite trees. In *Proceedings of the 2nd International Conference on Fifth Generation Computer Systems*, pages 85–99, 1984.

[5] H. Comon, M. Haberstrau, and J.-P. Jouannaud. Decidable problems in shallow equational theories (extended abstract). In *Seventh Annual IEEE Symposium on Logic in Computer Science*, pages 255–265, 1992.

[6] N. Heintze and J. Jaffar. A decision procedure for a class of set constraints (extended abstract). In *Fifth Annual IEEE Symposium on Logic in Computer Science*, pages 42–51, 1990.

[7] J. L. Lassez and K. McAloon. A constraint sequent calculus. In *Fifth Annual IEEE Symposium on Logic in Computer Science*, pages 52–61, June 1990.

[8] D. A. McAllester, R. Givan, C. Witty, and D. Kozen. Tarskian set constraints. In *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science*, pages 138–147, New Brunswick, New Jersey, July 1996. IEEE Computer Society Press.

• [9] W. Charatonik and A. Podelski. The independence property of a class of set constraints. In E. Freuder, editor, *Principles and Practice of Constraint Programming, Proceedings of the Second International Conference (CP'96)*, volume 1118 of *Lecture Notes in Computer Science*, pages 76–90, Cambridge, USA, August 1996. Springer.

• [10] W. Charatonik and A. Podelski. Set constraints for greatest models. Research Report MPI-I-97-2-004, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, April 1997.

• [11] W. Charatonik and A. Podelski. Set constraints with intersection. In G. Winskel, editor, *Proceedings of the Twelfth Annual IEEE Symposium on Logic in Computer Science (LICS)*, Warsaw, Poland, 1997. IEEE Computer Society, IEEE. To appear in June 1997.

• [12] M. Müller, J. Niehren, and A. Podelski. Inclusion constraints over non-empty sets of trees. In M. Dauchet, editor, *Proceedings of the 4th International Joint Conference on Theory and Practice of Software Development (TAPSOFT)*, volume 1214 of *Lecture Notes in Computer Science*, pages 345–356, Lille, France, April 1997. Springer.

Set-based program analysis

Witold Charatonik and Andreas Podelski

The algorithms that we have devised for solving set constraints can be used directly to enhance the power of existing systems performing the *set-based analysis* of (imperative, functional and logic) programs. This kind of program analysis computes the so-called monadic approximation (obtained by ignoring dependencies between variables) of the *collecting semantics* of a program (which associates with each point in the program the set of valuations of program variables that can occur at that point during execution). A set constraint is syntactically inferred from the program. The algorithmically important part of the set-based analysis is the solving of the set constraint.

Kozen has developed a system called $\mathit{CLP}(\mathcal{SC})$ (for: constraint logic programming over set constraints), in which one can program the constraint inference for a given class of programs and thus obtain a system performing the set-based analysis. The built-in set-constraint solver works over a general class of set constraints. In order to be practical, however, the constraint-solving algorithms have to be specialized for their specific application.

The original application of set-based analysis is the use of the statically inferred information about run-time values for compiler-optimization. Our principle motivation to work on set-based analysis is, however, the automated verification and falsification of programs, and here, in particular, concurrent constraint programs (CCP). An example of a CCP language is Oz [2] developed by Gert Smolka and his group at the DFKI in Saarbrücken.

In [1], we give an application of this set-based analysis to the automatic falsification of programs written in the pure CCP framework (such programs can be approximated by constraint logic programs). We have established a formal connection between the greatest-model semantics and certain temporal properties of executions of the programs (inevitability and universal safety in Clarke's CTL framework). The approximation via set-based program analysis yields either necessary or sufficient conditions for such properties to hold of the program. This yields an automated verification or falsification procedure (implementing semi-tests with yes-or-don't-know answers).

At the moment, we are working on the application of these results to the automatic verification and falsification of Oz programs. Oz is a CCP language and system developed by Gert Smolka and his group at DFKI in Saarbrücken; some Oz-specific programming constructs (cells and higher-order procedures go beyond the paradigm of pure CCP. The work on this application is done in collaboration with DFKI. Gert Smolka and Andreas Podelski co-supervise a doctoral student at the DFKI, Martin Müller, who will incorporate Oz-specific programming constructs into our analysis and implement the set-constraints inference and the solving algorithms. His thesis is to be finished this year.

References

- [1] W. Charatonik and A. Podelski. Set constraints for greatest models. Research Report MPI-I-97-2-004, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, April 1997.

[2] G. Smolka. The Oz Programming Model. In *Volume 1000 of Lecture Notes in Computer Science*. Springer, Berlin, 1995.

Constraint programming

Andreas Podelski

We now consider systems for programming constraint solvers. Such systems are needed since this programming task is very difficult and occurs often (namely, each time when the problems in the new domain are so difficult that they require a dedicated problem solver). For an introduction to the field of constraint programming, see the collection of papers in [3].

Defining programming constructs The paradigm of *constraint logic programming (CLP)* yields a universal method for building a constraint solver (that solves difficult problems from a very specific domain) on top of a family of given 'low-level' constraint solvers (that solve logical formulas in a specific logic, for example linear arithmetic).

The CLP languages have declarative as well as purely operational functionalities, such as facilities for coroutining. This appear in one form or another in most CLP- or Prolog-Systems that are being used (for example: as 'freeze' in Prolog [2] or as residuation in Life [1]). The dogma that their semantics should be declarative has lead to the obscure situation that no defining operational semantics for CLP-languages with coroutining has been devised yet.

In [4], we give the first definition of the operational semantics of logic programs with coroutining facilities. Our approach is based on *abstract-tree rewriting* which appeared in the π-calculus. The abstract-tree rewriting model proves useful for constraint programming since constraint propagation and simplification (for testing satisfiability as well as entailment) can be accommodated concisely by means of the congruence on trees. This congruence is defined using the logical meaning of constraints.

Constraints in local computation spaces The paradigm of *concurrent constraint programming (CCP)* combines constraint solving with concurrency in order to (1) enhance constraint solving and (2) embed constraint solving into reactive, possibly distributed applications. Examples of implemented CCP languages are AKL (developed at SICS in Sweden) and Oz (developed at DFKI in Saarbrücken).

The combination of reactiveness and constraint-solving requires an extension of the low-level constraint solvers used in CLP languages to act in *local computation spaces*; formally, they solve *deep guard* problems. The backtracking mechanism which implements the search of constraint solutions is incompatible with concurrent I/O and must be localized to these local computation spaces.

In [6] (preliminary version in [5]), we present and prove correct, for the first time, an incremental algorithm for solving rational tree constraints to be used in CCP systems with deep guards. The algorithm is presented as the simplification of the constraints which form the (possibly deep) guards and which are situated at different nodes in a tree (of arbitrary depth). The nodes correspond to local computation spaces. In this algorithm, a variable may have multiple bindings (which each represent a constraint on that same variable in a different node). These may be realized in various ways. We give a simple fixed-point algorithm and use it for proving that the tests implemented by another, practical algorithm are correct. We formulate the results in this paper for rational tree constraints; they can be adapted to finite and feature trees.

References

[1] H. Aït-Kaci, B. Dumant, R. Meyer, A. Podelski, and P. V. Roy. *Wild LIFE: A User Manual.* Digital Equipment Corporation, Paris Research Laboratory, Rueil-Malmaison, France, April 1993.

[2] M. Carlsson, J. Widèn, J. Andersson, S. Andersson, K. Boortz, H. Nilsson, , and T. Sjöland. *SICStus Prolog User's Manual.* SICS, Kista, Sweden, 1991.

• [3] Andreas Podelski, editor. *Constraint Programming: Basics and Trends*, volume 910 of *Lecture Notes in Computer Science.* Springer, Berlin, 1995.

• [4] A. Podelski and G. Smolka. Operational semantics of constraint logic programming with coroutining. In L. Sterling, editor, *Proceedings of the 12th International Conference on Logic Programming*, pages 449–463, Kanagawa, Japan, 1995. The MIT Press.

• [5] A. Podelski and G. Smolka. Situated simplification. In U. Montanari and F. Rossi, editors, *Proceedings of the First International Conference on Principles and Practice of Constraint Programming (CP'95)*, volume 976 of *Lecture Notes in Computer Science*, pages 328–344, Cassis, France, 1995. Springer.

• [6] A. Podelski and G. Smolka. Situated simplification. *Theoretical Computer Science*, 173(1):209–233, February 1997. Preliminary Version in Ugo Montanari, editor, Proceedings of the First International Conference on Principles and Practice of Constraint Programming (CP'95). Springer LNCS 976, 1995.

Unification in sort theories

Christoph Weidenbach

We have investigated unification in different sort theories. Sorts are not solely represented by a single monadic predicate, but also by sets of such predicates denoting their intersection. This sort concept naturally introduces a top sort (the empty set) that is always interpreted by the overall, non-empty domain. All other sorts may denote the empty set. The domain of variables is restricted by sorts. Then two terms are sorted unifiable, if there exists a syntactic unifier where assignments to variables satisfy the sort restrictions. The semantics of sorts is specified by a sort theory that is a finite set of monadic atoms, called *declarations.* Declarations establish the relationship between terms and sorts, e.g., $S(x_{T,R})$ expresses that the intersection between the sorts T and R is a subsort of S and $S(f(y_R, g(y_R)))$ states that all terms of the form $f(t, g(t))$, where t is of sort R, are of sort S. Note that declarations relativize to monadic Horn-clauses where only variables occur in body atoms. With respect to syntactic properties of terms in declarations we distinguish the sort theories presented in Figure 8.4.

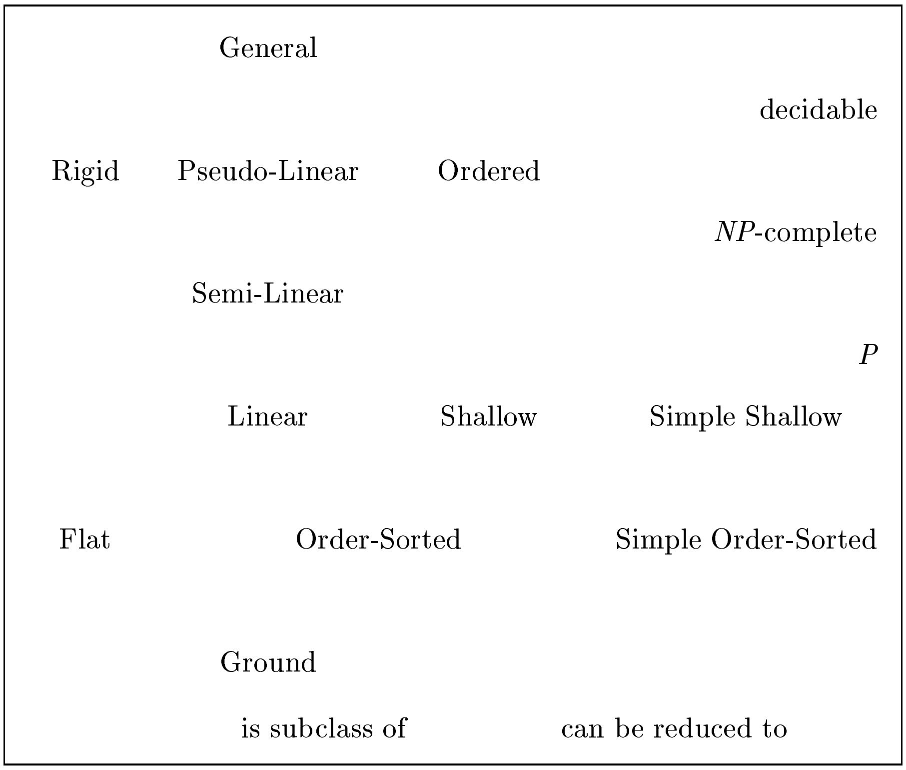


Figure 8.4: Complexity of Sorted Unification in Different Sort Theories

Our contributions to the shown complexity results concern in particular flat, rigid, shallow, ordered and pseudo-linear sort theories [4, 3, 5], whereas order-sorted, linear, semi-linear, and general sort theories have already been considered in the literature [6, 7]. We also showed that the border between pseudo-linear and general sort theories is strict: Sorted unification is already undecidable for sort theories where the depth difference between non-linear variable occurrences in declarations is at least one. It remains decidable if non-linear variable occurrences occur at the same depth (pseudo-linear sort theories). Due to our stronger sort language, the unification type for all sort theories is finitary, except for pseudo-linear and general sort theories. The latter two have unification type infinitary.

For applications of sorted unification to automated theorem proving the empty sort problem becomes important. For all sort theories listed in Figure 8.4 we showed that this problem is at most as hard as the sorted unification problem. For some of the sort theories, however, it is simpler: The empty sort problem is decidable in polynomial time for ground sort theories, whereas sorted unification for this class is already *NP*-complete. Furthermore, we presented a sort theory with a trivial empty sort problem but an undecidable unification problem correcting some remarks from the literature stating that the sorted unification problem can be effectively reduced to the empty sort problem.

The empty sort problem for sort theories is closely related to the emptiness test for tree automata. With respect to this problem, regular, $\mathit{Rec}_=$ [1] and $\mathit{GEA}_=$ [2] automata correspond to order-sorted, shallow, and ordered sort theories, respectively.

References

[1] B. Bogaert and S. Tison. Equality and disequality constraints on direct subterms in tree automata. In A. Finkel and M. Jantzen, editors, *Proceedings of 9th Annual Symposium on Theoretical Aspects of Computer Science, STACS92*, volume 577 of *LNCS*, pages 161–171. Springer, 1992.

[2] A.-C. Caron, H. Comon, J.-L. Coquidé, M. Dauchet, and F. Jacquemard. Pumping, cleaning and symbolic constraints solving. In S. Abiteboul and E. Shamir, editors, *Automata Languages and Programming. 21st International Colloquium, ICALP'94*, volume 820 of *LNCS*, pages 436–447. Springer, 1994.

• [3] C. Weidenbach. *Computational Aspects of a First-Order Logic with Sorts.* PhD thesis, Universität des Saarlandes, 1996.

• [4] C. Weidenbach. Unification in pseudo-linear sort theories is decidable. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 343–357, New Brunswick, USA, 1996. Springer.

• [5] C. Weidenbach. Unification in sort theories and its applications. *Annals of Mathematics and Artificial Intelligence*, 18(2-4):261–293, 1996.

[6] M. Schmidt-Schauß. *Computational aspects of an Order-sorted logic with term declarations*, volume 395 of *LNAI.* Springer, 1989.

[7] T. E. Uribe. Sorted unification using set constraints. In *11th International Conference on Automated Deduction, CADE-11*, volume 607 of *LNCS*, pages 163–177. Springer, 1992.

8.2 Non-elementary lower bounds for constraint systems

Sergei Vorobyov

It is by now commonly accepted that the complexity analysis is an important ingredient in the design of efficient algorithms and new logical systems for program analysis. This becomes especially crucial nowadays in connection with large-scale theorem proving, program verification, constraint logic and functional programming projects, where large problem instances are involved. Despite that fact that the decidability of virtually all classical theories is settled, the rapid evolution of Computer Science again and again puts forward new decidability and complexity problems related to new logical systems developed for the needs of program analysis and verification.

The hardest known decidable theory

Define the iterated exponential by $F(n,0) = n$ and $F(n,k+1) = 2^{F(n,k)}$, and the iterated logarithm by $\log^0(n) = n$ and $\log^{k+1}(n) = \log(\log^k(n))$.

It is generally believed that the NON-ELEMENTARY problems, that do not belong to the class $\bigcup_{k\in\omega} \mathrm{TIME}(F(n,k))$, are the worst possible decidable problems. However, the class NON-ELEMENTARY is not homogeneous, and some problems in it are much more difficult that the others. By the time hierarchy theorem, the class of non-elementary problems $C = \mathrm{TIME}(F(n,\log^9(n))) \setminus \mathrm{TIME}(F(n,\log^{10}(n)))$ is non-empty, but for all inputs of size up to $n = F(1,9)$ (one will never encounter in practice) the function $F(n,\log^9(n))$ is majorated by the simple exponential 2^n. So the mere fact that the problem in C is non-elementary is practically immaterial, and thus non-elementary does not necessarily mean bad. In this respect it is important to know, once a problem is proved non-elementary, how large is k in $\mathrm{TIME}(F(n, \log^k(n)))$ in its lower and upper bounds.

On the other hand, it is interesting to know how difficult a 'natural' decidable problem can be ('natural' informally means a problem, which has arisen previously in the mathematical literature, excluding the complexity theory; for example, decision problems drawn from logic, automata theory, algebra; by using hierarchy theorems, diagonalization, or bounded variants of undecidable problems one can easily construct arbitrarily hard 'unnatural' problems).

In his survey paper [4] A. Meyer mentioned, as a curious empirical observation, that all known natural decidable non-elementary problems require at most (upper bound)

$$F(1,n) = 2^{2^{\cdot^{\cdot^{2}}}} \Big\}n$$

Turing machine steps on inputs of length n to decide. Since the mid-seventies and until recently the highest known lower bounds for natural decidable theories were at most $F(1, cn)$ with 'only' linearly high towers of 2's, and it was an open problem whether there exist natural decidable theories requiring time exceeding $F(1, f(n))$, where $f(n)$ is not linearly bounded (e.g. n^2 or 2^n), cf., A. Meyer, and K. Compton and C. Henson [1]. Theories of this kind could be identified with practically undecidable ones, and could yield very strong lower bounds when used in reductions to other theories.

In [7] we settled this problem in the positive sense by presenting the 'hardest' currently known natural decidable theory Ω due to L. Henkin [2]. Roughly, Ω is the rudimentary higher-order theory of stratified hereditary finite sets over the two-element base set $\{0, 1\}$ formulated in the signature $\{\in\}$ with atomic inclusions of the form $x^k \in y^{k+1}$ (meaning that a k-level set x is an element of a $k+1$-level set y). By using the uniform lower bounds techniques due to Compton and Henson we have proved that every decision procedure for Ω should necessarily make more than $F(1, 2^{cn})$ steps on infinitely many inputs of length n, where the height of the tower of 2's is exponential in the length of input. The best earlier known lower bound for Ω was 'only' $F(n, \varepsilon \cdot \log(n))$ due to M. Fischer and A. Meyer.

Improved lower bounds for the lambda calculus

R. Statman's well-known result on non-elementary recursiveness of the $\beta(\eta)$-equality in the simply typed lambda calculus left open a deep gap between the known upper bound on the $\beta(\eta)$-equality of the form $F(1, dn)$ due to W. W. Tait and the lower bound of the form $F(n, c\log(\log(n))$ implicit in Statman's proof. One might speculate that this weak lower bound, despite its indisputable mathematical interest, is immaterial and can be ignored in practical applications of the simply typed lambda calculus, where the lengths of inputs n are relatively small.

In [7], by applying the strong lower bound $F(1, 2^{cn})$ for Ω, described in §8.2, we settled the new improved lower bound of the form $F(1, cn)$ for the $\beta(\eta)$-equalities in the simply typed lambda calculus. Our lower bound matches (with a different constant, of course) with Tait's upper bound. As another consequence, we immediately get the $F(1, cn)$ lower bound on any normalization algorithm in the simply typed lambda calculus (a result due to H. Schwichtenberg; note that Schwichtenberg's result does not imply Statman's; on the other hand, Statman's yields only the weak $F(n, c\log(\log(n)))$ lower bound on normalization).

By using the $F(1, 2^{cn})$ lower bound for Ω we also reinforced in the lower bound in Statman's analog of Rice's theorem for the simply typed lambda calculus. This theorem asserts that any nontrivial set of closed terms of the same type closed under $\beta\eta$-equality is not elementary recursive. We have improved it by replacing 'not elementary recursive' with 'requiring time exceeding $F(1, cn)$. Only a weak $F(n, c\log(\log(n)))$ followed from previous results.

New lower bound for the higher-order matching problem

Theorem proving in higher-order logical frameworks and higher-order term rewriting (both in the heart of the higher-order logic and functional programming) are strongly based on the key notions of higher-order unification and matching.

A long-standing, still open, *higher-order matching* problem in the simply typed lambda calculus due to G. Huet consists in deciding, given a term t of type $\sigma_1 \to \cdots \to \sigma_n \to \tau$ and a term u of type τ (both in normal forms), whether there exist terms s_i of types σ_i $(1 \leq i \leq n)$ respectively such that $ts_1 \ldots s_n =_{\beta\eta} u$. The corresponding unification problem has been proved undecidable by Huet. By applying the $F(1, 2^{cn})$ lower bound for Ω, described in §8.2, we have settled the new $F(1, cn/\log(n))$ lower bound for the higher-order matching problem. The best lower bound one could have gotten for the problem without the new $F(1, 2^{cn})$ lower bound for Ω would be only $F(n, c\log(\log(n/\log(n))))$. This gives an example asked for in [1]: *Give nontrivial lower bounds for mathematically interesting theories whose decidability is still open.* The only nontrivial known lower bound for the higher-order matching problem was NP-hardness.

Tree constraints, bounded theories of trees

Tree-like structures are fundamental for almost all domains of Computer Science, and are especially relevant to logic programming, symbolic computation, data types, constraint solving, automated theorem proving, data bases, knowledge representation. Whenever the reasoning about a class of data structures is involved, it is interesting to know what is the *inherent computational complexity* of this reasoning. This may be crucial in practical implementations of theorem provers, constraint solvers, systems of logic programming.

The first-order theory of *finite trees*, also known as the theory of *term algebras*, or *Clark's equational theory*, and also the theory of the so-called *feature trees* all possess complete recursive axiomatizations and are decidable by quantifier elimination. The complexity of the quantifier-elimination procedures remained unknown. Very little was known about the inherent computational complexity of the decision problem for the theories. K. Kunen conjectured [3] that the decision problem for the theory of finite trees in infinite signatures is PSPACE-complete. In [5], by applying the uniform lower bound techniques due to Compton and Henson based on encoding of large binary relations we have proved that all theories of finite, rational, and feature trees, as well as all their subtheories, are non-elementary in the sense of Kalmar. More strongly, the lower bounds for all these theories are of the form $F(1, cn)$, i.e. towers of 2's growing linearly with the length of input.

As an alternative, in [6] we suggested a practical substitute for the theory of finite trees, called the *bounded theory of finite trees.* In this theory, instead of the unique usual equality $=$, one has an infinite family of equalities $\{=^d\}_{d\in\omega}$, with $s =^d t$ interpreted as true if and only if the trees s and t coincide to depth d, where d is written in binary. Thus instead of claiming for the complete equality, one has to specify explicitly which precision is needed in every comparison. We demonstrated that the bounded theory of trees is decidable within elementary space $2^{2^{cn}}$ for some $c > 0$, and thus can be considered a more practical alternative to the usual (unbounded) non-elementary recursive theory of finite trees. Trees are one of the basic domains in Constraint Logic Programming. One can hardly expect to use the full first-order theory of trees to express constraints, because of its non-elementary complexity. Allowing only existential quantification (as is usually done) seems to be a serious restriction of expressiveness (it can be solved in quasi-linear time). In this respect the bounded theory of finite trees, allowing for the full first-order quantification and being elementary, may be considered useful. It is known that the first-order theory of finite trees with the subtree predicate is undecidable. By using the same machinery we showed that the bounded theory of trees with the *to be a subtree at bounded depth* predicate is decidable and elementary recursive.

References

[1] K. J. Compton and C. W. Henson. A uniform method for proving lower bounds on the computational complexity of logical theories. *Annals of Pure and Applied Logic*, 48:1–79, 1990.

[2] L. Henkin. A theory of propositional types. *Fundamenta Mathematicæ*, 52:323–344, 1963.

[3] K. Kunen. Answer sets and negation as failure. In *4th International Conference on Logic Programming*, volume 1, pages 219–228. MIT Press, 1987.

[4] A. R. Meyer. The inherent computational complexity of theories of ordered sets. In *International Congress of Mathematicians*, pages 477–482, Vancouver, 1974.

• [5] S. Vorobyov. An improved lower bound for the elementary theories of trees. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 275–287, New Brunswick, USA, 1996. AT&T, Bell Labs, Lucent, IEEE, CADE inc, Springer.

• [6] S. Vorobyov. On the bounded theories of finite trees. In J. Jaffar and R. H. C. Yap, editors, *Second Asian Computing Science Conference, ASIAN'96*, volume 1179 of *Lecture Notes in Computer Science*, pages 152–161, Singapore, 1996. Springer.

• [7] S. Vorobyov. The 'hardest natural decidable theory. In *Proceedings of the 1997 IEEE Symposium on Logic in Computer Science (LICS'97)*, Warsaw, Poland, 1997. IEEE Computer Society, IEEE.

8.3 Pseudo-boolean constraint programming

Alexander Bockmayr, Peter Barth and Thomas Kasper

Constraint programming has become a promising new technology for solving complex combinatorial problems arising in industry, e.g. in scheduling, transportation, or personal assignment. It combines recent advances in declarative programming, like the constraint logic programming scheme or the use of symbolic constraint abstractions, with efficient algorithms from mathematics, artificial intelligence or operations research.

The standard approach to solving a combinatorial problem with constraint programming is to build a model with finite domain constraints, i.e. each variable takes its values from a finite set of natural numbers, and then this model is solved by enumeration and constraint propagation. The basic inference mechanism used here domain reasoning. Applying local consistency techniques from artificial intelligence, the domains of the variables are reduced in order to prune the search tree.

In [3, 2], we have developed, for the special case of pseudo-Boolean constraints, where all variables have the domain $\{0, 1\}$, a new approach for constraint solving, which is based on a more powerful inference mechanism. Instead of the very simple constraints of the form $x \geq l, x \leq u, x \neq v$ that can be obtained by domain reasoning, we derive more general linear inequalities or cutting planes that are used to tighten some relaxation of the problem within a branch-and-cut framework.

This general idea has been realized by us in two different ways. In the first, Peter Barth has developed in [5] a logic-based approach for pseudo-Boolean constraint solving, which extends techniques from computational propositional logic, in particular the Davis-Putnam procedure and propositional resolution, to the more general pseudo-Boolean framework. The basic formulas of his constraint solver are extended clauses saying that at least m out of n literals have to be true, where a literal is either a 0-1 variable or its negation. He has developed a comprehensive collection of new techniques for transforming general, possibly non-linear, 0-1 constraints into extended clauses, for simplifying sets of extended clauses by the generation of logic cutting planes, and for computing solved forms in the context of constraint programming. A 0-1 optimization system OPBDP that grew out of this work [1] has been made available via the World Wide Web and is used by a large number of people, both in academia and industry.

In the second, Thomas Kasper is currently developing the polyhedral counterpart, i.e. a polyhedral branch-and-cut solver for pseudo-Boolean constraint solving. The main goal of this work is to integrate techniques from integer linear programming and finite domain constraint programming. In contrast to the logic-based approach, the basic formulas are now general linear inequalities, which are solved by linear optimization. Disjunctive programming is used as an inference method to derive cutting planes that strengthen the linear programming relaxation of a pseudo-Boolean problem. This method can also handle mixed 0-1 problems which contain both continuous and discrete variables. A polyhedral branch-and-cut solver for pseudo-Boolean constraint solving, COUPE, has been implemented on top of the commercial linear programming package CPLEX. The COUPE system provides a platform to integrate state-of-the-art techniques from mathematical programming into constraint programming. It also allows us for the first time to exploit logic-based methods for constraint solving and optimization within a polyhedral framework.

Currently, we are on the way to combine the logic-based and the polyhedral approach. On the one hand, we can use information from solving the linear programming relaxation (e.g. upper/lower bounds, reduced costs) in the logic-based solver, on the other hand we can apply symbolic transformation and propagation techniques from the logic-based solver in order to tighten the linear programming relaxation.

We have applied our techniques to various practical problems, including warehouse location [9] and reconstruction of binary patterns in discrete tomography [4]. Alexander Bockmayr and Thomas Kasper visited COSYTEC SA, one of the leading companies in constraint technology, in April and May 1997, in order to apply the methods and tools that have been developed at MPI to real world industrial problems and to find out how they might be combined with COSYTEC's CHIP system.

Algebraic modeling Practical problem solving usually involves three phases: building a model of the problem, solving the model, understanding and analysing the solution. In [7, 8, 6], we investigate how constraint logic programming can support the first phase of this process, the modeling part, independently of the solver that is used in the second phase. This can be either a constraint programming or a mathematical programming system. In a first step, we show that the basic functionality of algebraic modeling languages as used in mathematical programming can be realized very easily in a pure logic programming system like PROLOG and that, even without any constraints, various additional features are available. If an algebraic modeling language is embedded into a logic programming framework, we have a full logic-based programming language at our disposal that allows us to model in a declarative way all those properties that we cannot directly express in the algebraic language. In a second step, we focus on the constraint solving facilities offered by constraint logic programming systems. We discuss the additional modeling power that we can get from symbolic constraints and show how our algebraic modeling approach can be extended to generate constraint logic programming models over finite domains. Then we further restrict our attention to pure 0-1 problems and explain how the complete constraint solver of our constraint logic programming system CLP($\mathcal{PB}$) can be used to generate automatically linear models of 0-1 problems that involve non-linearities and logical conditions between constraints. A prototype algebraic modeling system, PLAM (ProLog and Algebraic Modeling), has also been implemented and be made available via the World Wide Web.

References

[1] P. Barth. A Davis-Putnam based enumeration algorithm for linear pseudo-Boolean optimization. Research Report MPI-I-95-2-003, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, January 1995.

[2] P. Barth and A. Bockmayr. Finite domain and cutting plane techniques in CLP($\mathcal{PB}$). In L. Sterling, editor, *Proceedings of the Twelfth International Conference on Logic Programming*, Logic Programming, pages 133–147, Shonan Village Center, Hayama-Machi, Kanagawa, Japan, 1995. The MIT Press.

[3] A. Bockmayr. Solving pseudo-boolean constraints. In A. Podelski, editor, *Constraint Programming: Basics and Trends*, volume 910 of *Lecture Notes in Computer Science*, pages 22–38, Chtillon-sur-Seine, 1995. Springer.

[4] A. Bockmayr, T. Kasper, and T. Zajac. Reconstructing binary patterns from orthogonal projections using constraint programming. Presented at Dagstuhl Seminar on 'Discrete Tomography: Algorithms and Complexity ', Paper in preparation, January 1997.

- [5] P. Barth. *Logic-based 0-1 constraint programming.* Operations Research/Computer Science Interfaces. Kluwer, Boston, November 1995.

- [6] P. Barth and A. Bockmayr. Modelling mixed-integer optimisation problems in constraint logic programming. Research Report MPI-I-95-2-011, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, November 1995.

- [7] P. Barth and A. Bockmayr. Modelling 0-1 problems in CLP($\mathcal{PB}$). In *Proceedings of the Second International Conference on Practical Application of Constraint Technology*, pages 1–9, London, UK, 1996. The Practical Application Company.

- [8] P. Barth and A. Bockmayr. PLAM: Prolog and algebraic modelling. In *Proceedings of the Fifth International Conference on the Practical Application of Prolog*, pages 73–82, London, UK, 1997. The Practical Application Company.

- [9] A. Bockmayr and T. Kasper. Pseudo-boolean and finite domain constraint programming: A case study. In U. Geske and H. Simonis, editors, *Deklarative Constraint Programmierung*, number 297 in GMD-Studien, pages 29–41, Dresden, Germany, September 1996. GDM-Forschungszentrum Informationstechnik GmbH, GMD.

9 Systems

Along with 'pure' research, the Logic of Programs group has also developed (or is developing) various pieces of software which implement as practical tools our theoretical work.

In this section we describe our progress on these systems, their availability, and (where applicable) their performance as measured against other competitive systems.

9.1 SPASS & FLOTTER – version 0.55

Harald Ganzinger, Christoph Meyer, Andreas Nonnengart and Christoph Weidenbach

SPASS[5] and FLOTTER[6] [4, 10, 12, 8] are a clause normal form translator for first-order logic and a theorem prover for first-order logic with equality, respectively. The algorithm implemented by FLOTTER includes an improved form of the renaming technique originally developed by Thierry Boy de la Tour [3] and the optimized Skolemization technique by Ohlbach and Weidenbach [5]. Currently we are extending the Skolemization technique to incorporate recent results obtained by Nonnengart [9].

SPASS is based on the superposition calculus developed by Bachmair and Ganzinger [1, 2] extended with the sort techniques developed by Weidenbach [6, 7]. In addition to the usual inference, deletion and simplification rules, SPASS includes a rule for case analysis, called Splitting.

SPASS and FLOTTER have demonstrated their performance in various ways: SPASS won the open category at the CADE-13 theorem proving competition, it is one of the strongest provers on the TPTP problem library [11], and, with respect to the functional translation techniques, it shows an impressive behavior on propositional multi-model logic problems (§4.2, §6.2, §6.3). FLOTTER is one of the fastest and most powerful clause normal form translators currently available. For some problems it even produces the shortest clause normal forms known, allowing clause based theorem provers to obtain significant speed-ups compared to standard clause normal forms.

SPASS and FLOTTER can be obtained from the MPI homepage following the link `Software Projects/SPASS` or via anonymous ftp from our ftp-server in the directory `pub/SPASS`. In addition, we also have a (currently experimental) www-based interface to the systems available at

`http://spass.ag2.mpi-sb.mpg.de/~brahm/WebSPASS/`

The following students participated in the development of SPASS and FLOTTER: Uwe Brahm, Christian Cohrs, Thorsten Engel, Bernd Gaede, Enno Keen, Christian Cohrs and Georg Rock.

References

[1] L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994. Revised version of Technical Report MPI-I-91-208, 1991.

[2] L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder. Basic paramodulation. *Information and Computation*, 121(2):172–192, 1995. Revised version of TR MPI-I-93-236, 1993.

[3] T. Boy de la Tour. An optimality result for clause form translation. *Journal of Symbolic Computation*, 14:283–301, 1992.

[4] B. Gaede. Superposition extended with sorts. Master's thesis, Universität Kaiserslautern, July 1995.

• [5] H. J. Ohlbach and C. Weidenbach. A note on assumptions about skolem functions. *Journal of Automated Reasoning*, 15(2):267–275, 1995.

• [6] C. Weidenbach. *Computational Aspects of a First-Order Logic with Sorts.* PhD thesis, Universität des Saarlandes, 1996.

• [7] C. Weidenbach. Unification in sort theories and its applications. *Annals of Mathematics and Artificial Intelligence*, 18(2-4):261–293, 1996.

• [8] C. Weidenbach. Spass version 0.49. *Journal of Automated Reasoning*, 1997. To appear.

---

[5] **S**ynergetic **P**rover **A**ugmenting **S**uperposition with **S**orts, SPASS means 'fun' in German.

[6] FLOTTERmeans "faster" in German.

[9] A. Nonnengart. Strong skolemization. Research Report MPI-I-96-2-010, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, 1996.

[10] G. Rock. Transformations of first-order formulae for automated reasoning. Master's thesis, Universität des Saarlandes, July 1995.

[11] G. Sutcliffe, C. B. Suttner, and T. Yemenis. The TPTP problem library. In A. Bundy, editor, *Twelfth International Conference on Automated Deduction, CADE-12*, volume 814 of *Lecture Notes in Artificial Intelligence, LNAI*, pages 252–266, Nancy, France, June 1994. Springer.

[12] C. Weidenbach, B. Gaede, and G. Rock. Spass & flotter, version 0.42. In M. McRobbie and J. Slaney, editors, *13th International Conference on Automated Deduction, CADE-13*, volume 1104 of *LNAI*, pages 141–145. Springer, 1996.

9.2 The Saturate system

The Saturate system [1] is an experimental theorem prover written in Prolog. It is an implementation of chaining-based saturation calculi with an emphasis on exploiting redundancy. Within the period relevant to this report, the system has been extended in several ways. Additional reduction orderings have been implemented, together with a module for the automatic determination of precedences for non-logical symbols. Saturation and tableau expansion have been combined. Arbitrary first-order formulas are allowed as input. Tableau expansion is performed lazily, that is, only when subformulas are eligible for non-redundant chaining inferences. In particular, equivalences of the form $A \equiv \phi$, with atoms A and arbitrary ϕ are dealt with equationally. With these extensions we have experimented with the concept of order locality (cf. §4.1) and with proving problems in set theory.

References

[1] H. Ganzinger and R. Nieuwenhuis. The Saturate system. User manual maintained on the Web, 1994.

9.3 Pseudo-boolean constraint programming

OPBDP: Optimizing Pseudo-Boolean Davis-Putnam

Peter Barth

OPBDP [1, 2] is a generalized Davis-Putnam procedure for solving possibly non-linear 0-1 (or pseudo-Boolean) optimization problems with integer coefficients and thus an alternative to linear programming based methods in 0-1 programming. It supports different heuristics and a number of preprocessing techniques, e.g. fixing of literals, equation detection, or coefficient reduction. The most recent version provides an interactive user interface, linearization routines for non-linear problems, logic cut generation, and a dynamic memory allocation. The system is implemented in C++ and available on the World-Wide Web at

`http://www.mpi-sb.mpg.de/~barth/opbdp/opbdp.html`

COUPE: Constraint Programming and Cutting Plane Environment

Thomas Kasper

The COUPE system is a prototype implementation of a polyhedral branch-and-cut based constraint solver for pseudo-Boolean constraint programming. It provides an efficient implementation of the basic branch-and-cut framework, while preserving flexibility for modifications and extensions. The current version supports the generation of disjunctive cutting planes from arbitrary disjunctions, which includes as a special case the well-known lift-and-project method. Furthermore, COUPE has an interface to the algorithms of the OPBDP system developed by Peter Barth. This allows to combine the functionality of both systems and to build new hybrid algorithms for pseudo-Boolean constraint solving. COUPE is implemented in C++ and consists of about 25000 lines of code. To solve linear programs, it uses the commercial linear programming package CPLEX.

PLAM: ProLog and Algebraic Modeling

Peter Barth, Alexander Bockmayr

PLAM is an algebraic modeling system for mixed-integer linear optimization problems, which is based on logic programming [3]. On the one hand, it offers the basic functionality of traditional algebraic modeling languages from mathematical programming, on the other hand, a full declarative programming language is available, due to the embedding into a logic programming environment. The modeler can use logic to formulate his problem in a high-level and declarative way. Familiar algebraic notation can be incorporated very easily by suitable PROLOG predicates. The logic programming system will then automatically generate the corresponding input for a mathematical programming solver. Currently, a built-in linear programming solver of SICSTUS-PROLOG, the commercial solver CPLEX, and the free solver `lp_solve` are supported. Furthermore, the solver independent MPS-format of MILP problems can be produced. The PLAM system is available on the World Wide Web at

`http://www.mpi-sb.mpg.de/~barth/plam/plam.html`

References

[1] P. Barth. A Davis-Putnam based enumeration algorithm for linear pseudo-Boolean optimization. Research Report MPI-I-95-2-003, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, January 1995.

• [2] P. Barth. *Logic-based 0-1 constraint programming.* Operations Research/Computer Science Interfaces. Kluwer, Boston, November 1995.

• [3] P. Barth and A. Bockmayr. PLAM: Prolog and algebraic modelling. In *Proceedings of the Fifth International Conference on the Practical Application of Prolog*, pages 73–82, London, UK, 1997. The Practical Application Company.

9.4 Lisa: a front-end for Mona

Abdelwaheb Ayari, David Basin and Andreas Podelski

Motivated by the success of the MONA system by Klarlund and Schwartzbach, a number of research groups (e.g., those of Steffen at Passau, Manna at Stanford and Pnueli at the Weizmann) have implemented verification tools based on a decision procedure for the weak monadic second-order logic with one (WS1S) or two (WS2S) successors. Experience, cf. [2], indicates that while such tools are powerful aids to verification, their usefulness is limited by two major problems. First, the specification language is low-level; writing specifications in WS2S is an experience akin to programming in assembly language. Second, the worst-case complexity of verification is very high (nonelementary time).

We are working on a language and system called LISA tackling those two problems. Our idea is to integrate two concepts from programming languages into WS2S-based specification languages, namely high-level data structures such as records and recursively-defined datatypes. Our integration is based on a new logic whose variables range over record-like trees (feature trees) and an algorithm for translating datatypes into tree automata. LISA compiles specifications written in that logic into WS2S formulas. Thus, when coupled with a decision procedure for WS2S like the MONA system, LISA results in a verification tool that supports both high-level specifications and complexity estimations for the running time of the decision procedure. Up to now, it has been taken for granted that a specification and verification tool based on WS2S algorithmically had to be based on WS2S also semantically. This is the first time that a programming language (i.e. with its own semantics) has been built on top of WS2S. We have written up a draft describing the language and the system and giving a theoretical foundation and analysis [1].

Our present implementation makes it possible to try out our ideas; it is, however, not robust enough to be demonstrated. Right now, it is coupled only with the MONA system; in fact, part of its implementation was done during a visit of Abdu Ayari at BRICS in Aarhus. We envisage coupling it with WS2S-based systems of other groups with whom we are in contact.

References

[1] A. Ayari, D. Basin, and A. Podelski. LISA : A specification language based on ws2s. submitted for publication, April 1997.

• [2] D. A. Basin and N. Klarlund. Hardware verification using monadic second-order logic. In *Proceedings of the 7th International Conference on Computer-Aided Verification (CAV '95)*, volume 939 of *Lecture Notes in Computer Science*, pages 31–41, Liege, Belgium, 1995. Springer.

9.5 Evolutionary algorithms

Jörn Hopf

We have been developing systems for studying evolutionary mechanisms in two areas.

Evolution of fuzzy rules

Unlike, e.g., artificial neural networks, where knowledge is hidden in the architecture of a network and its weighted connections, a 'fuzzy' system provides a readable knowledge base. For complex nonlinear control problems a self-learning a controller has to choose and assimilate the significant parameters to develop a rule base. Evolutionary algorithms are an efficient way to handle this process of self-learning. We have developed a tool [3] for exploring and evaluating the behavior of evolutionary strategies for these rule bases. A first application for the prediction of nonlinear dynamic systems, we have been trying to develop rule sets for predicting the behavior of economic models [1].

Optimizing photo mask layout for Grey-tone Lithography

Silicon micro machining uses the equipment and processes of IC-technology to produce highly miniaturized three-dimensional micro structures like sensors, actuators, membranes, bridges, etc. The process of layer patterning is usually performed by photolithography followed by dry etching. The photolithographic step requires masks which control the grey-tone level during the exposure process. The photomask pattern, with its restrictions caused by mechanical limits of the production process and diffraction during the exposure process could be improved by giving up the fixed rasters that conventional optimization processes need.

As part of the *Laser 2000* project of the BMBF-Program we have been developing a system [2] which applies 'evolution' principles to photomasks in an attempt to optimise 'free' hole arrangement. First results have shown that the average deviation for even a complex structure like a Fresnel-lens can be kept below the wavelength of light. Currently we are able to optimize an area of about $100\mu m^2$ in one minute. Our current research investigates the problems that arise if we try to divide a larger area into computationally tractable sections: interference between neighboring areas means that simple 'sequential' optimization does not work. We are thus working to develop a concept of 'parallel/sequential' optimization.

References

- [1] J. Hopf. Decision making in an economy exploiting fuzzy rules obtained from a genetic algorithm. In M. Jamshidi, M. Fathi, and F. Pierrot, editors, *Soft Computing with Industrial Applications*, volume 5, pages 331–336, Montpellier, France, May 1996. World Automation Congress (WAC), TSI Press.
- [2] J. Hopf. Optimizing photo mask layout for grey-tone lithography. Dagstuhl-Seminar-Report 140, IBFI, IBFI GmbH, Schloss Dagstuhl, D-66687 Wadern, Germany, March 1996.
- [3] J. Hopf. Cooperative coevolution of fuzzy rules. In N. Steele, editor, *Fuzzy Logic and Applications*, pages 337–381, Zurich, Switzerland, February 1997. International Computer Science Conventions (ICSC), ICSC Academic Press.

10 Other Work (Guests)

As well as the work of members of the group proper, work has been carried out by researchers who have visited the group for extended periods.

10.1 Alan Bundy

During his visit to the institute, Alan Bundy developed and investigated a new version of rippling, called *relational rippling* [2]. Rippling is a general heuristic for guiding proof search in rewriting systems, especially in the step cases of inductive proofs [1]. Relational rippling is designed for representations in which value passing is by shared existential variables, as opposed to function nesting. Thus relational rippling can be used to guide reasoning about logic programs or circuits represented as relations.

References

[1] A. Bundy, A. Stevens, F. van Harmelen, A. Ireland, and A. Smaill. Rippling: A heuristic for guiding inductive proofs. *Artificial Intelligence*, 62:185–253, 1993.

• [2] A. Bundy and V. Lombart. Relational rippling: a general approach. In C. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 175–181, Montréal, Canada, 1995. IJCAI, Morgan Kaufmann.

10.2 Evan Tick

Research was conducted in simulating digital circuits with Oz, a concurrent object-oriented programming language. The Oz language has higher-order functions, as well as true concurrency, object-orientation, finite-domain constraints, and multiple-solution search capabilities. These enable several programming paradigms within a single framework. Evan was interested in effective paradigms for the simulation of large networks of simple logic gates. Such an application requires *inherent concurrency* to avoid programmer-induced deadlocks; *objects* to reduce the complexity of circuit description and debugging; *inheritance* to share attributes among gate objects; *first-class functions* to minimize program size; and *logical variables* for synchronization.

The goals of the research were to demonstrate the applicability of the language technology for detailed circuit simulation wherein no timing information is lost. To this end, three simulators were analyzed: (i) **eager**: a baseline system developed previously by MPI and DFKI researchers which represented a signal by a *truth value*, and propagated signals directly from user input. Thus gate delays were not modeled. This code is approximately 75 lines of Oz. (ii) **delayed trace**: the classic technique was implemented by extending a signal to include a *starting time*. Full timing information is retained at the expense of redundant method invocations (output messages are propagated whenever inputs change). This code is approximately 220 lines of Oz, although this includes several gate classes previously undefined. (iii) **selective trace**: the standard technique for removing redundancies was implemented by extending a signal to include *both start and finish times*. The `reduce` method in the core gate class in the previous simulator was optimized to avoid sending output messages whenever possible. This code is approximately 420 lines of Oz.

In addition, a library of flip-flops and adders has been built, and researchers at DFKI built a tcl/tk interface to dynamically display circuit timings. The research results to date indicate that Oz is suitable, for small circuits, for the simulation application. Extensive evaluation of performance has not yet been conducted. Future work includes implementing *inertial delay*, wherein gates do *not* accept inputs that change during arbitrarily short intervals, i.e., very short glitches will not normally be passed by real, inertial gates. Further, we are exploring "simulation for free" using a finite domain variable to represent all the truth values of a signal over its lifetime. Gates will be modeled by appropriate set operations over the domain, and hopefully the the Oz constraint solver can be coerced into producing (unique) solutions.

10.3 Deepak Kapur

Deepak Kapur held discussions with various researchers on diverse topics. With Harald Ganzinger, he discussed chaining inference rules for handling transitive relations. With Harald Ganzinger and David Basin, he discussed saturation and redundancy control based approach for study tractable local inference rules. With Uwe Waldmann, he discussed specialized inference rules for cancellative monoids. With Christoph Weidenbach, he discussed possible relationship between sorted unification and ground reducibility, as well as satisfaction of ordering constraints. With Alexander Bockmayr, he discussed possible application of Groebner basis computations for integer programming. In addition, he held extensive discussions with Dieter Hutter and Werner Stephan (DFKI) on INKA and VSE systems.

He gave a colloquium on maximal extensions of simplification orderings in which he presented joint work with G. Sivakumar of IIT, Bombay (India). He discussed an ordering based on paths, called *mpo*, which is total on ground terms as well as maximal in the sense that two terms s and t are incomparable only if there is a witness (substitution) for which $s > t$ as well as another witness for which $s < t$. He also discussed an efficient quadratic time constraint satisfaction procedure for recursive path ordering.

Inspired by Harald Ganzinger's question, he developed an extension of mpo to make it AC-compatible in the presence of AC symbols. This ordering may specially be of potential use in the Saturate system as it is likely to be maximal and total, since mpo is maximal and total, on AC-terms.

Inspired by specific problems in the elementary first-order theory of cancellative monoids proposed by Uwe Waldmann, he came up with heuristics/approach based on embedding the structure of a cancellative monoid within a group using which it is possible to mechanically search proofs/counter-examples of formulas. There are two ideas: (i) normalization of literals in the group structure, the use of unification for generating constraints and checking assumptions about embedding, (ii) splitting a problem into subproblems based on constraints generated using unification.

10.4 David Plaisted

David Plaisted's research effort was mostly devoted to solving some problems related to rigid E-unification. This technique is typically applied to equality theorem proving in first-order logic in the following way: From a set S of first-order clauses, one constructs an amplification T which consists of a set of copies of the clauses in S. Then one applies rigid E-unification to T. It turns out that S is unsatisfiable (relative to equality) iff there is an amplification T for which this technique will work. David Plaisted had contributed to a proof that the simple rigid E-unification problem is NP-complete. What one needs for theorem proving is the simultaneous rigid E-unification problem. The simultaneous problem was thought to be NP-complete, too, but was recently shown to be undecidable. This is disturbing, because this problem is important for theorem proving with equality. In fact, similar approaches may also work for higher-order logic.

David Plaisted has found a number of common special cases in which simultaneous rigid E-unification is either NP-complete or another (decidable) technique can be substituted for it, for purposes of theorem proving. He has also found techniques that can be substituted for simultaneous rigid E-unification in the general case, but these may require a much larger amplification T. In addition, he found a new proof of the undecidability of simultaneous rigid E-unification. This proof is based on Post's Correspondence Problem and has the interesting feature that all of the equations appearing in the problem are ground equations.

He also spent about a month working on a difficult problem related to simultaneous rigid E-unification. This problem has to do with the size of the amplification T one needs in order for a substitute method to be applicable. It could be that by using a substitute for rigid E-unification, one could get by with an amplification that is not much bigger than that needed for simultaneous rigid E-unification.

He also found a proof of the completeness of a theorem proving method combining clause linking with equality. Clause linking was originally implemented by Shie-Jue Lee from the University of North Carolina at Chapel Hill. On near-propositional problems, this technique often performs spectacularly better than resolution and other common strategies. He recently obtained about a third of the TPTP problems with a Prolog implementation of clause linking, and this is comparable to the performance of other provers implemented in faster languages and possibly with special equality methods.

Geoff Alexander's work is related to a re-implementation of clause linking which only links one literal at a time. Such a re-implementation seems necessary for equality, because linking all of the literals at once leads to too many possibilities when one deals with equality. David Plaisted found a proof of the completeness of a theorem proving strategy combining clause linking and equality. He had presented the idea of the proof in a talk at MPI a couple of years ago. Now he made the proof completely rigorous, prompted by an idea given by Geoff Alexander. Geoff is writing this proof up, and it is planned to publish this proof soon along with results of an implementation. This combination is interesting because it should give the propositional efficiencies of clause linking combined with the equational efficiencies of Knuth-Bendix completion on sets of equations, and thus has a chance of being a very powerful method overall.

10.5 Andreas Herzig

Given a knowledge base KB_1 and new information A expressing that some change, the problem is to construct the resulting knowledge base KB_2. In logic programming and databases, A corresponds to an update, while in plan generation and robotics, it corresponds to the execution of an action. Conditional logics are non-classical logics that allow us to reason about this sort of change. They extend classical logic with conditionals $A > C$, which are read "if A then C". Andreas Herzig is generally interested in automated reasoning methods for non-classical logics that are based on translations into classical logic. A very similar approach has been developed independently at MPI by Hans Jürgen Ohlbach and Andreas Nonnengart.

In earlier work Dov Gabbay and Hans Jürgen Ohlbach have proposed the SCAN algorithm which can be used to semi-automatically generate automated theorem proving procedures for a large class of non-classical logics. SCAN has been implemented and there exists a stable version. The aim of the project was to investigate the extend to which SCAN can be applied to the family of conditional logics, which have strictly more complex semantics than the modal logics to which SCAN has been applied up to now.

As there is no complete overview of the basic systems of conditional logic and their extensions, a first thing to do was to give a "road-map" of the domain. Andreas Herzig has presented a list of basic systems, which are built on three basic semantical structures of increasing strength together with several versions of truth conditions for these systems that have been proposed in the literature.

He has also collected a list of axioms extending these basic systems. An initial problem was that SCAN can only been applied to first-order presentations of the basic systems, whereas almost all of the semantical presentations of conditional logics are second order. This has been solved in all cases by giving equivalent first-order presentations, except that of the so-called Limit Assumption (which is related to well-foundedness). Using such presentations, he has "scanned" the list of axioms for each of the basic systems in collaboration with Hans Jürgen Ohlbach.

The results were satisfying for "small" axioms, while for more complex ones, due to the complex truth conditions, the procedure does not terminate in almost all cases. In particular, he was able to find semantical structures for only one of the three axioms of an extension that has been proposed recently by P. Needham. Recently, an alternative to SCAN has been proposed by Andreas Nonnengart and Andrzej Szałas where termination is ensured, and which is going to be implemented at the MPI. To test it could be a subject of further collaboration. The overview of the basic conditional systems as well as the results of the tests with SCAN are written up as a technical report (see §6.4).

In addition to the original project, Andreas Herzig has worked with E. Weydert on the relation between conditional logic, non-monotonic reasoning and belief revision. They discussed the similarities and technical differences between these formal approaches.

10.6 Murdoch James Gabbay

During his stay, from mid-July to early September 1995, M. J. Gabbay started a C-implementation of the Fixpoint-Algorithm for the elimination of predicate quantifiers as described in 6.4. Due to some unforeseen technical difficulties this implementation could not entirely be finished. However, his intermediate steps may very well serve as a basis for a complete implementation in the near future.

11 Journal and conference activities

11.1 Editorial positions

The editorial activities of staff of the programming logics group are as follows:

Harald Ganzinger is an editor of the journal *Information Processing Letters*, the *Journal of Automated Reasoning*, *Theory of Computing Systems* and *Discrete Mathematics and Computer Science.* He is also the editor of the 1996 proceedings of the conference *Rewriting Techniques and Applications* (RTA'96).

Hans-Jürgen Ohlbach is an editor of *Methods in Computer Science*, and the *Bulletin of the Interest Group in Pure and Applied Logics.* He is also editor of the 1996 conference *Formal and Applied Practical Reasoning* (FAPR'96).

Andreas Podelski is the editor of *Constraint Programming: Basics and Trends*, the proceedings of the 1995 conference *Constraint Programming.*

11.2 Conference activities

The participation of the members of the programming logics group in the organisation of various conferences is as follows.

Program committees

David Basin was a member of the program committees for the 'Formal Aspects of Software Engineering (FASE) colloquium of *Theory and Practice of Software Development* (TAPSOFT'97), the 1996 *International Conference on Automated Deduction* (CADE-13), the 1997 conference *Logic in Computer Science* (LICS'97), the 1996 workshop on *Tools and Algorithms for the Construction and Analysis of Systems* (TACAS'96).

Alexander Bockmayr was a member of the organising committees for the 1996 workshop *Frontiers of Combining Systems* and the sixth *French Conference on Logic and Constraint Programming* (JFPLC'97).

Harald Ganzinger was chair of the program committee for the 1996 conference on *Rewriting Techniques and Applications* (RTA'96) as chair, and also member of the program committees of the 1997 *International Conference on Automated Deduction* (CADE-14), the 1997 conference on *Rewriting Techniques and Applications* (RTA'96), the 1997 conference on *Principles of Programming Languages* (POPL'97) and the 1996 converence on *Algebraic Methodology and Software technology* (AMAST'96).

Andreas Nonnengart was a member of the program committee of the first *International Joint Conference on Qualitative and Quantitative Practical Reasoning* (ECSQARU/FAPR'97).

Hans-Jürgen Ohlbach was a member of the program committee of the 1996 *International Conference on Automated Deduction* (CADE-13) and the 1995 European Summer School on Langage Logic and Information (ESSLI'95).

Andreas Podelski was a member of the program committees of the first *International Conference on Foundations of Software Science and Computation Structures* (FoSSaCS'98), part of *European Joint Conferences on Theory and Practice of Software* (ETAPS), the third *International Conference on Principles and Practice of Constraint Programming* (CP97), the fourteenth *International Logic Programming Symposium* (ILPS'97), the second *International Workshop on Constraint Databases* *and their Applications* (CDB'97) and the fifth *French Conference on Logic and Constraint Programming* (JFPLC'96).

Emil Weydert was a member of the program committees of the 1996 *International Conference on Formal and Applied Practical Reasoning* (FAPR 96) and the the first *International Joint Conference on Qualitative and Quantitative Practical Reasoning* (ECSQARU/FAPR'97).

Organisation of workshops and conferences

Peter Barth was a member of the program committee of the Workshop on Computational Propositional Logic, held at the 19th German Conference on Artificial Intelligence, Bielefeld, 1995.
David Basin organised the 1996 Clam-Inka workshop on inductive theorem proving.

Alexander Bockmayr organised a session on Constraint programming at the *European Conference on Operations Research.*

Jörn Hopf was a member of the committee of the 1996 Dagstuhl seminar on Evolutionary Algorithms and their applications.

Andreas Nonnengart was a member of the organising committee of the first *International Joint Conference on Qualitative and Quantitative Practical Reasoning* (ECSQARU/FAPR'97).

Andreas Podelski was joint organiser of the 1996 Dagstuhl *Seminar on Concurrent Constraint Programming Set Constraints - Second International Workshop* and the dagstuhl Seminar on Concurrent Constraint Programming. October 6 - 10, 1997.

Christoph Weidenbach organised the workshop on propositional logic at the 1995 *German Conference on Artificial Intelligence*, (KI'95).

Emil Weydert organised the third Dutch/German Workshop on Nonmonotonic Reasoning Techniques and their Applications (DGNMR 97).

12 Teaching

Apart from the core Computer Science course on programming languages, the group also contributes a wide range of general and specialist lectures and seminars to to the logic and computation curriculum organised together with DFKI and the Computer Science Department. (Unless specified, courses were taught at the University of Saarbrücken.)
Key: L – Lectures, LE – Lectures and exercises, E – Exercises, S – Seminar,
FoPra – Project class.

Summer Semester 1995

Lambda-Kalkül und Typtheorie H. Ganzinger, D. Basin – LE
Beweistheorie S. Matthews – L
Logik und Wahrscheinlichkeit E. Weydert – S
Constraint logic programming A. Podelski – LE (at the Université de Nice - Sophia Antipolis)
Logical frameworks D. Basin, S. Matthews – L (ESSLLI'95 in Barcelona)

Winter Semester 1995/1996

Programmiersprachen H. Ganzinger – LE
Logical framework and logic engineering D. Basin, S. Matthews – LE
Modellbildung in der mathematischen und logischen Programmierung A. Bockmayr – LE
Einführung in die universelle Algebra und Modelltheorie H. J. Ohlbach – LE

Summer Semester 1996

Praxis des Constraintprogrammierens A. Bockmayr – LE
Logik und Verifikation H. Ganzinger, D. Basin, A. Podelski – S
Recent trends in constraint programming A. Bockmayr – L (at the University of Orléans)
Logical frameworks and philosophical logics D. Basin, S. Matthews – L (ESSLLI'96 in Prague)

Winter Semester 1996/1997

Informatik III H. Ganzinger – LE
Programmiersprachen A. Podelski – LE
Gröbner-Basen A. Bockmayr – LE
Formal engineering of software architectures H. Ganzinger, D. Basin, S. Matthews – S
Verifikation verteilter Systeme H. Ganzinger, D. Basin, A. Podelski – S

Summer Semester 1997

Informatik IV H. Ganzinger – LE
Praxis des Constraintprogrammierens A. Bockmayr – LE
Verifikation H. Ganzinger, A. Podelski – S
Set constraints, their use for program analysis and for solving constraint problems over (feature) trees A. Podelski – L (ESSLLI'97 in Aix-en-Provence)

13 Dissertations and Habilitations

Doctorates

P. Barth *Logic-Based 0-1 Constraint Solving in Constraint Logic Programming* (February 1995)
P. Graf, *Term Indexing*, (July 95)
M. Jaeger *Default Reasoning about Probabilities* (November 95)
A. Nonnengart, *A Resolution-Based Calculus for Temporal Logics* (December 95)
C. Weidenbach, *Computational Aspects of a First-Order Logic with Sorts* (December 96)

Habilitations

A. Bockmayr

Masters Theses in Progress

Michael Christen: *A Calculus of Simplification for Superposition* under H. Ganzinger and J. Stuber.
Thorsten Engel: *Quantorenelimination in Prädikatenlogik 2. Stufe* under H. J. Ohlbach.
Stefan Friedrich: *Integration of a Decision Procedure for Second-Order Monadic Logic in a Higher-Logic Theorem Proving Environment* under D. Basin.
Hichem Harakirti: *Compilation of recursive two-literal clauses into unification algorithms* under H. J. Ohlbach.
Peter Leven: *Integrating Clausal Decision Procedures in a Tactic Based Theorem Prover* under D. Basin.
Rainer Rau: *Decision Making in an Economy Exploiting Fuzzy Rules Obtained from a Genetic Algorithm* under J. Hopf.
Christian Schmolzi: *Cooperative coevolutionäre Entwicklung von Fuzzy-Controllern* under J. Hopf.
Jan Timm: *Testing of the Satisfiability of Ordering Constraints* under C. Weidenbach.

Masters Theses

Ayari Abdelwaheb: *A Reinterpretation of the Deductive Tableaux System in Higher-Order Logic* under D. Basin.
Alexander Bach: *Static Analysis of Functional Programs via Linear Logic* under D. Basin and A. Tönne.
Bernd Gaede: *Superposition extended with sorts* under C. Weidenbach (MPI and Universität Kaiserslautern).
Christoph Meyer: *Parallel Unit Resulting Resolution* under P. Graf.
Erik Mohr: *Resolution-based Calculi for Modal Logics* under A. Nonnengart.
Georg Rock: *Transformations of First-order Formulae for Automated Reasoning* under C. Weidenbach.
Stefan Schlobach: *Untersuchung nicht-monotoner Logiken - Fallstudie zur Automatisierung* under H. J. Ohlbach.
Jan-Georg Smaus: *Finding Resolution K-Transformation* under H. J. Ohlbach.

14 Grants and cooperations

CCL: Construction of Computational Logics

Description

The past two decades have seen a proliferation of different programming styles: functional, logical, constraint-based, object-oriented, among others. More recently, it has been recognized that these styles complement rather than exclude each other by being suitable for particular problem domains. As a consequence, combining programming paradigms has emerged as a significant research direction of its own. Fortunately the modes of computation are in each case firmly based on logic. Computation means simplifying or solving problems represented by logical formulae. Hence combination of programming paradigms means combination of logics in a common logical framework.

Constraint logic programming has been the first entirely successful step towards this ambition of combining logics. Constraints are logical systems specifically tailored to particular theories, for example, numbers, trees, orderings. Constraints allow convenient notations for particular problem domains, efficiency thanks to dedicated solvers, and modularity by isolating the solver from the purely logical part of the computation. Besides new ways of exploiting the above properties, we are also interested in new applications of constraints. Exploiting the structure of (large) constraint logic programs for automating proofs that would simply fail otherwise belongs to the first category. Investigating the use of new systems of constraints for type checking purposes or search guiding information belongs to the second.

According to the above two basic lines, the aims of the CCL-project are the following:

- to investigate specific instances of combination problems for logics and constraints of particular interest;
- to investigate new constraints and to design algorithms for combining existing constraint systems;
- to develop or improve theorem proving techniques for certain logics of special importance for programming, by taking advantage of constraint systems;
- to contribute to a coherent framework for combining programming paradigms and other logical theories, thus enabling the programmer to combine elements of each of them in a unified environment.

Technical Data

Starting date:	July 24, 1992
Duration:	3 years (ended July 23, 1995)
Funding:	ESPRIT Basic Research Action
Staff at MPI f. Informatik:	Peter Barth Alexander Bockmayr Harald Ganzinger Uwe Waldmann

Partners

Cosytec, Paris; DFKI, Saarbrücken; INRIA-Lorraine, Nancy; TU München; CIS, Univ. München; LRI, Université Paris–Sud; RWTH Aachen; Universidad Complutense de Madrid; Universitat Politécnica de Catalunya.

CCL II: Construction of Computational Logics II

Description

TEXT.

Technical Data

Starting date:	September 1996
Duration:	3 years
Funding:	ESPRIT Basic Research Working Group 22457
Staff at MPI f. Informatik:	Harald Ganzinger Alexander Bockmayr Andreas Podelski

Partners

Cosytec, Paris; DFKI, Saarbrücken; INRIA-Lorraine, Nancy; TU München; CIS, Univ. München; LRI, Université Paris–Sud; RWTH Aachen; Universidad Complutense de Madrid; Universitat Politécnica de Catalunya; LIFL; JWG; Hebrew University; University of Wroclaw.

COMPASS: A Comprehensive Algebraic Approach to System Specification and Development

Description

In software technology, concepts, methods and development environments for the construction of data-processing systems from self-contained, generic and reusable components are becoming increasingly important, if not mandatory. The decomposition of systems supports a breakdown of the production process into feasible tasks. Generic and reusable components help to avoid duplications of effort, to ease prototyping, testing and verification and to speed up production processes. The industrial production of generic and reusable software components, however, is only possible under certain conditions:

- The requirements on a component must be specifiable in a precise way.
- The functional behaviour of a component must be determined in a precise way.
- For each component, especially for a critical one, the correctness (meaning that the behaviour satisfies the requirements) must be provable.
- The integration of components into large systems must be supported in such a way that the behaviour and the correctness of the components are preserved.

The state of the art in software technology does not yet allow the systematic development of system components that meet the four demands above. In particular, tools are missing that support such a development with strong requirements on the correctness of the components properly and fully. The algebraic approach to system specification and development is most promising in this respect, but there is still a broad gap between the state of the art of the algebraic approach and the practical needs of the specification of system components.

The main objective of the Basic Research Working Group `compass` is to bridge this gap by further development and consolidation of the algebraic approach in a comprehensive way. In spite of more than fifteen years of research in the algebraic approach, there is a considerable need of clarification, unification, extension and integration of other programming and specification paradigms. Most partners are involved in national and/or European-funded projects, and much of the work described in the objectives will be done in the context of these other projects. The purpose of `compass` is largely to provide opportunities for interaction between the members of these separate projects.

Technical Data

Starting date:	October 1992
Duration:	3 years and 0.5 year extension (ended April 1, 1996)
Funding:	ESPRIT Basic Research Working Group
Staff at MPI f. Informatik:	Harald Ganzinger Hubert Baumeister

Partners

Århus Universitet; Universitat Politécnica de Catalunya, Barcelona; Technische Universität Berlin; Technische Universität Braunschweig; Universität Bremen; Technische Universität Dresden; University of Edinburgh; Universitá di Genova; Universitá degli Studi de L'Aquila; INESC, Lisboa; Technische Universität München; CRIN, Nancy; University of Nijmegen; University of Oslo; Oxford University; CNRS - Université de Paris-Sud; Ecole Normale Supérieure/CNRS, Paris; Ludwig-Maximilians-Universität, München.

MEDLAR II: MEchanizing Deduction in the Logics of prActical Reasoning

Description

MEDLAR II builds on the success of the original MEDLAR project in mechanizing logics of practical reasoning to handle time, action, belief, knowledge and intent. In MEDLAR II the concept of a practical reasoner will be developed, an agent capable of acting autonomously and interacting flexibly with its real world environment. Specific reasoning capabilities are being synthesised for combinations of logics within a general framework for knowledge representation, so that examples of reasoning in natural language dialogue and the planning of robots can be demonstrated.

Technical Data

Starting date:	24 July 1992
Duration:	3 years (ended 23 July 1995)
Funding:	ESPRIT Basic Research Action 6471
Staff at MPI f. Informatik:	Hans Jürgen Ohlbach Christoph Weidenbach

Partners

RISC-Linz; Technische Hochschule Darmstadt; Universität München; ONERA-CERT, Toulouse; Institut Nationale Polytechnique de Grenoble; Universite Paul Sabatier, Toulouse; Universita di Torino; University of Oslo; The Imperial College of Science, Technology and Medicine, London; International Computers Limited.

Automation of Proof by Mathematical Induction

Description

Mathematical induction is required for reasoning about objects or events containing repetition, e.g. computer programs with recursion or iteration, electronic circuits with feedback loops or parameterised components. Thus mathematical induction is a key enabling technology for the use of formal methods in information technology. The goal of this collaboration is to permit an exchange of ideas and cross-fertilization between the leading research groups in this field. The collaboration takes the form of research visits between individuals working on similar or identical problems and more generally annual project seminars. The collaboration is addressing research topics including synthesis of induction rules, generalization, conjecturing of lemmata, strategic search guidance, and applications.

Technical Data

Starting date:	July 1993
Duration:	2 years, 1 year extension (ended 1 July 1996)
Grant:	DAAD/British Council Academic Research Collaboration Grant
Staff at MPI f. Informatik:	David Basin Seán Matthews Luca Viganò

Partners

University of Edinburgh; Universität des Saarlandes; Universität Darmstadt.

ACCLAIM: Advanced Concurrent Constraint Languages: Application, Implementation, and Methodology

Description

The purpose of this project is to further the conceptual, mathematical and practical foundations for concurrent constraint programming, and in so doing, provide a framework for, design and implement advanced computational tools for the development of complex, symbolic computational tasks. The objectives of the four work-packages are:

- To extend the foundations of concurrent constraint programming to account for a substantially richer class of computational phenomena, and to establish connections with graph-grammars.
- To develop efficient constraint techniques to tackle new application areas and to produce extensible general-purpose constraint systems, reactive (incremental) constraint solving, and hypothetical reasoning.
- To develop frameworks and techniques for compile-time analysis and optimization of concurrent constraint programs, to allow efficient execution of programs on a wide variety of target architectures.
- To improve the implementation technology of concurrent constraint languages to be competitive with imperative languages, such as C, on single-processor architectures; and to achieve a high degree of parallel execution on a wide variety of multi-processor architectures.

Technical Data

Starting date:	1 September 1992
Duration:	3 years
Foundation:	ESPRIT Basic Research Action 7195
Staff at MPI f. Informatik:	Peter Barth Alexander Bockmayr Andreas Podelski

Partners

Swedish Institute of Computer Science; Deutsches Forschungsinstitut für Künstliche Intelligenz; INRIA; Katholieke Universiteit Leuven; Universitat Politécnica de Madrid; Università di Pisa; Université d'Aix-Marseille II; RISC-Linz.

PROCOPE: Construction of Non-Classical Logics

Description

In computer science and in particular in the area of artificial intelligence there is an increasing need for logics which allow to reason with knowledge, time and in fact any kind of modality and conditional.

The aim of this project is to develop logic engineering systems which support the examination of such logics and which allow to find suitable and efficient corresponding calculi.

Technical Data

Starting date:	January 1993
Duration:	3 years
Funding:	PROCOPE Programme, Deutscher Akademischer Austauschdienst
Staff at MPI f. Informatik:	Hans Jürgen Ohlbach Andreas Nonnengart Ullrich Hustadt Christoph Weidenbach Renate Schmidt Emil Weydert

Partners

Université Paul Sabatier; Institut de Recherche CNRS.

EDDS: Efficient Data Structures for Deduction Systems

Description

All operations in deduction systems (inference rules, deletion rules, simplifications, and so on) are defined as operations on single objects. Therefore, the performance of a theorem prover crucially depends upon the speed of the basic retrieval operations, such as finding terms that are unifiable with (instances of, or more general than) some *query term.* In order to find resolution partners for a given literal, for example, a theorem prover has to find unifiable literals. Subsumption of clauses can be detected by the retrieval of generalizations or instances of literals of clauses. Even the retrieval of rewrite rules, demodulators, and paramodulants can be accelerated by indexing if the indexing mechanism also supports retrieval in the subterms of the indexed term set.

The importance of the usage of indexing has been shown by the OTTER theorem prover. Due to the consequent usage of Path–Indexing and Discrimination Tree Indexing, this prover became one of the most powerful and fastest deduction systems.

We are developing and implementing new methods such as extended Path–Indexing and Abstraction Tree Indexing in order to speed up term retrieval. Our software is currently being used in the deduction systems STOP (MPI Saarbrücken) and SETHEO (TU München). Very soon we will also embed it into the provers KEIM (Universität des Saarlandes) and DISCOUNT (Uni Kaiserslautern).

Technical Data

Starting date:	1 June 1992
Duration:	3 years (ended 31 May 1995)
Funding:	DFG Schwerpunkt Deduktion
Staff at MPI f. Informatik:	Hans Jürgen Ohlbach Peter Graf

Partners

The project is part of the 'Schwerpunktprogramm Deduktion', the partners of which include: Universitaet Braunschweig; Universitaet Ulm; Universitaet Kaiserslautern; Universitaet Berlin.

TRALOS: Transformation of Logical Systems

Description

We want to investigate methods for finding more or less automatically transformations of axiom systems such that theorems are provable from the original axioms if and only if the transformed theorems are provable from the transformed axioms. Of course the transformations should be such that proving transformed theorems is easier than proving the original theorems. One class of transformations we are investigating are based on representation theorems for algebras.

Technical Data

Starting date:	March 1995
Duration:	2 years
Funding:	DFG
Staff at MPI f. Informatik:	Hans Jürgen Ohlbach Renate Schmidt

CONSOLE: Constraint Solving in Europe

Description

Constraint solving has taken more and more importance and that in automated theorem proving, logic programming, algebraic specifications, as well as for industrial applications like constraint programming in CHIP.

The goal of this project is to facilitate the interactions between European research teams in the field of constraint solving, especially concerning visits and exchanges of young researchers. The work is mainly focused on symbolic constraints (i.e. logic formulae interpreted in some tree structure) and on the application of constraints to constraint logic programming languages. We are thereby particularly interested in equality constraints, ordering constraints, membership constraints, set constraints, and combinations of constraint solvers.

The MPI-part in this project is mainly concerned with

- Paramodulation and Superposition Calculi
- Set Constraints and the Monadic Class
- Non-Linear Constraints in CLP($\mathcal{R}$)
- 0-1 Constraints in CLP($\mathcal{PB}$)

Technical Data

Starting date:	1 January 1995
Duration:	3 years
Funding:	Human Capital and Mobility
Staff at MPI f. Informatik:	Peter Barth Alexander Bockmayr Witold Charatonik Harald Ganzinger Andreas Podelski Uwe Waldmann

Partners

University of Barcelona; University of Lille; ECRC, München; INRIA Lorraine; University of Orsay; Cosytec, Orsay; University of Padova.

DEPRO: Defaults and Probability

Description

This project is concerned with the investigation of quasi-probabilistic approaches to default reasoning, the logical analysis of nonmonotonic inference notions in probability/statistics and their relation with default formalisms, and the integration of probabilistic and default knowledge in a first-order context.

Technical Data

Starting date: May 1, 1996
Duration: 2 years (possible extension for 2 years)
Funding: DFG

Staff at MPI f. Informatik: Harald Ganzinger
Emil Weydert
Manfred Jaeger
Leon van der Torre

Partners

University of Stanford; University of Rotterdam; University of Amsterdam; University of Konstanz; University of Leipzig; IRIT, Toulouse.

Theorembeweisen und Algebra

Description

The commonest techniques of automated theorem proving (e.g. resolution) are not well suited for working with typical algebraic structures, since the large number of inferences possible at any time (because of, e.g., associativity, commutatity or transitivity) means that that the search space can grow uncontrollably.

One possible solution to this problem is to integrate the algebraic axioms into the theorem prover itself, where we can better control their application according to the particular circumstances. The search space can be further controlled by exploiting quantifier elimination techniques and the redundancy criteria of the theory.

The purpose of this project is to exploit and further develop these techniques for new algebraic structures (e.g. abelian monoids or groups with further properties such as torsion-freeness or partial or total ordering).

Technical Data

Starting date:	September 1, 1996
Duration:	2 years
Funding:	DFG
Staff at MPI f. Informatik:	Harald Ganzinger Jürgen Stuber Uwe Waldmann

Partners

The project is part of the 'Schwerpunktprogramm Deduktion', the partners of which include: Universitaet Braunschweig; Universitaet Ulm; Universitaet Kaiserslautern; Universitaet Berlin.

Steuerung der Beweissuche durch Abstraktion

Description

The aim of this project is to investigate abstraction techniques that allow us to direct the search for a proof. We have proposed a variant of ordered resolution with semantic restrictions based on interpretations which are identified by the given atom ordering and selection function. Techniques for automatically and effectively approximating validity (satisfiability) via abstraction in these interpretations are presented and related to methods of soft typing for programming languages. The abstracted interpretations are then used to detect redundant clauses and to select inferences. The framework is shown to be strictly more general than certain previously introduced approaches. Implementation of some of our techniques in the SPASS(see §9.1) prover has lead to encouraging experimental results.

Technical Data

Starting date:	September 1, 1996
Duration:	2 years
Funding:	DFG
Staff at MPI f. Informatik:	Harald Ganzinger Ullrich Hustadt Christoph Mayer Christoph Weidenbach

Partners

The project is part of the 'Schwerpunktprogramm Deduktion', the partners of which include: Universitaet Braunschweig; Universitaet Ulm; Universitaet Kaiserslautern; Universitaet Berlin.

CONTESSA

Description

CONTESSA is a working group that is funded by the European Commission (EC) for three years (1994–1996) to investigate the use of constraints as means for data representation, especially in the context of database management systems. The acronym CONTESSA stands for CON s T raints and E xtended S upport for S torage and A ccess.

The main objective of this working group is to study the storage of constraints in databases, the use of constraint handling techniques in database query and update, and the incorporation of constraints in object data modelling and database specification. The formal properties of constraints in databases and database queries are studied; their role in query optimisation; the consequences of stored constraints on database indexing; constraints on stored objects; and the relationship between constraints as data, active constraints and integrity constraints.

On the practical side, the focus is on supporting the decision making process by the application of constraint handling techniques to large scale geographical data bases.

The technical basis is the integration of a number of advanced technologies in the fields of: geographical information systems (GIS), data base evaluation techniques, constraint logic programming, advanced persistent logic programming languages and advanced indexing techniques for complex objects.

Technical Data

Starting date:	1.2. 1994
Duration:	three years
Funding:	European Commission (EC)
Staff at MPI f. Informatik:	Andreas Podelski

Partners

IC Parc, Imperial College, London; INRIA, Paris; University of Twente, Enschede; Humboldt-University of Berlin; Humboldt-University of Berlin; ECRC, Munich; Technical University of Vienna; University of Athens; University of Birmingham; MPI, Saarbrücken.

15 Publications

Books and monographs

[1] Andreas Podelski, editor. *Constraint Programming: Basics and Trends*, volume 910 of *Lecture Notes in Computer Science*. Springer, Berlin, 1995.

[2] P. Barth. *Logic-based 0-1 constraint programming.* Operations Research/Computer Science Interfaces. Kluwer, Boston, November 1995.

[3] D. Gabbay. *Labelled Deductive Systems, vol. 1.* Clarendon, Oxford, 1996.

[4] H. Ganzinger, editor. *Rewriting Techniques and Applications, 7th International Conference, RTA-96*, volume 1103 of *Lecture Notes in Computer Science*, Berlin, 1996. Center for Discrete Mathematics and Computer Science (DIMACS), Springer. One of the conferences participating in the 1996 Federated Logic Conference (FLoC).

[5] P. Graf. *Term Indexing*, volume 1053 of *Lecture Notes in Artificial Intelligence.* Springer, Berlin, 1996.

[6] H. J. Ohlbach, R. A. Schmidt, and U. Hustadt. Translating graded modalities into predicate logic. In H. Wansing, editor, *Proof Theory of Modal Logic*, Studies in Applied Logic Series, pages 253–291. Kluwer, Dordrecht, The Netherlands, 1996. Also available as Research Report MPI-I-95-2-008.

Journals and chapters

[1] L. Bachmair and H. Ganzinger. A theory of resolution. In J. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning.* Elsevier, Amsterdam, 1997. To appear.

[2] D. A. Basin and S. Matthews. Adding metatheoretic facilities to first-order theories. *Journal of Logic and Computation*, 6(6):835–849, 1996.

[3] D. A. Basin, S. Matthews, and L. Viganò. Labelled modal logics: quantifiers. *Journal of Logic, Language and Information*, 1997.

[4] D. A. Basin, S. Matthews, and L. Viganò. Labelled propositional modal logics: Theory and practice. *Journal of Logic and Computation*, 7, ? 1997.

[5] D. A. Basin, S. Matthews, and L. Viganò. Natural deduction for non-classical logics. *Studia Logica*, 1997. To appear.

[6] D. A. Basin and T. Walsh. A calculus for and termination of rippling. *Journal of Automated Reasoning*, 16(1-2):147–180, 1996.

[7] I. Kraan, D. A. Basin, and A. Bundy. Middle-out reasoning for synthesis and induction. *Journal of Automated Reasoning*, 16(1/2):113–145, March 1996.

[8] A. Nonnengart, H. J. Ohlbach, and A. Szalas. Quantifier elimination for second-order predicate logic. To appear in *Logic, Language and Reasoning. Essays in honour of Dov Gabbay, Part I, Kluwer Academic Press*, 1997.

[9] A. Nonnengart and A. Szalas. A fixpoint approach to second-order quantifier elimination with applications to correspondence theory. To appear in *Essays in Memoriam of Helena Rasiowa*, 1997.

[10] H. J. Ohlbach and R. A. Schmidt. Functional translation and second-order frame properties of modal logics. To appear in *Journal of Logic and Computation.* Available as Research Report MPI-I-95-2-002, 1997.

[11] H. J. Ohlbach and C. Weidenbach. A note on assumptions about skolem functions. *Journal of Automated Reasoning*, 15(2):267–275, 1995.

[12] A. Podelski and G. Smolka. Situated simplification. *Theoretical Computer Science*, 173(1):209–233, February 1997. Preliminary Version in Ugo Montanari, editor, Proceedings of the First International Conference on Principles and Practice of Constraint Programming (CP'95). Springer LNCS 976, 1995.

[13] C. Weidenbach. Unification in sort theories and its applications. *Annals of Mathematics and Artificial Intelligence*, 18(2-4):261–293, 1996.

[14] C. Weidenbach. Spass version 0.49. *Journal of Automated Reasoning*, 1997. To appear.

Conference proceedings

[1] D. Basin, S. Matthews, and L. Viganò. a topography of labelled modal logics. In F. Baader and K. U. Schulz, editors, *Proc. Frontiers of Combining Systems.* Kluwer, Dordrecht, 1996.

[2] D. Basin, S. Matthews, and L. Viganò. a modular presentation of modal logics in a logical framework. In *Proc. Tbilisi Symp. on Language, Logic and Computation.* CSLI, Stanford, to appear.

[3] P. Anderson and D. A. Basin. Deriving and applying logic program transformers. In *Algorithms, Concurrency and Knowledge (1995 Asian Computing Science Conference)*, volume 1023 of *Lecture Notes in Computer Science*, pages 301–318, Pathumthani, Thailand, December 1995. Springer.

[4] A. Ayari and D. A. Basin. Generic system support for deductive program development. In T. Margaria and B. Steffen, editors, *Second International Workshop, TACAS'96: Tools and Algorithms for the Construction and Analysis of Systems*, volume 1055 of *Lecture Notes in Computer Science*, pages 313–328, Passau, Germany, March 1996. Springer.

[5] P. Barth and A. Bockmayr. Modelling 0-1 problems in CLP($\mathcal{PB}$). In *Proceedings of the Second International Conference on Practical Application of Constraint Technology*, pages 1–9, London, UK, 1996. The Practical Application Company.

[6] P. Barth and A. Bockmayr. PLAM: Prolog and algebraic modelling. In *Proceedings of the Fifth International Conference on the Practical Application of Prolog*, pages 73–82, London, UK, 1997. The Practical Application Company.

[7] D. A. Basin and S. Friedrich. Modeling a hardware synthesis methodology in isabelle. In J. von Wright, J. Grundy, and J. Harrison, editors, *Theorem Proving in Higher Order Logics. 9th International Conference, TPHOLs'96*, volume 1125 of *Lecture Notes in Computer Science*, pages 33–50, Turku, Finland, 1996. Turku Centre for Comput. Sci.; Res. Inst. Found. Abo Acad.; Acad. Finland, Springer.

[8] D. A. Basin and H. Ganzinger. Complexity analysis based on ordered resolution. In *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS'96)*, pages 456–465, New Brunswick, New Jersey, USA, 1996. IEEE Comput. Soc. Tech. Committee on Math. Found. Comput.; ACM Special Interest Group on Automata & Comput. Theory; Assoc. Symbolic Logic; Eur. Assoc. Theoret. Comput. Sci, IEEE Comput. Soc. Press.

[9] D. A. Basin, S. Matthews, and L. Viganò. Implementing modal and relevance logics in a logical framework. In L. Carlucci-Aiello, J. Doyle, and S. Shapiro, editors, *Proceedings of the Fifth International Conference on Principles of Knowledge Representation and Reasoning (KR'96)*, pages 386–397, Cambridge, MA, USA, 1996. Morgan Kaufmann.

[10] D. A. Basin, S. Matthews, and L. Viganò. A new method for bounding the complexity of modal logics. In A. Leitsch, editor, *Proceedings of the 5th Kurt Gödel Colloquium on Computational Logic and Proof Theory (KGC'97)*, Lecture Notes in Computer Science, Vienna, 1997. Springer.

[11] A. Bundy and V. Lombart. Relational rippling: a general approach. In C. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 175–181, Montréal, Canada, 1995. IJCAI, Morgan Kaufmann.

[12] F. Cantu, A. Bundy, A. Smaill, and D. A. Basin. Experiments in automating hardware verification using inductive proof planning. In M. Srivas and A. Camilleri, editors, *Proceedings of the Formal Methods for Computer-Aided Design Conference (FMCAD'96)*, volume 1166 of *Lecture Notes in Computer Science*, pages 94–108, Palo Alto, CA, USA, 1996. Springer.

[13] W. Charatonik and A. Podelski. The independence property of a class of set constraints. In E. Freuder, editor, *Principles and Practice of Constraint Programming, Proceedings of the Second International Conference (CP'96)*, volume 1118 of *Lecture Notes in Computer Science*, pages 76–90, Cambridge, USA, August 1996. Springer.

[14] W. Charatonik and A. Podelski. Set constraints with intersection. In G. Winskel, editor, *Proceedings of the Twelfth Annual IEEE Symposium on Logic in Computer Science (LICS)*, Warsaw, Poland, 1997. IEEE Computer Society, IEEE. To appear in June 1997.

[15] H. Ganzinger, C. Meyer, and C. Weidenbach. Soft typing for ordered resolution. In *Proceedings of the 14th International Conference on Automated Deduction (CADE-14)*, Lecture Notes in Computer Science, Townsville, Australia, 1997. Springer. to appear.

[16] H. Ganzinger and U. Waldmann. Theorem proving in cancellative abelian monoids (extended abstract). In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 388–402, New Brunswick, USA, 1996. Springer. Full version: Technical Report MPI-I-96-2-001, Max-Planck-Institut für Informatik, Saarbrücken, Germany, January 1996.

[17] P. Graf. Path indexing for AC-theories. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 718–732, New Brunswick, New Jersey, USA, 1996. Springer.

[18] P. Graf and C. Meyer. Advanced indexing operations on substitution trees. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 553–567, New Brunswick, New Jersey, USA, 1996. AT&T, Bell Labs, Lucent, IEEE, CADE inc, Springer.

[19] M. Hanus. On extra variables in (equational) logic programming. In *Proceedings of the Twelfth International Conference on Logic Programming*, pages 665–679, Kanagawa, Japan, 1995. MIT Press.

[20] J. Hopf. Decision making in an economy exploiting fuzzy rules obtained from a genetic algorithm. In M. Jamshidi, M. Fathi, and F. Pierrot, editors, *Soft Computing with Industrial Applications*, volume 5, pages 331–336, Montpellier, France, May 1996. World Automation Congress (WAC), TSI Press.

[21] J. Hopf. Cooperative coevolution of fuzzy rules. In N. Steele, editor, *Fuzzy Logic and Applications*, pages 337–381, Zurich, Switzerland, February 1997. International Computer Science Conventions (ICSC), ICSC Academic Press.

[22] U. Hustadt and R. A. Schmidt. On evaluating decision procedures for modal logics. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, Nagoya, Japan, 1997. International Joint Conferences on Artificial Intelligence, Inc. (IJCAII) and Japanese Society for Artificial Intelligence (JSAI), Morgan Kaufmann. To appear.

[23] M. Jaeger. Minimum cross-entropy reasoning: A statistical justification. In C. S. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 1847–1852, Montréal, Canada, 1995. Morgan Kaufmann.

[24] M. Jaeger. Representation independence of nonmonotonic inference relations. In L. Carlucci-Aiello, J. Doyle, and S. Shapiro, editors, *Principles of Knowledge Representation and Reasoning, Proceedings of the Fifth International Conference (KR-96)*, pages 461–472, Cambridge, Massachusetts, 1996. Morgan Kaufmann.

[25] M. Jaeger. Relational bayesian networks. To appear in Proceedings of UAI-97, 1997.

[26] M. R. K. Krishna Rao. Graph reducibility of term rewriting systems. In J. Wiedermann and P. Hájek, editors, *Proceedings of Mathematical Foundations of Computer Science*, volume 969 of *Lecture Notes in Computer Science*, pages 371–381, Prague, Czech Republic, August 1995. Springer.

[27] M. R. K. Krishna Rao. Completeness results for basic narrowing in non-copying implementations. In M. Maher, editor, *Logic Programming*, JICSLP, pages 393–407, Bonn, Germany, 1996. The Association for Logic Programming, MIT Press.

[28] M. R. K. Krishna Rao. Modularity of termination in term graph rewriting. In H. Ganzinger, editor, *Rewriting Techniques and Applications*, volume 1103 of *Lecture Notes in Computer Science*, pages 230–244, New Brunswick, July 1996. Springer.

[29] S. Matthews. Extending a logical framework with a modal connective for validity. In *Proceedings of TACS'97*, Lecture Notes in Computer Science, Sendai, Japan, September 1997. Tohoku University, ACM SIGACT, the Information Processing Society of Japan, the Japan Society of Software Science, Association for Symbolic Logic, Springer.

[30] S. Matthews. A practical implementation of simple consequence relations using inductive definitions. In W. McCune, editor, *Proceedings of CADE'14*, Lecture Notes in Artificial Intelligence, Townsville, Australia, July 1997. Association for Automated Reasoning, Springer.

[31] D. A. Basin and N. Klarlund. Hardware verification using monadic second-order logic. In *Proceedings of the 7th International Conference on Computer-Aided Verification (CAV '95)*, volume 939 of *Lecture Notes in Computer Science*, pages 31–41, Liege, Belgium, 1995. Springer.

[32] M. Müller, J. Niehren, and A. Podelski. Inclusion constraints over non-empty sets of trees. In M. Dauchet, editor, *Proceedings of the 4th International Joint Conference on Theory and Practice of Software Development (TAPSOFT)*, volume 1214 of *Lecture Notes in Computer Science*, pages 345–356, Lille, France, April 1997. Springer.

[33] A. Nonnengart. Resolution-based calculi for modal and temporal logics. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 598–612, New Brunswick, USA, 1996. CADE Incorporated, Springer.

[34] H. J. Ohlbach. SCAN-elimination of predicate quantifiers. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 161–165, New Brunswick, New Jersey, USA, 1996. AT&T, Bell Labs, Lucent, IEEE, CADE inc, Springer.

[35] A. Podelski and G. Smolka. Operational semantics of constraint logic programming with coroutining. In L. Sterling, editor, *Proceedings of the 12th International Conference on Logic Programming*, pages 449–463, Kanagawa, Japan, 1995. The MIT Press.

[36] A. Podelski and G. Smolka. Situated simplification. In U. Montanari and F. Rossi, editors, *Proceedings of the First International Conference on Principles and Practice of Constraint Programming (CP'95)*, volume 976 of *Lecture Notes in Computer Science*, pages 328–344, Cassis, France, 1995. Springer.

[37] R. A. Schmidt. Resolution is a decision procedure for many propositional modal logics: Extended abstract. To appear in Kracht, M., de Rijke, M., Wansing, H. and Zakharyaschev, M. (eds), *Advances in Modal Logic '96*. CSLI Publications. The long version is available as Research Report MPI-I-97-2-002, 1997.

[38] Seán Matthews. Implementing FS_0 in Isabelle: adding structure at the metalevel. In J. Calmet and C. Limongelli, editors, *Proc. Disco'96*. Springer, Berlin, 1996.

[39] G. Struth. On the word problem for free lattices. In *Rewriting Techniques and Applications, 8th International Conference (RTA-97), Proceedings*, Lecture Notes in Computer Science, Sitges, Spain, June 1997. Springer. to appear.

[40] J. Stuber. Superposition theorem proving for abelian groups represented as integer modules. In H. Ganzinger, editor, *Rewriting Techniques and Applications, 7th International Conference, RTA-96*, volume 1103 of *Lecture Notes in Computer Science*, pages 33–47, New Brunswick, USA, 1996. Springer.

[41] S. Vorobyov. An improved lower bound for the elementary theories of trees. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 275–287, New Brunswick, USA, 1996. AT&T, Bell Labs, Lucent, IEEE, CADE inc, Springer.

[42] S. Vorobyov. On the bounded theories of finite trees. In J. Jaffar and R. H. C. Yap, editors, *Second Asian Computing Science Conference, ASIAN'96*, volume 1179 of *Lecture Notes in Computer Science*, pages 152–161, Singapore, 1996. Springer.

[43] S. Vorobyov. The first-order theory of one step rewriting in linear noetherian systems is undecidable. In *Proceedings of the 8th International Conference on Rewriting Techniques and Applications (RTA'97)*, Lecture Notes in Computer Science, Sitges, Spain, 1997. Springer. To appear.

[44] S. Vorobyov. The 'hardest natural decidable theory. In *Proceedings of the 1997 IEEE Symposium on Logic in Computer Science (LICS'97)*, Warsaw, Poland, 1997. IEEE Computer Society, IEEE.

[45] C. Weidenbach. Unification in pseudo-linear sort theories is decidable. In M. A. McRobbie and J. K. Slaney, editors, *Proceedings of the 13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 343–357, New Brunswick, USA, 1996. Springer.

[46] E. Weydert. Default entailment. a preferential construction semantics for defeasible inference. In I. Wachsmuth, C.-R. Rollinger, and W. Brauer, editors, *Advances in Artificial Intelligence: Proceedings of the 19th German Annual Conference on Artificial Intelligence*, volume 981 of *Lecture Notes in Artificial Intelligence*, pages 173–184, Bielefeld, Germany, September 1995. Springer.

[47] E. Weydert. Defaults and infinitesimals. defeasible inference by nonarchimedean entropy maximization. In P. Besnard and S. Hanks, editors, *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence (UAI 95)*, pages 540–547, Montréal, Canada, August 1995. Morgan Kaufmann.

[48] E. Weydert. Numeric defaults. about an expressive first-order framework for reasoning with infinitesimal probabilities. In C. Froideveaux and J. Kohlas, editors, *Symbolic and Quantitative Approaches to Reasoning and Uncertainty (ECSQARU 95)*, volume 946 of *Lecture Notes in Artificial Intelligence*, pages 420–427, Fribourg, Switzerland, July 1995. Springer.

[49] E. Weydert. Doxastic normality logic: A qualitative probabilistic modal framework for defaults and belief. In A. Fuhrmann and H. Rott, editors, *Logic, Action, and Information*, pages 152–171. Walter de Gruyter, New York, 1996.

[50] E. Weydert. System J – revision entailment: Default reasoning through ranking measure updates. In D. Gabbay and H. J. Ohlbach, editors, *Practical Reasoning - International Conference on Formal and Applied Practical Reasoning, FAPR'96*, volume 1085 of *Lecture Notes in Computer Science*, pages 637–649, Bonn, Germany, June 1996. Springer.

[51] E. Weydert. Rational default quantifier logic. In D. Gabbay, R. Kruse, and A. Nonnengart, editors, *Qualitative and Quantitative Practical Reasoning*, volume 1244 of *Lecture Notes in Artificial Intelligence*, pages 589–600, Bad Honnef, Germany, June 1997. Springer. to be published.

Workshop proceedings

[1] A. Bockmayr and T. Kasper. Pseudo-boolean and finite domain constraint programming: A case study. In U. Geske and H. Simonis, editors, *Deklarative Constraint Programmierung*, number 297 in GMD-Studien, pages 29–41, Dresden, Germany, September 1996. GDM-Forschungszentrum Informationstechnik GmbH, GMD.

[2] J. Hopf. Optimizing photo mask layout for grey-tone lithography. Dagstuhl-Seminar-Report 140, IBFI, IBFI GmbH, Schloss Dagstuhl, D-66687 Wadern, Germany, March 1996.

[3] M. Jaeger, H. Mannila, and E. Weydert. Data mining as selective theory extraction in probabilistic logic. In R. Ng, editor, *Proceedings of the 1996 SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, Technical Report 96-08, Montreal, Canada, May 1996. Department of Computer Science, British Columbia.

[4] H. J. Ohlbach, R. A. Schmidt, and U. Hustadt. Symbolic arithmetical reasoning with qualified number restrictions. In A. Borgida, M. Lenzerini, D. Nardi, and B. Nebel, editors, *Proceedings of International Workshop on Description Logics'95*, number 07.95 in Rap., pages 89–95, Rome, Italy, May 1995. Dipartimento di Informatica e Sistemistica, Univ. degli studia di Roma.

[5] C. Weidenbach. Sorted unification and its application to automated theorem proving. In M. Hermann and G. Salzer, editors, *Proceedings of the CADE-13 Workshop: Term Schematizations and Their Applications*, pages 67–76, New Brunswick, USA, 1996. Self. To appear in form of a technical report at the University of Wien, Austria.

[6] C. Weidenbach. Unification in sort theories. In K. U. Schulz and S. Kepser, editors, *Proceedings of the 10th International Workshop on Unification, UNIF'96*, volume 96 of *CIS-Bericht*, pages 16–25, Harsching, Germany, 1996. Universität München.

[7] E. Weydert. Default quantifier logic. In I. Niemelä, editor, *Proceedings of the ECAI'96 Workshop on Integrating Nonmonotonicity into Automated Reasoning Systems*, volume 18-96 of *Fachberichte Informatik*, pages –, Budapest, Hungary, September 1996. Universität Koblenz-Landau, Institut für Informatik.

[8] E. Weydert. Qualitative entropy maximization - a preliminary report. In E. Weydert, G. Brewka, and C. Witteveen, editors, *DGNMR97 - Proceedings of the Third Dutch/German Workshop on Nonmonotonic Reasoning Techniques and their Applications*, pages 63–72, Saarbrücken, Germany, 1997. Max-Planck-Institut für Informatik.

Theses

[1] P. Barth. *Logic-based 0-1 Constraint Solving in Constraint Logic Programming.* PhD thesis, Fachbereich Informatik, Universität des Saarlandes, Germany, February 1995.

[2] T. Engel. Quantifier elimination in second-order predicate logic. Master's thesis, Universität des Saarlandes, 1996.

[3] P. Graf. *Term Indexing.* PhD thesis, Universität des Saarlandes, 1995.

[4] M. Jaeger. *Default Reasoning about Probabilities.* PhD thesis, Universität des Saarlandes, November 1995.

[5] C. Meyer. Parallel unit resulting resolution. Master's thesis, Universität des Saarlandes, February 1996.

[6] A. Nonnengart. *A Resolution-Based Calculus for Temporal Logics.* PhD thesis, Universität des Saarlandes, December 1995.

[7] C. Weidenbach. *Computational Aspects of a First-Order Logic with Sorts.* PhD thesis, Universität des Saarlandes, 1996.

Technical reports

[1] L. Bachmair and H. Ganzinger. Ordered chaining calculi for first-order theories of binary relations. Research Report MPI-I-95-2-009, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, October 1995.

[2] L. Bachmair and H. Ganzinger. A theory of resolution. Research Report MPI-I-97-2-005, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, 1997.

[3] P. Barth and A. Bockmayr. Modelling mixed-integer optimisation problems in constraint logic programming. Research Report MPI-I-95-2-011, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, November 1995.

[4] D. A. Basin and H. Ganzinger. Automated complexity analysis based on ordered resolution. Research Report MPI-I-95-2-006, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, November 1995.

[5] D. A. Basin and N. Klarlund. Beyond the finite in automatic hardware verification. Research Report MPI-I-96-2-009, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, October 1996.

[6] D. A. Basin, S. Matthews, and L. Viganò. Natural deduction for non-classical logics. Research Report MPI-I-96-2-006, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, August 1996.

[7] D. A. Basin, S. Matthews, and L. Viganò. Labelled modal logics: Quantifiers. Research Report MPI-I-97-2-001, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, January 1997.

[8] D. A. Basin, S. Matthews, and L. Viganò. Labelled propositional modal logics: Theory and practice. Research Report MPI-I-96-2-002, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, January 96.

[9] H. Baumeister. Using algebraic specification languages for model-oriented specifications. Research Report MPI-I-96-2-003, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, February 1996.

[10] W. Charatonik and A. Podelski. Set constraints for greatest models. Research Report MPI-I-97-2-004, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, April 1997.

[11] H. Ganzinger and U. Waldmann. Theorem proving in cancellative abelian monoids. Research Report MPI-I-96-2-001, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, January 1996.

[12] A. Herzig. Scan and systems of condtional logic. Research Report MPI-I-96-2-007, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, September 1996.

[13] U. Hustadt and R. A. Schmidt. On evaluating decision procedures for modal logic. Research Report MPI-I-97-2-003, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, February 1997.

[14] A. Nonnengart. Auxiliary modal operators and the characterization of modal frames. Research Report MPI-I-96-2-005, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, June 1996.

[15] A. Nonnengart. Strong skolemization. Research Report MPI-I-96-2-010, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, December 1996.

[16] A. Nonnengart and A. Szalas. A fixpoint approach to second-order quantifier elimination with applications to correspondence theory. Research Report MPI-I-95-2-007, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, March 1995.

[17] H. J. Ohlbach and J. Koehler. Reasoning about sets via atomic decomposition. Technical Report TR-96-31, International Computer Science Institute, Berkeley, CA, 1996. Submitted to the *AI Journal*.

[18] H. J. Ohlbach, R. A. Schmidt, and U. Hustadt. Translating graded modalities into predicate logic. Research Report MPI-I-95-2-008, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, May 1995.

[19] D. A. Plaisted. Special cases and substitutes for rigid e-unification. Research Report MPI-I-95-2-010, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, November 1995.

[20] R. A. Schmidt. Resolution is a decision procedure for many propositional modal logics. Research Report MPI-I-97-2-002, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, January 1997.

[21] G. Struth. Non-symmetric rewriting. Research Report MPI-I-96-2-004, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, June 1996.

[22] S. Vorobyov. On the decision complexity of the bounded theories of trees. Research Report MPI-I-96-2-008, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, November 1996.

Part IV

Index

Index