BRICS Newsletter

Basic Research in Computer Science No 11, December 2001

Welcome to the eleventh issue of the BRICS newsletter with lots of information on BRICS activities.

The main event since our last newsletter was the signing of a new BRICS contract with the Danish Research Foundation, merging the previous BRICS Research Centre and the BRICS International PhD School into one unit for the period 2001 – 2006.

This happened at the same time as our previous research centre director *Glynn Winskel* took up a professorship at the University of Cambridge. This was obviously a big loss, but fortunately, we still have very close links to Glynn, professionally and otherwise.

Co-director for many years, *Erik Meineche Schmidt* will soon be busy elsewhere, — Erik has been elected as the new Dean of the Faculty of Science at the University of Aarhus from February 1, 2002.

It has been quite a while since we published our last newsletter, and since then we have had a lot of activities. In fact, the summer of 2001 was particularly busy, with a large number of conferences, symposia, workshops and summer schools, as seen from the large number of reports from BRICS events. Future BRICS activities are listed in the Calendar of Events.

Also, in 2001 we saw the effect on the PhD School initiative with no less than 13 BRICS PhDs graduating from our two universities. All dissertation abstracts can found in this newsletter.

As a PhD School BRICS now has the status as a Marie Curie Training Site, including a number of grants for foreign PhD students to spend time at BRICS as part of their PhD studies. Details on Marie Curie grants, PhD grants, and research positions may be found on our web page.

BRICS has been involved in the start of a number of new research projects, including EU projects such as ALCOM-FT, QAIP, and SECURE, and national centres such as The Bioinformatics Centre, BiRC, at University of Aarhus and The Centre for Embedded Systems, CISS, at Aalborg University. Short descriptions of these may also be found in this newsletter.

Furthermore, this newsletter contains information on new BRICS appointments, publications, courses, etc., including some short news on page 46.

We hope you enjoy reading this newsletter, and if you would like any further information, please don't hesitate to contact us (addresses on the back page).

Mogens Nielsen and *Uffe H. Engberg*

Turn for table of contents.

In this Issue

Coming Events

For details and updates, see the BRICS Activities web page:

www.brics.dk/Activities.

Mobile Calculi

Aarhus, February 2002 *Philippa Gardener*, Imperial College, London, UK, will give a mini-course on Mobile Calculi.

Computable Analysis and its Applications

Aarhus, March 2002 *Margarita Korovina*, BRICS, will give a mini-course on Computable Analysis and its Applications.

Automata and Logic

Aarhus, April 2002 *Igor Walukiewicz*, Labri, Grenoble, France, will give a mini-course on Automata and Logic.

EEF Summer School on Massive Data Sets

June 27 – July 1, 2002, BRICS in Aarhus, organises a summer school on algorithmic issues in Massive Data Sets. It is organised as a part of the European Educational Forum (EEF) foundations series of summer schools, and is supported by the EU Research DG (Human Potential Programme, High-Level Scientific Conferences).

The programme of the school will consist of lectures given by following invited speakers:

Lars Arge, Duke	— Data structures, geometry
Erik D. Demaine, MIT	— Cache-obliviousness
Paolo Ferragina, Pisa	— String algortihms and data structures
Jeffrey S. Vitter, Duke	— Sorting, searching, parallel disks
Norbert Zeh, Carleton	— Graph algorithms

In the afternoons a slot will be reserved for short presentations by the participants.

The school will offer grants for students who wish to attend the school, covering registration, accommodation and local costs. Updated information on the summer school can be found at www.brics.dk/MassiveData02/.

FLoC '02: Federated Logic Conference

The most significant Theoretical Computer Science event in Denmark next year is undoubtedly the third *Federated Logic Conference* to be held in July in Copenhagen.

The participating conferences are:

- CADE, *Conference on Automated Deduction*, July 27–30
- CAV, *Conference on Computer-Aided Verification*, July 27–31
- FME, *Formal Methods Europe*, July 22–24
- ICLP, *International Conference on Logic Programming*, July 29 – August 1
- LICS, *IEEE Symposium on Logic in Computer Science*, July 22–25
- RTA, *Conference on Rewriting Techniques and Applications*, July 22–24
- TABLEAUX, *Automated Reasoning with Analytic Tableaux and Related Methods*, July 30 – August 1

A large number of workshops are affiliated these FLoC conferences.

FloC is hosted jointly by the IT University of Copenhagen, the Technical University of Denmark and the University of Copenhagen. The conference will be held at the University of Copenhagen.

The following BRICS people are involved in FLoC:

- *Kim G. Larsen* is in the Program Committee of CADE as well as Co-Chair of the CAV Program Committee.
- *Ulrich Kohlenbach* is LICS Conference Chair.
- *Zoltán Ésik* and *Anna Ingólfsdóttir* are Organisers of FICS, *Fixed Points In Computer Science*, a workshop affiliated LICS.

More about FLoC at floc02.diku.dk.

COMPLEXITY 2003 – 18th IEEE Conference on Computational Complexity

In the summer of 2003, BRICS, will host the 18th Annual IEEE Conference on Computational Complexity. The conference is held in Europe every third year. Chair of the Organising Committee is *Peter Bro Miltersen*, BRICS.

Reports on Events

BRICS PhD Workshop 1999

On October 18–20, 1999, BRICS had a retreat on the theme of "meta-issues" of research and PhD studies in computer science. The agenda included sessions on issues like:

- How to give write a research paper and give a talk
- Responsible Conduct in Research
- "Drivers" for Basic Research in Computer Science

The special invited guest was *Jan Rubæk Pedersen*, CCI Europe, Denmark, who talked on *Working in Industry*.

The retreat was held at the nice University course facilities at Sandbjerg Manor, Sønderborg (the very south of Jutland).

The complete agenda and slides can be found at www.brics.dk/Activities/99/PhDWorkshop.

The Java Security Model

February 29 and March 7, 2000, *Tommy Thorn*, BRICS, gave two double lectures on The Java Security Model.

Games and Free mu-Lattices

June 7 and 9, 2000, *Luigi Santocanale*, BRICS, gave two double lectures on Games and Free mu-Lattices.

Proof Theory for the Working Category Theorist!

June 19 and 21, 2000, *Robin Cockett*, Department of Computer Science, University of Calgary, Alberta, Canada, gave three lectures on Proof Theory for the Working Category Theorist.

Figure 1: The PMCO participants.

Figure 2: Participants of the BRICS PhD workshop 2000. In the back ground one the newly restored wings of the Sandbjerg Manor.

Workshop on Probabilistic Methods in Combinatorial Optimisation

In the week of August 28–31, 2000, BRICS hosted a workshop on Probabilistic Methods in Combinatorial Optimisation (PMCO) in Aarhus.

The workshop aimed at bringing together distinguished researchers in the field, including many young people. In order to foster the exchange of ideas, the participants were a mixture of researchers from the fields of probability theory and combinatorial optimisation.

The 18 talks of about an hours length were attended by some 45 participants— see Figure 1 and Figure 3. The journal Random Structures & Algorithms will devote a special issue to the event.

The full programme with speakers as well as more pictures from the event can be found at www.brics.dk/Activities/00/PMCO/.

The workshop was organised by *Michal Karonski*, *Tibor Jordán*, *Erik Meineche Schmidt*, and *Alessandro Panconesi*.

Figure 3: *Michael Steele*, *Kati (Katalin) Marton*, and *Devdatt Dubhashi* looking at a “PMCO problem”.

Model Checking Java Source Code Using the Bandera Tool Set

October 2-4, 2000, *John Hatcliff*, Department of Computing and Information Sciences, Kansas State University, USA, gave three double lectures on Model Checking Java Source Code Using the Bandera Tool Set.

BRICS PhD Workshop 2000

On October 18–20, 2000, BRICS had a retreat focusing on a series of presentations by the individual BRICS research groups on their research areas and BRICS contributions, providing a foundation for ongoing discussions on the future research themes of BRICS.

The special invited speaker, *Thiemo Krink*, University of Aarhus, talked on *Bio-computation — What computers can learn from real bugs*.

As in 1999, the retreat was held at Sandbjerg Manor, Sønderborg.

Figure 2 shows the participants in the BRICS PhD workshop. Agenda, slides and pictures from the workshop are available at www.brics.dk/Activities/00/Retreat.

Challenges in Combinatorics

In the period October 24 – November 20, 2000, *Zsolt Tuza*, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, Hungary, gave five double lectures on Challenges in Combinatorics.

Growing out of the course, Zsolt Tuza subsequently made the lecture notes *Unsolved Combinatorial Problems, Part I*, see [LS-01-1].

Hyper-searching the Web

May 9-11 and 14, 2001, *Devdatt Dubhashi*, Computing Science, Chalmers University of Technology and Göteborg University, Sweden, gave four double lectures on Hyper-searching the Web—Or How Algebra and Probability can help you surf better.

Figure 4: *Dana Scott*, *John Reynolds*, and *Andrzej Filinski* at an ICC ’01 break.

Graph Colourings

May 17, 18, 29 and 31, 2001, *Zsolt Tuza*, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, Hungary, gave four double lectures on Graph Colourings.

ICC ’01

May 20–21, 2001, the Third international workshop on Implicit Computational Complexity-2001 was held at the University of Aarhus. 29 attended the workshop. Topics of interest are characterised in the following.

The synergy between Logic, Computational Complexity and Programming Language Theory has gained importance and vigour in recent years, cutting across areas such as Proof Theory, Computation Theory, Functional Programming, and Philosophical Logic. Several machine-independent approaches to computational complexity have been developed, which characterise complexity classes by conceptual measures borrowed primarily from mathematical logic. Collectively these approaches have been dubbed “Implicit Computational Complexity”.

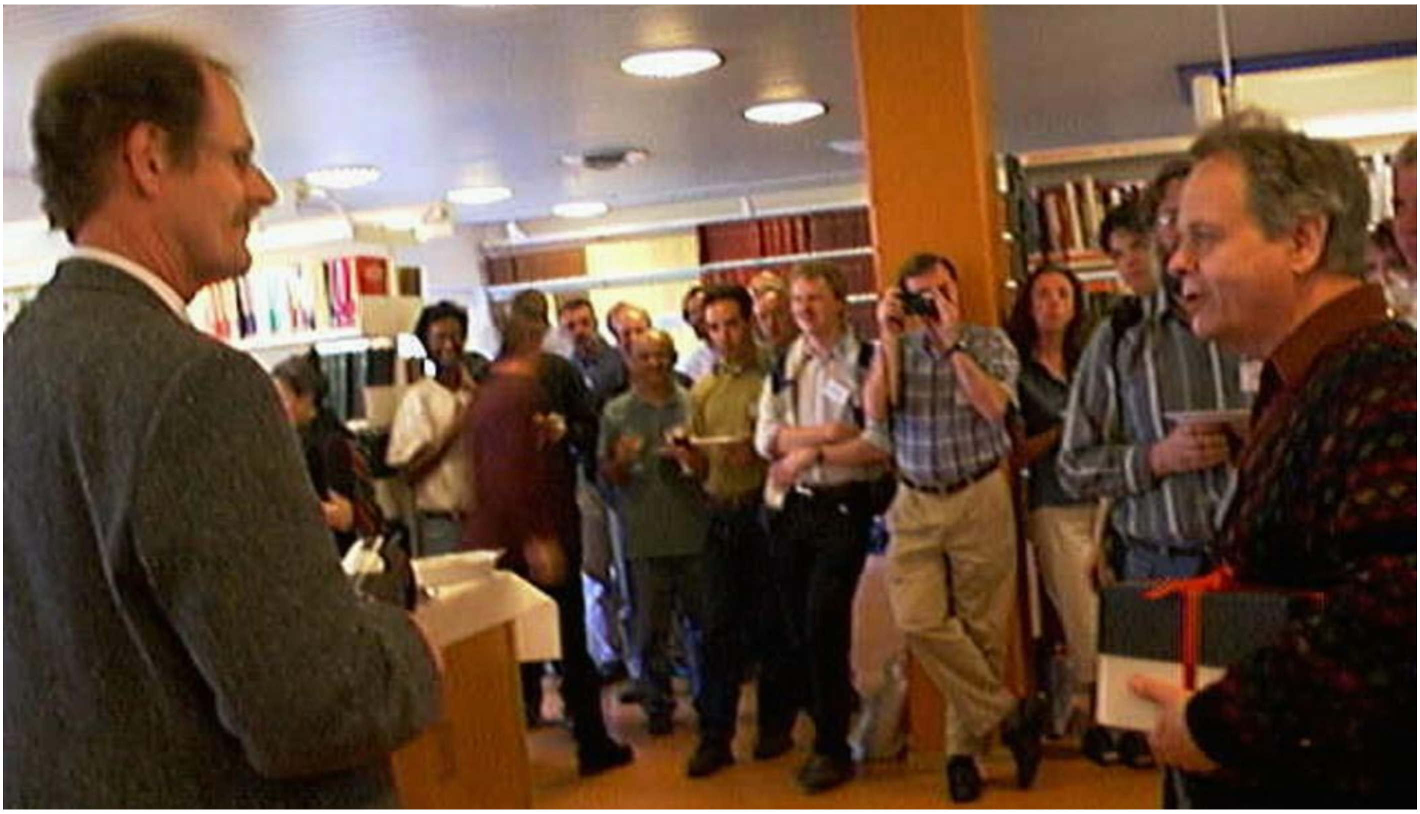


Figure 5: *John Reynolds* congratulating *Neil Jones* at the reception on May, 23.

Practically, implicit computational complexity provides a framework for a streamlined incorporation of computational complexity into areas such as formal methods in software development and programming language theory. In addition to research reports on theoretical advances in implicit computational complexity, practical contributions bridging the gap between Computational Complexity and Programming Language Theory are therefore of particular interest.

The proceedings of ICC '01 have been published in the BRICS Notes Series [NS-01-3].

Further information on the workshop at `www.dcs.ed.ac.uk/home/mxh/ICC01.html`.

PADO II

May 21–23, 2001, the Second Symposium on Programs as Data Objects was held at the University of Aarhus.

PADO II brought together 53 researchers working in the areas of programming and programming languages. The symposium focused on techniques and supporting theory for treating programs as data objects. Technical topics included, but were not limited to:

- Program manipulation: program specialisation, type specialisation, partial evaluation, normalisation, reflection, rewriting, run-time code generation, self-application.

Figure 6: *John Hughes*, PADO II

Figure 7: Participants of MFPS XVII.

- Program analysis: abstract interpretation, constraints, type inference, binding-time analysis.
- Theoretical issues in representing and classifying programs: semantics, algorithmics, logics.
- Applications: interpretation, compilation, compiler generation, verification, certification, meta-programming, instrumentation, incremental computation, staging, prototyping, debugging.

PADO II was held on the occasion of Professor *Neil Jones*' 60th birthday, and at his request, it was held as a full-fledged research meeting. In 1985, Neil Jones and Harald Ganzinger organised the first instance of PADO in Copenhagen. Therefore, we invited them both to give the opening talk and the closing talk of PADO II.

The proceedings of PADO II have been published as LNCS Volume 2053 of Springer-Verlag. See www.brics.dk/pado2/.

MFPS XVII

May 23–27, 2001, the Seventeenth Conference on the Mathematical Foundations of Programming Semantics was held at the University of Aarhus. MFPS attracted 66 participants.

The MFPS conferences are devoted to the areas of mathematics, logic and computer science that are related to the semantics of programming languages. The series has continuously stressed to providing a forum where both mathematicians and computer scientists can meet and exchange ideas about problems of common interest.

The preliminary proceedings of MFPS XVII have been published in the BRICS Notes Series [NS-01-2], and the final version has been published as Volume 45 in the series 'Electronic Notes in Theoretical Computer Science' of Elsevier Science Publishers.

Read more about the Conference at www.math.tulane.edu/mfps17.html.

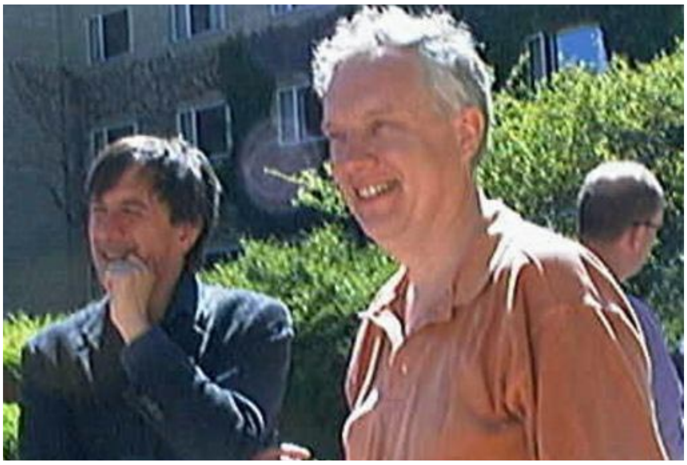


Figure 8: *Gordon Plotkin* and *Steve Brookes* at a MFPS break.

Figure 9: Participants of Summer School on Logical Methods at the beach.

Summer School on Logical Methods

On June 25 – July 6, 2001, Ulrich Kohlenbach as part of the European Educational Forum, organised a summer school on Logical Methods at Aarhus with BRICS support. The were around 60 participants, mainly PhD students, coming from all round the world (including Colombia, Israel, Russia, Armenia, India, and USA).

From one of the participants, *Ivar Rummelhoff*, from University of Oslo, Norway, we have received the following report.

If I were to characterise this summer school in one word, I would say 'variation'. First of all, there was a variation in the subjects, ranging from complexity theory to type theory and category theory, from proof theory and computability theory to model checking and logic in computer security. This clearly shows how 'logical methods' play an important role in almost every branch of theoretical computer science.

Not only was there a great variation in subjects and styles of presentation; the students were also a very diverse group, coming from all over the world and from different backgrounds. The presence of so many different perspectives was very stimulating; and there was plenty to learn, not only from the lecturers, but also from the other students.

Of course, the diversity among the students posed a challenge to the lecturers. But they handled this challenge well, starting each series of lectures at a basic level, gradually increasing the pace and in many cases reaching quite advanced material by the end. Once again the key word was 'variation'.

Certain things, however, were constant: Every school day was well organised, and the organisers were always kind and helpful. On top of that, the sun was shining all the time! This stimulated various forms of outdoor activities, such as football, barbecue and even swimming (in constantly cold water). Besides, there were two very nice excursions, which were welcome breaks in a rather compact programme of lectures.

Among the Scandinavians, the Danes are said to be the most relaxed and friendly. However that may be, the atmosphere at the summer school was certainly such; and the lecturers were always open for questions and comments – even during the wonderful lunches, equally gratifying for vegetarians and non-vegetarians.

As a Norwegian mathematician with some background in computer science, I had few

Figure 10: The Polish chapter entertaining at the sommer school dinner.

communication problems at BRICS, the only exception being my visit to a Turkish hairdresser nearby. Now, five months later, the effects of this unlucky encounter are no longer visible. What remains is a better understanding of logical methods in computer science, a number of new friends and contacts, and many lasting memories. Who can ever forget the great social dinner, with all its ‘extraordinary’ performances? ☶

Impressions of CONCUR ’01

Hans Hüttel

Writing this early December, it can be difficult to recall the CONCUR ’01 conference held in Aalborg in late August 21–24. The tragic events in the United States only a couple of weeks later have definitely overshadowed the memories of this pleasant conference which once again brought a large contingent of researchers (156, to be precise) within theoretical computer science to Aalborg.

As always, the conference provided everyone with the chance to live out the conference experience of meeting and talking to former colleagues, kindred spirits and long lost collaborators or – if you were a young researcher – to see the faces and hear the voices behind the names found in conference proceedings and in journals.

Figure 11: *Robin Milner* giving an interview to DR (Danish Broadcasting Corporation).

Unlike the earlier major conference that the Aalborg branch of BRICS hosted, viz. ICALP’98, CONCUR ’01 and the associated workshops were held at Aalborg University itself and in the vicinity of the Department of Computer Science.

The CONCUR conference series grew out of the ESPRIT project of the same name and has since become the most important conference within its area, as witnessed by the 78 papers that were submitted this time. The field of concurrency theory has widened quite a bit in recent years as there are now also quite a few more applied approaches to the subject based in case studies and implementable verification techniques – a definite sign that the area has matured. This year’s conference was no exception.

The satellite workshops

I am bound to be heavily biased, having been so deeply involved myself. After all, my main responsibility (apart from being the CONCUR ’01 webmaster) was that of organising the pre- and post-conference workshops of which there were five: EXPRESS *(Expressiveness in Concurrency)*

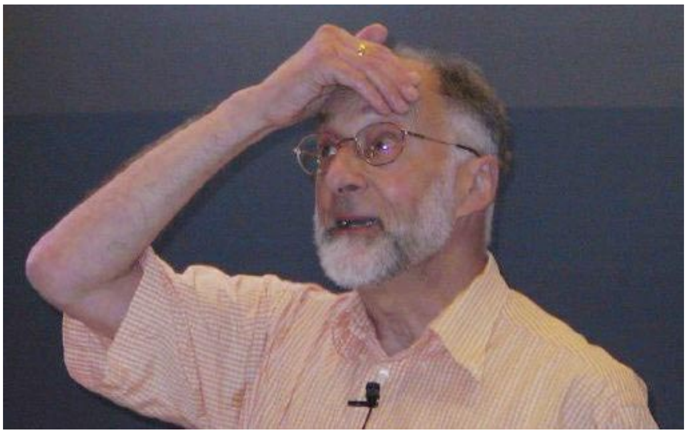


Figure 12: *Robin Milner* at his talk.

and RT-TOOLS *(Real-Time Tools)* on 20 August and GETCO *(Geometric and Topological approaches to Concurrency),* FATES *(Formal Approaches to Testing of Software)* and MTCS *(Models for Time-Critical Systems)* on 25 August.

In my opinion the workshops went quite well and were definitely well-attended; a minor glitch came on Saturday when the main entrance to the NOVI cafeteria (which provided the lunches for all of CONCUR '01) had somehow been locked and the workshop participants had to find a different entrance. In such situations it is always important to stay calm and collected when looking for an alternative solution; I hope we succeeded and would like to use the opportunity to apologise for any inconveniences caused.

The invited speakers

We were fortunate enough to have a number of distinguished invited speakers. I hope I do not offend anyone by saying that we were especially pleased to have *Robin Milner* (Cambridge University, UK) as one of them. After all, Robin is among the founders of the by now extremely wide-ranging subject of concurrency theory. It is no exaggeration to say that, had it not been for Robin's profound insights and deeply original contributions, the CONCUR conference series would probably not have seen the light of day. Not surprisingly, of course, as Robin is a recipient of the ACM Turing Award.

Figure 13: *Bengt Jonsson* playing the piano.

Robin gave us an interesting talk on the topic of bigraphs, the latest development in his work on unifying models of concurrency. Robin's talk was the most well-attended talk of the entire conference as quite a few individuals who were definitely not participants of CONCUR showed up!

The other invited speakers were

- *Steve Schneider*, University of Royal Holloway, UK, who gave an interesting overview of his work on applying CSP techniques to the verification of security protocols.
- *Bengt Jonsson*, University of Uppsala, Sweden, who told us about his work on verifying probabilistic processes.
- *Shankar Sastry*, UC Berkeley, USA, who provided us with an important alternative point of view, namely that of automatic verification as seen through the eyes of control theory.

CONCUR '01 also hosted two *tutorials*: one by *Joost-Pieter Katoen* and *Holger Hermanns* on performance modelling and another by *John Hatcliff* and *Matthew Dywer* on model checking Java programs.

Figure 14: *Arne Skou*, *Ilaria Castellani*, *Kim G. Larsen*, and *Anna Ingólfsdóttir* where the oceans and everybody meets.

Accepted papers

It would be an arduous task, were I to summarise all 32 papers accepted for this year's CONCUR. So let me simply say that the contents of the papers ran the whole gamut from foundational papers on process calculi and transition graphs to applied topics within the area of automatic verification techniques – reflecting the state of concurrency theory at the beginning of the 21st century.

The award for the *best student paper* went to Majumdam who wrote *Symbolic Algorithms for Infinite-State Games* with *Henzinger* and *de Alfaro*. Members of this writers' collective were extremely well-represented in this year's proceedings.

Social events

The reception and dinner at the North Jutland Museum of Modern Art gave us as organisers a rare opportunity to show everyone that one of our invited speakers, *Bengt Jonsson*, is also a classically trained pianist. Bengt treated us to pieces by Chopin and Albeniz (and the following morning to an interesting talk but that is another story.)

The other social event, the traditional Wednesday afternoon excursion, took us to three of the best-known sights of Northern Jutland. First, the North Sea Museum in Hirtshals with the ambience of seals. Next, Råbjerg Mile – a huge wandering dune which is a really a tiny desert. And finally, Skagen, the northernmost tip of Denmark with a rare chance to dip one's feet where Skagerak meets Kattegat. The programme ended with a conference dinner in Skagen. One of the definite highlights of the conference dinner was *Prakash Panangaden*'s dinner speech, peppered as it was with witty observations and interesting anecdotes about colleagues (who shall remain nameless here – their names can be found in the proceedings . . .)

The excursion was well planned by the tourist office. Still, in retrospect, one might have considered a slightly less extensive programme with a little more time to dwell on the interesting sights of Northern Jutland.

Pictures from CONCUR '01

Our very own *Emmanuel Fleury* took a long series of snapshots during the conference. His pictures are now on display at the CONCUR '01 website: `concur01.cs.auc.dk`. ☷

ALGO 2001

Gerth Stølting Brodal

August 28–31, 2001, BRICS in Aarhus was hosting ALGO 2001 which covered the jointly organised algorithmic conferences and workshops:

ESA 2001 - The ninth Annual European Symposium on Algorithms. The Symposium covers research in the use, design, and analysis of efficient algorithms and data structures in computer science, discrete applied mathematics and mathematical programming.

WAE 2001 - The fifth Workshop on Algorithm Engineering. The workshop covers research in all aspects of algorithm engineering. The goal is to present recent research results and to identify and explore directions of future research.

WABI 2001 - The first Workshop on Algorithms in BioInformatics. The workshop covers research in all aspects of algorithmic work in bioinformatics. The emphasis is on discrete algorithms that address important problems in molecular biology, that are founded on sound models, that are computationally efficient, and that have been implemented and tested in simulations and on real datasets. The goal is to present recent research results, including significant work-in-progress, and to identify and explore directions of future research.

ALGO 2001 was on August 27 preceded by the satellite event

ARACNE 2001 - The second International Workshop on Approximation and Randomised Algorithms in Communication Networks

The events took place at the Department of Computer Science, University of Aarhus and were organised by BRICS, Aarhus. ALGO 2001 attracted a total of 178 participants from 25 different countries, with the top-five countries being Denmark (37), United States (27), Germany (25), France (14) and Italy (13).

In total there were 94 talks during the five days. The presentations of contributed papers were given in two tracks, one track designated to ESA 2001 and one track alternating between presentations of papers contributed to WAE 2001 and WABI 2001.

There were six invited talks, with speakers both from academia and industry. The invited talks where common to all ALGO 2001 event. *Susanne Albers*, Freiburg University, gave an invited talk on Some Algorithmic Problems in Large Networks, *Lars Arge*, Duke University, on External Memory Data Structures, *Andrei Broder*, AltaVista, on Some Algorithmic Challenges in Web Search, *Herbert Edelsbrunner*, Duke University, on Bio-Geometric Modelling, *Jotun Hein*, University of Aarhus, on Algorithms for Statistical Multiple Alignment, and *Uri Zwick*, Tel Aviv University, on Exact and Approximate Distances in Graphs. Unfortunately the seventh announced invited speaker, Gene Myers, Celera Genomics, had to cancel his participation just prior to ALGO 2001.

The proceedings of ESA 2001, WAE 2001, and WABI 2001 are available in the Lecture Notes in Computer Science series, as volumes 2161, 2141, and 2149, published by Springer-Verlag, Heidelberg. The proceedings are online available via the homepage of ALGO 2001 (www.brics.dk/algo2001).

On the business meeting of ALGO 2001 the program committee chair of ESA 2001, *Friedhelm Meyer auf der Heide*, awarded the two best student papers of ESA 2001: “Cuckoo Hashing” by *Rasmus Pagh* and *Flemming Friche Rodler*, BRICS, and “Algorithms for Efficient Filtering in Content-Based Multicast” by *Stefan Langerman*, *Sachin Lodha*, and *Rahul Shah*, Rutgers University. On the business meeting it was decided that the format of ALGO 2001 would be continued next year in Rome, in the context of ALGO 2002 (www.dis.uniroma1.it/~algo02/). It was decided, from 2002, to merge ESA and WAE to one conference with two tracks: *Design and Analysis* and *Engineering and Applications*.

During ALGO 2001 there were two social events. On the evening of Wednesday August 30, all participants were invited by the City of Aarhus to a reception in the Concert Hall and on Thursday August 31 we had a conference dinner in the Mathematical Canteen at the University of Aarhus, including a traditional song contest.

The submissions of ESA 2001, WAE 2001, and WABI 2001 were handled electronically using the server and software offered by ACM SIGACT. The program committee reviewing process was also handled electronically using the ACM SIGACT software. For ESA 2001 there was a final program committee meeting at the Heinz Nixdorf Institute, University of Paderborn. The number of submissions for the three conferences was respectively 102, 25, and 50, and the number of accepted papers was respectively 41, 15, and 23.

ALGO 2001 was sponsored by BRICS and the European Association for Theoretical Computer Science (EATCS).

Aalborg Wing Retreat

On October 25–26, 2001, The Aalborg wing of BRICS had a retreat at Dronninglund Castle (16 participants). The Department of Computer Science, Aalborg University, has recently been evaluated by an international panel with respect to its research contributions during 1995–2000. Also, the computer science study has recently been changed in a way which offers new possibilities to recruit Phd students. The purpose of the retreat was to summarise the relevant parts of the evaluation, and also to identify future research strategies.

The research evaluation of BRICS@Aalborg turned out very well: The quality of the research was highly appreciated, and also a number of important recommendations for future strategies were given by the evaluation panel[1]. These included focusing on the combination of practice and foundations (e.g. for embedded systems), industrial involvement, dissemination of research results into the graduate studies and also on the distribution of responsibility to young researchers. The discussion concluded that the recommendations are already being taken into account. Also a number of common research areas within the Aalborg wing were identified.

Figure 16: Participanst of the Aalborg Wing Retreat.

Externally funded projects were summarised. These include CJT (centre for agricultural technology), Siemens (testing), Mindpass (cluster computing), DeCode (Islandic company on bioinformatics), CISS (centre for embedded software systems, further details in this newsletter) as well as pending applications on embedded systems (EU) and GRID computing (Danish research councils).

The final year of the computer science study now includes the possibility to prepare the enrolment as Phd student. It was agreed to pursue this possibility by offering advanced courses in coordination with the BRICS Phd School.

A session included inputs from the visiting researchers on how to improve the administration and the daily interaction as seen from a visitors point of view. Several important points were raised - including the establishing of an informal meeting point.

During the retreat dinner, several speeches were held for *Bertrand Jeannet*, who now has left his post-doc position at BRICS in order to enter a permanent position at INRIA, Rennes, France.

[1] The evaluation report may be ordered from the department administration.

Figure 15: Participants of the BRICS PhD workshop 2001.

Events in Security Protocols

September 11 and 13, 2001, *Glynn Winskel*, Computer Laboratory, University of Cambridge, England, gave two double lectures on Events in Security Protocols.

BRICS PhD Workshop 2001

On October 22–23, 2001, BRICS had a retreat on the theme of BRICS PhD-studies and Research Activities. The retreat was held at the Søhøjlandet near Gjern (the middle of Jutland).

This year there were two invited speakers

- *Poul V. Thomsen*, Director of Centre for Studies in Science Education, University of Aarhus, who talked on Tutoring, and
- *Birgit Pedersen*, head of the DAIMI Library, who talked on Literature search.

Figure 15 shows the participants in the BRICS PhD workshop. Agenda, slides and pictures from the workshop are available at www.brics.dk/Activities/01/Retreat.

Ongoing Activities

A part from the mini-courses and the broad seminar series in both Aalborg and Aarhus, BRICS has a number of ongoing activities.

In the specialised seminars,

- ALCOM Seminars, *Algorithms and Complexity Theory*,
- CCT Seminars, *Category Theory and its Applications*,
- CoW Seminars, *Concurrency Workshop*,
- *Cryptology Seminars*, and
- *Logic and Semantics Seminars*,

people with particular interests meet on a regular basis. Another ongoing activity is the

Juniorklubben

Juniorklubben is an organisation for the PhD students at BRICS and Daimi. We have regular informal talks given by students and invited speakers, as well as social events.

The last two years have been as busy and as successful as ever; some of the student talks given last year included

- *Daniele Varacca* with Do categoricians go to Cantor's paradise?,
- *Bernd Grobauer* with The Ways of Paradox and
- *Rasmus Pagh* with What's randomness got to do with it?

We also featured an inspiring invited talk by *Anders Yeo* titled "The optimal strategy in blackjack", although the subsequent trips to the casino weren't official outings. We did, however, have a very hyggelig dinner at "Chez Tony".

Some of the talks given this year included

- *Pablo Arrighi* with "Quantum Computation for Dummies",
- *Bartek Klin* with Charles Babbage - the grandfather of computers and
- a great invited talk by *Olvier Danvy* entitled The Art of Thinking.

The social occasions included a great dinner at "La Fiesta" which was also an occasion to say goodbye to our outgoing president *Bernd Grobauer*. We have also discovered that powerful machines such as *Mads Jurik*'s laptop are suitable not only for science but also for watching movies, especially when connected to some very expensive university projection equipment.

Juniorklubben maintains a web page at www.brics.dk/Juniorklubben at which one can find out about upcoming talks and browse through the abstracts of past talks.

Newly Appointed Researchers, Guests and PhDs

Ronald Cramer
Ronald Cramer received his PhD degree from the Computer Science Department, University of Amsterdam, 1997, and his MSc degree from the Mathematics Department, Leiden University, 1992. Before joining BRICS in April 2000, Cramer held positions at ETH Zürich, 1997–2000, first as a post-doc and later as a senior researcher. Prior to that, he held positions at CWI Amsterdam, 1992–1997. Cramer's main research interest is cryptography. He received the 1998 Christian Huygens Award of the Royal Netherlands Academy of Sciences and Arts (KNAW) for his PhD thesis. Cramer has served on several program committees of international conferences such as ICALP, Eurocrypt and Crypto and he is a member of the editorial board of Journal of Cryptology.

Zoltán Ésik
Zoltán Ésik has been visiting BRICS and the Department of Computer Science, Aalborg, since May 1, 2001. He holds a permanent position at the Computer Science Department, University of Szeged, Hungary. His main research interests are the equational theory of fixed points and its applications to axiomatisation questions in computer science. He is also interested in automata and tree automata, formal languages and generalisations, categories and logic.

Emmanuel Fleury
Emmanuel Fleury got his PhD thesis on *Updatable Timed Automata* at École Normale Supérieur of Cachan (France). He studied mathematics (Functional Analysis) and received his

Masters degree in Computer Science at the University of Orsay (Paris XI). His research interests cover both theoretical and practical topics in verification (model-checking, static analysis, theorem proving), complexity theory and algorithmic, networks and operating systems. He joined BRICS in January 2001.

Bertrand Jeannet

Bertrand Jeannet obtained his PhD thesis,*Dynamic Partitioning in Linear Relation Analysis and Application to the Verification of Synchronous Programs*, from Institut National Polytechnique de Grenoble in 2000. He held a post doc grant from Institut National de Recherche en Automatique et Informatique (INRIA) and joined BRICS in Aalborg in October 2000. His broad research interests lie in abstract interpretation, program analysis and program verification and more specifical ly the quantitative verification of probabilistic systems by means of abstraction and automatic refinement.

Margarita Korovina

Margarita Korovina received her PhD from Sobolev Institute of Mathematics, Novosibirsk, Russia in December 1996. Her thesis, *Generalised computability on the real numbers* presented a theory of computation of real-valued functions, functionals and operators of finite types. Her current research interests include computable analysis, definability and computability theories, domain theory and hybrid systems. Margarita joined BRICS in September 2001.

Anna Östlin

Anna Östlin received her PhD from Lund University, Sweden, earlier this year. She has mostly been working on problems in computational biology, especially evolutionary tree construction, which was the topic of her thesis *Constructing Evolutionary Trees – Algorithms and Complexity*. She has also worked on algorithms for network communication and computational geometry. Anna joined BRICS in August 2001.

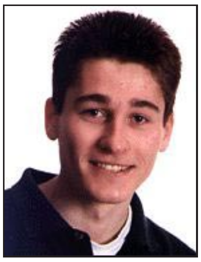
Anders P. Ravn

Anders P. Ravn holds a Doctor of Technology (dr. techn.) degree from the Technical University of Denmark (DTU), and a MSc (cand. scient.) degree in computer science and mathematics from the University of Copenhagen (KU). He has been employed in private industry, and has also been lecturer at KU and later DTU. He has been visiting researcher for longer periods at IBM Research, at Oxford University and at Kiel University. Anders takes a keen interest in methods for development of *embedded systems*, in particular the engineering of software for such systems using formal methods. Currently he is investigating application of UML to the design and implementation of a complex system. Anders has also worked on *Duration Calculus* - a class of interval logics, that has been tested successfully on a variety of case studies. Hard real-time systems are often found in complex control systems. Thus, Anders takes an interest in *Hybrid Systems*, which combine non-trivial discrete changes with continuous evolutions.

BRICS is also happy to welcome the following newly admitted PhD students and Marie Curie fellows.

Pablo Javier Arrighi

Pablo Javier Arrighi graduated in July 2000 from Imperial College, London, and is PhD student at the Computer Laboratory in Cambridge. He is visiting BRICS under the Marie Curie EU scheme in order to benefit from the local expertise in Quantum Information, mainly in the name of Dr. Louis Salvail. His current focus is on quantum secure computation with a wider interest on models for distributed quantum computation.



Alex Rune Berg

Alex R. Berg was accepted as PhD student in January 2000 after 4 years of study at the Department of Computer Science at the University of Aarhus. His main areas of interest are graph theory and graph algorithms, with a special interests in connectivity of graphs. He is under the supervision of Erik M. Schmidt and has Tibor Jordán as co-supervisor. Other areas of interest include cryptology and complexity theory.

Marco Carbone

Marco Carbone got his MSc (Laurea) in Computer Science in June 2001 from University of Pisa, Italy. His thesis, written under the supervision of Pierpaolo Degano, concerned Static Analysis on the Ambient Calculus. Since August 2001 he is a Phd student at BRICS. His research interests are within concurrency theory. His preliminary supervisor is Mogens Nielsen.

Aske Simon Christensen

Aske Simon Christensen is a newly accepted PhD student after four years of study in Aarhus. His main areas of interest are programming languages, program analysis and compiler technology. Some others are virtual machines, runtime systems and low level optimisations. His supervisor is Michael I. Schwartzbach.

Kasper Dupont

Kasper Dupont is a newly accepted PhD student after 4 years of study in Aarhus. Some of his main interests are cryptology and algorithmic. Under supervision of Ivan B. Damgård he will amongst other things be searching for cryptographic algorithms that can be proved secure under other and perhaps weaker assumptions than are usually used.

Serge Fehr

Serge Fehr graduated in Mathematics in March 1998 from the Swiss Federal Institute of Technology (ETH) Zürich, specialising in computer algebra and cryptography. The following three years he spent as a PhD student in the cryptography research group of Ueli Maurer at the ETH Zurich, and he is going to finish his PhD study here under the supervision of Ivan B. Damgård. His primary interest is security in distributed settings. Other areas of interest include interactive proofs and electronic voting.

Jens Groth

Jens Groth graduated with an MSc in March 2001 from the Department of Mathematical Sciences at the University of Aarhus. He is now a PhD student in cryptography with main interest in electronic voting. As part of his PhD studies he works part time at CRYPTOMATHIC. His advisor is Ivan B. Damgård.

Jesús Fernando Almansa Guera

Jesús F. Almansa graduated as Mathematician in June 1996 and then got a MSc Degree in Computer Science in August 1998, in Colombia. His interests by then focused on type theory applied to different calculi (lambda, concurrent, object and constraint calculi). Since August 2001, Jesús became a PhD student at BRICS, where his main research area is Cryptology, initially on Multiparty Authentication. His preliminary supervisor is Peter Bro Miltersen.

Jørgen Iversen

Jørgen Iversen became a PhD student at BRICS in February 2001 after four and a half years of study in Aarhus. His main area of interest is programming language design and semantics, with a special interest in semantics based compiler generation and action semantics. One of his goals is to generate compilers comparable to hand-coded compilers, by transforming and compiling actions into efficient code. His supervisor is Peter D. Mosses.

Bolette Ammitzbøll Madsen
Bolette Ammitzbøll Madsen was accepted as a PhD student at BRICS under the supervision of Sven Skyum in February 2001. Her primary fields of interest are algorithms and complexity theory. She is currently working on exact algorithms with low worst-case complexity for certain NP-complete problems together with Jesper M. Nielsen and Bjarke Skjernaa.

Kirill Morozov
Kirill Morozov graduated in June 1998 from the State University of Telecommunications, Saint-Petersburg, Russia. During his under-graduate studies he worked as a research assistant in cryptology. He became a PhD student at BRICS in August 2001. His main area of interest is cryptology, with a special interest in unconditionally secure cryptographic primitives based on noisy channels, under the supervision of Ivan B. Damgård.

Bartek Klin
Bartek Klin graduated in October 2000 from the faculty of Mathematics, Mechanics and Computer Science of Warsaw University, with a thesis on the specification language CASL written under the supervision of Andrzej Tarlecki. His areas of interest include semantics of programming languages and algebraic specifications. His preliminary supervisor is Peter D. Mosses.

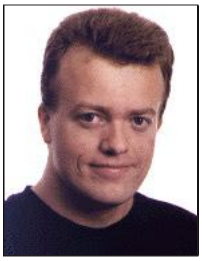
Sunil Kothari
Sunil Kothari is a newly accepted PhD student under the supervision of Michael Schwartzbach. He did his masters in Artificial Intelligence from the University of Edinburgh where he worked on applying genetic programming paradigm to planning problems. He is mainly interested in compilers, programming languages and web technology.

Jesper Makholm Nielsen
Jesper Makholm Nielsen became a PhD student at BRICS in August 2000 after five years of study in Aarhus. His primary fields of interest are algorithms, complexity theory and discrete math and his supervisor is Peter Bro Miltersen. Right now he is working together with Bolette A. Madsen and Bjarke Skjernaa on exact algorithms for NP-complete problems.

Bjarke Skjernaa
Bjarke Skjernaa became a PhD student at BRICS in August 2000 after four years of study at the University of Aarhus. His primary fields of interest are algorithms, complexity theory and discrete mathematics. Right now he is working together with Bolette A. Madsen and Jesper M. Nielsen on exact algorithms for NP-complete problems. His supervisor is Sven Skyum.

Pawel Sobocinski
Pawel started in the summer of 2000. I graduated with an honours degree in Pure Mathematics from the University of Sydney, NSW, Australia. His honours thesis was about a generalisation of classical Galois Theory to pure Category Theory. He is mainly interested in categorical models for concurrency and process algebra. His supervisor is Mogens Nielsen.

René Thomsen
René Thomsen became a PhD student at EVALife/BRICS in February 2001 after five years of study in Aarhus. His supervisors are Thiemo Krink and Brian Mayoh. His primary fields of interest are applications of evolutionary computation in Bioinformatics with the main focus on evolutionary algorithms. Initially he will work on problems such as multiple sequence alignment, side-chain prediction, and secondary structure prediction. These are all components that will be useful in the latter part of the PhD study where he will concentrate his work on prediction of the tertiary structure of proteins also referred to as the protein folding problem.



Rasmus Kjær Ursem
Rasmus K. Ursem initiated his PhD studies in August 1999 in the EVALife project located at Dept. of Computer Science, University of Aarhus. The main focus in his PhD project is to develop novel evolutionary algorithms for optimisation and to apply them to industrial problems in control engineering, in particular system identification and control of nonlinear dynamic systems. Rasmus' supervisors are Thiemo Krink and Brian H. Mayoh.

Dissertation Abstracts

Using Theory to Make Better Tools

Niels Damgaard
In this dissertation four tools based on theory will be presented. The tools fall in four quite different areas.

The first tool is an extension of parser generators, enabling them to handle side constraints along with grammars. The side constraints are specified in a first-order logic on parse trees. The constraints are compiled into equivalent tree automata, which result in good runtime performance of the generated checkers. The tool is based on language theory, automata theory and logic.

The second tool is a static checker of assembly code for interrupt-based micro-controllers. The tool gives lower and upper bounds on the stack height, checks an implicit type-system and calculates the worst-case delay from an interrupt fires until it is handled. The tool is based on the theory of data-flow analysis.

The third 'tool' is a security extension of a system (`<bigwig>`) for making interactive web-services. The extension provides confidentiality, integrity and authenticity in the communication between the client and the server. Furthermore, some primitives are introduced to ease the programming of cryptological protocols, like digital voting and e-cash, in the `<bigwig>` language. The extension uses the theory of cryptography.

The fourth tool is a black-box system for finding neural networks good at solving a given classification problem. Back-propagation is used to train the neural network, but the rest of the parameters are found by a distributed genetic-algorithm. The tool uses the theory of neural networks and genetic algorithms. See [DS-01-8].

On Static and Dynamic Control-Flow Information in Program Analysis and Transformation

Daniel Damian
This thesis addresses aspects of static and dynamic control-flow information in programming languages, by investigating its interaction with program transformation and analysis.

Control-flow information indicates for each point in a program the possible program points to be executed next. Control-flow information in a program may be static, as when the syntax of the program directly determines which parts of the program may be executed next. Control-flow information may be dynamic, as when run-time values and inputs of the program are required to determine which parts of the program may be executed next. A control-flow analysis approximates the dynamic control-flow information with conservative static control-flow information.

We explore the impact of a continuation-passing-style (CPS) transformation on the result of a constraint-based control-flow analysis over Moggi's computational metalanguage. A CPS transformation makes control-flow explicitly available to the program by abstracting the remainder of the computation into a continuation. Moggi's computational metalanguage supports reasoning about higher-order programs in the presence of computational effects. We show that a non-duplicating CPS transformation does not alter the result of a mono-variant constraint-based control-flow analysis.

Building on control-flow analysis, we show that traditional constraint-based binding-time analysis and traditional partial evaluation benefit from the effects of a CPS transformation, while the same CPS transformation does not affect continuation-based partial evaluation and its corresponding binding-time analysis. As an intermediate result, we show that reducing a program in the computational metalanguage to monadic normal form also improves binding times for traditional partial evaluation while it does not affect continuation-based partial evaluation.

In addition, we show that linear β-reductions have no effect on control-flow analysis. As a corollary, we solve a problem left open in Palsberg and Wand's CPS transformation of flow information. Furthermore, using Danvy and Nielsen's first-order, one-pass CPS transformation, we present a simpler CPS transformation of flow information with a simpler correctness proof.

We continue by exploring Shivers's time-stamps-based technique for approximating program analyses over programs with dynamic control flow. We formalise a time-stamps-based algorithm for approximating the least fixed point of a generic program analysis over higher-order programs, and we prove its correctness.

We conclude by investigating the translation of first-order structured programs into first-order unstructured programs. We present a one-pass translation that integrates static control-flow information and that produces programs containing no chains of jumps, no unused labels, and no redundant labels. See [DS-01-5].

Multi-party Computations — Information-Theoretically Secure Against an Adaptive Adversary

Stefan Dziembowski

In this thesis we study a problem of doing Verifiable Secret Sharing (VSS) and Multi-party Computations in a model where private channels between the players and a broadcast channel is available. The adversary is active, adaptive and has an unbounded computing power. The thesis is based on two papers [1,2].

In [1] we assume that the adversary can corrupt any set from a given *adversary* structure. In this setting we study a problem of doing efficient VSS and MPC given an access to a secret sharing scheme (SS). For all adversary structures where VSS is possible at all, we show that, up to a polynomial time black-box reduction, the complexity of adaptively secure VSS is the same as that of ordinary secret sharing (SS), where security is only required against a passive, static adversary. Previously, such a connection was only known for linear secret sharing and VSS schemes.

We then show an impossibility result indicating that a similar equivalence does not hold for Multi-party Computation (MPC): we show that even if protocols are given black-box access for free to an idealised secret sharing scheme secure for the access structure in question, it is not possible to handle all relevant access structures efficiently, not even if the adversary is passive and static. In other words, general MPC can only be black-box reduced efficiently to secret sharing if extra properties of the secret sharing scheme used (such as linearity) are assumed.

The protocols of [2] assume that we work against a *threshold* adversary structure. We propose new VSS and MPC protocols that are substantially more efficient than the ones previously known.

Another contribution of [2] is an attack against a Weak Secret Sharing Protocol (WSS) of [3]. The attack exploits the fact that the adversary is adaptive. We present this attack here and discuss other problems caused by the adaptiveness.

All protocols in the thesis are formally specified and the proofs of their security are given.

[1] Ronald Cramer, Ivan Damgård, Stefan Dziembowski, Martin Hirt, and Tal Rabin. Efficient multi-party computations with dishonest minority. In Advances in Cryptology — Eurocrypt '99, volume 1592 of Lecture Notes in Computer Science, pages 311–326, 1999.

[2] Ronald Cramer, Ivan Damgård, and Stefan Dziembowski. On the complexity of verifiable secret sharing and multi-party computation. In Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, pages 25–334, May 2000.

[3] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multi-party protocols with honest majority (extended abstract). In Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing, pages 73–85, Seattle, Washington, 15–17 May 1989.

See [DS-01-1].

Topics in Semantics-based Program Manipulation

Bernd Grobauer

Programming is at least as much about manipulating existing code as it is about writing new code. Existing code is *modified*, for example to make inefficient code run faster, or to accommodate for new features when reusing code; existing code is *analysed*, for example to verify certain program properties, or to use the analysis information for code modifications. Semantics-based program manipulation addresses methods for program modifications and program analyses that are formally defined and therefore can be verified with respect to the programming-language semantics. This dissertation comprises four articles in the field of semantics-based techniques for program manipulation: three articles are about *partial evaluation*, a method for program specialisation; the fourth article treats an approach to *automatic cost analysis*.

Partial evaluation optimises programs by specialising them with respect to parts of their input that are already known: Computations that depend only on known input are carried out during partial evaluation, whereas computations that depend on unknown input give rise to residual code. For example, partially evaluating an interpreter with respect to a program written in the interpreted language yields code that carries out the computations described by that program; partial evaluation is used to remove interpretive overhead. In effect, the partial evaluator serves as a compiler from the interpreted language into the implementation language of the interpreter. Compilation by partial evaluation is known as the first *Futamura projection*. The second and third Futamura projection describe the use of partial evaluation for compiler generation and compiler-generator generation, respectively; both require the partial evaluator that is employed to be *self applicable*.

The first article in this dissertation describes how the second Futamura projection can be achieved for type-directed partial evaluation (TDPE), a relatively recent approach to partial evaluation: We derive an ML implementation of the second Futamura projection for TDPE. Due to the differences between 'traditional', syntax-directed partial evaluation and TDPE, this derivation involves several conceptual and technical steps. These include a suitable formulation of the second Futamura projection and techniques for making TDPE amenable to self-application.

In the second article, compilation by partial evaluation plays a central role for giving a unified approach to goal-directed evaluation, a programming-language paradigm that is built on the notions of backtracking and of generating successive results. Formulating the semantics of a small goal-directed language as a *monadic* semantics—a generic approach to structuring denotational semantics—allows us to relate various possible semantics to each other both conceptually and formally. We thus are able to explain goal-directed evaluation using an intuitive list-based semantics, while using a continuation semantics for semantics-based compilation through partial evaluation. The resulting code is comparable to that produced by an optimised compiler described in the literature.

The third article revisits one of the success stories of partial evaluation, the generation of efficient string matchers from intuitive but inefficient implementations. The basic idea is that specialising a naive string matcher with respect to a pattern string should result in a matcher that searches a text for this pattern with running time independent of the pattern and linear in the length of the text. In order to succeed with basic partial-evaluation techniques, the naive matcher has to be modified in a non-trivial way, carrying out so-called *binding-time improvements*. We present a step-by-step derivation of a binding-time improved matcher consisting of one problem-dependent step followed by standard binding-time improvements. We also consider several variants of matchers that specialise well, amongst them the first such matcher presented in the literature, and we demonstrate how variants can be derived from each other systematically.

The fourth article is concerned with program analysis rather than program transformation. A challenging goal for program analysis is to extract information about time or space complexity from a program. In complexity analysis, one often establishes *cost recurrences* as an intermediate step, and this step requires an abstraction from data to data size. We use information contained in dependent types to achieve such an abstraction: Dependent ML (DML), a conservative extension of ML, provides dependent types that can be used to associate data with size information, thus describing a possible abstraction. We automatically extract cost recurrences from first-order DML programs, guiding the abstraction from data to data size with information contained in DML type derivations. See [DS-01-6].

Logics and Automata for Verification: Expressiveness and Decidability Issues

Jesper G. Henriksen

This dissertation investigates and extends the mathematical foundations of logics and automata for the interleaving and synchronous non-interleaving view of system computations with an emphasis on decision procedures and relative expressive powers, and introduces extensions of these foundations to the emerging domain of non-interleaving asynchronous computations. System computations are described as occurrences of system actions, and tractable collections of such computations can be naturally represented by finite automata upon which one can do formal analysis. Specifications of system properties are usually described in formal logics, and the question whether the system at hand satisfies its specification is then solved by means of automata-theoretic constructions.

Our focus here is on the linear time behaviour of systems, where executions are modelled as sequence-like objects, neither reflecting nondeterminism nor branching choices. We consider a variety of linear time paradigms, such as the classical interleaving view, the synchronous non-interleaving view, and conclude by considering an emerging paradigm of asynchronous non-interleaving computation. Our contributions are mainly theoretical though there is one piece of practical implementation work involving a verification tool. The theoretical work is concerned with a range of logics and automata and the results involve the various associated decision procedures motivated by verification problems, as well as the relative expressive powers of many of the logics that we consider.

Our research contributions, as presented in this dissertation, are as follows. We describe the practical implementation of the verification tool Mona. This tool is basically driven by an engine which translates formulas of monadic second-order logic for finite strings to deterministic finite automata. This translation is known to have a daunting complexity-theoretic lower bound, but surprisingly enough, it turns out to be possible to implement a translation algorithm which often works efficiently in practice; one of the major reasons being that the internal representation of the constituent automata can be maintained symbolically in terms of binary decision diagrams. In effect, our implementation can be used to verify the so-called safety properties because collections of finite strings suffice to capture such properties.

For reactive systems, one must resort to infinite computations to capture the so-called liveness properties. In this setting, the predominant specification mechanism is Pnueli's LTL which turns out to be computationally tractable and, moreover, equal in expressive power to the first-order fragment of monadic second-order logic. We define an extension of LTL based on the regular programs of PDL to obtain a temporal logic, DLTL, which remains computationally feasible and is yet expressively equivalent to the full monadic second-order logic.

An important class of distributed systems consists of networks of sequential agents that synchronise by performing common actions together. We exhibit a distributed version of DLTL and show that it captures exactly all linear time properties of such systems, while the verification problem, once again, remains tractable.

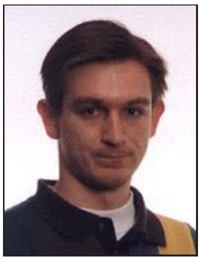
These systems constitute a subclass of a more general class of systems with a static notion of independence. For such systems the set of computations constitute interleavings of occurrences of causally independent actions. These can be grouped in a natural manner into equivalence classes corresponding to the same partially-ordered behaviour. The equivalence classes of computations of such systems can be canonically represented by restricted labelled partial orders known as (Mazurkiewicz) traces. It has been noted that many properties expressed as LTL-formulas have the “all-or-none” flavour, i.e. either all computations of an equivalence class satisfy the formula or none do. For such properties (e.g. “reaching a deadlocked state”) it is possible to take advantage of the non-interleaving nature of computations and apply the so-called partial-order methods for verification to substantially reduce the computational resources needed for the verification task. This leads to the study of linear time temporal logics interpreted directly over traces, as specifications in such logics are guaranteed to have the “all-or-none” property. We provide an elaborate survey of the various distributed linear time temporal logics interpreted over traces.

One such logic is TLC, through which one can directly formulate causality properties of concurrent systems. We strengthen TLC to obtain a natural extended logic TLC* and show that the extension adds non-trivially to the expressive power. In fact, very little is known about the relative expressive power of the various logics for traces. The game-theoretic proof technique that we introduce may lead to new separation results concerning such logics.

In application domains such as telecommunication software, the synchronous communication mechanism that traces are based on is not appropriate. Rather, one would like message-passing to be the basic underlying communication mechanism. Message Sequence Charts (MSCs) are ideally suited for the description of such scenarios, where system executions constitute partially ordered exchanges of messages. This raises the question of what constitutes reasonable collections of MSCs upon which one can hope to do formal analysis. We propose a notion of regularity for collections of MSCs and tie it up with a class of finite-state devices characterising such regular collections. Furthermore, we identify a monadic second-order logic which also defines exactly these regular collections.

The standard method for the description of multiple scenarios in this setting has been to employ MSC-labelled finite graphs, which might or might not describe collections of MSCs regular in our sense. One common feature is that these collections are finitely generated, and we conclude by exhibiting a subclass of such graphs which describes precisely the collections of MSCs that are both finitely generated and regular. See [DS-00-6]. ☷

Analysing Real-Time Systems: Theory and Tools



Thomas S. Hune

The main topic of this dissertation is the development and use of methods for formal reasoning about the correctness of real-time systems, in particular methods and tools to handle new classes of problems. In real-time systems the correctness of the system does not only depend on the order in which actions take place, but also the timing of the actions. The formal reasoning presented here is based on (extensions of) the model of timed automata and tools supporting this model, mainly UPPAAL. Real-time systems are often part of safety critical systems e.g. control systems for planes, trains, or factories, though also everyday electronics as audio / video equipment and (mobile) phones are considered real-time systems. Often these systems are concurrent systems with a number of components interacting, and reasoning about such systems is notoriously difficult. However, since most of the systems are either safety critical or errors in the systems are costly, ensuring correctness is very important, and hence formal reasoning can play a role in increasing the reliability of real-time systems.

We present two classes of cost extended timed automata, where a cost is associated to an execution of the automaton. We show that calculating the minimum cost of reaching a location in the automaton, the minimum-cost reachability problem, is decidable for both classes. Since a number of optimisation problems, e.g. scheduling problems in a natural way, can be modelled using cost extended timed automata, we can now solve these problems using extensions of timed model checkers. The state-space of the simpler class, *uniformly priced timed automata* (UPTA), which is a subclass of *linearly priced timed automata* (LPTA), can effectively be represented using a slightly modified version of the well known difference bounded matrix (DBM) data-structure for representing zones, used in most timed model checkers. Using an extension of the region construction, the minimum-cost reachability problem can also be solved for LPTAs. However, the standard way of using zones for representing the state-space cannot be used for LPTAs, since there is no way of capturing the cost of states. Based on the notion of facets, zones can be split into smaller zones which can be represented by extended DBMs in an effective way. Minimum-cost reachability for both UPTAs and LPTAs have been implemented in extensions of UPPAAL, and successfully tested on a number of case studies. In particular, part of the Sidmar steel production plant, which is a case study of the Esprit VHS project, has been studied. Schedulability, without considering cost and optimality, has also been addressed using standard timed automata and UPPAAL. In order to solve the schedulability problem in UPPAAL it proved crucial to add a number of *guides* to the model, in order to limit the search space. In the cost extended versions of UPPAAL, guiding in terms of changing the order in which states are searched has also been used, and shown to be effective both for finding solutions to optimisation problems and in ordinary timed model checking.

The second extension of timed automata is parametric timed automata, where parameters can be used in expressions for guards and invariants. We consider the problem of synthesising values for the parameters ensuring satisfiability of reachability properties. Since there are in most cases infinitely many values ensuring that a property is satisfied, the result is presented in terms of constraints for the parameters. We present a semi-decision procedure synthesising the constraints. The problem of synthesising constraints for the parameters has been show to be undecidable. To represent the state-space we extend the DBM data-structure to parametric DBMs, capable of representing zones were the bounds are given by expressions including parameters. The semi-decision procedure is implemented in UPPAAL and constraints ensuring correctness of a number of industrial protocols is synthesised.

Since (timed) reachability checking requires large amounts of resources in terms of memory and CPU time, we have studied the possibility of distributing the reachability checking to a network of computers. We have implemented a distributed version of UPPAAL and tested it on a number of the largest known case studies for UPPAAL. Not only did we achieve effective usage of all the connected computers (close to linear speedup in the number of computers) we also discovered that the breadth-first search order, which previously has been considered to be the best known, is not optimal.

We apply the general categorical framework of *open maps* to timed automata by presenting a category where the elements are timed automata, and a subcategory suitably for representing observations, timed words. Following the framework, maps in the category can be seen as simulations, and two timed automata $\mathcal{A}$ and $\mathcal{B}$ are timed bisimilar if and only if there exists a timed automaton $\mathcal{C}$ and open maps $\mathcal{C} \to \mathcal{A}$ and $\mathcal{C} \to \mathcal{B}$. We show that this notion of timed bisimulation coincides with the know notion of timed bisimulation, and using the region construction show that the bisimulation is decidable.

Building timed automata models of systems can be an error prone and time consuming task. We address this problem by presenting a translation from a low level programming language used in the programmable LEGO® RCX™ brick to timed automata. Programs for the RCX™ brick can consist of several tasks running concurrently. Therefore an important part of the model of the program is a model of the scheduler. The translation has been implemented and tested on a control program for a car.

Finally, we consider a kind of partial program synthesis for un-timed systems. Given a safety specification written in monadic second order logic, we use the Mona tool to derive an automaton accepting the language of the specification. The automaton is used to restrict the executions of a handwritten control program, ensuring that the safety requirements are met. To demonstrate the approach we consider a control program for a crane, written for the RCX™ brick. We also discuss more generally what should happen when there is a conflict between the actions of the control program and the specification. See [DS-01-3].

Robust and Flexible Scheduling with Evolutionary Computation

Mikkel T. Jensen

Over the last ten years, there have been numerous applications of evolutionary algorithms to a variety of scheduling problems. Like most other research on heuristic scheduling, the primary aim of the research has been on deterministic formulations of the problems. This is in contrast to real world scheduling problems which are usually not deterministic. Usually at the time the schedule is made some information about the problem and processing environment is available, but this information is uncertain and likely to change during schedule execution. Changes frequently encountered in scheduling environments include machine breakdowns, uncertain processing times, workers getting sick, materials being delayed and the appearance of new jobs. These possible environmental changes mean that a schedule which was optimal for the information available at the time of scheduling can end up being highly suboptimal when it is implemented and subjected to the uncertainty of the real world. For this reason it is very important to find methods capable of creating *robust* schedules (schedules expected to perform well after a minimal amount of modification when the environment changes) or *flexible* schedules (schedules expected to perform well after some degree of modification when the environment changes).

This thesis presents two fundamentally different approaches for scheduling job shops facing machine breakdowns. The first method is called *neighbourhood based robustness* and is based on an idea of minimising the cost of a neighbourhood of schedules. The scheduling algorithm attempts to find a small set of schedules with an acceptable level of performance. The approach is demonstrated to significantly improve the robustness and flexibility of the schedules while at the same time producing schedules with a low implementation cost if no breakdown occurs. The method is compared to a state of the art method for stochastic scheduling and concluded to have the same level of performance, but a wider area of applicability. The current implementation of the method is based on an evolutionary algorithm, but since the real contribution of the method is a new performance measure, other implementations could be based on tabu search, simulated annealing or other powerful "blind" optimisation heuristics.

The other method for stochastic scheduling uses the idea of coevolution to create schedules with a guaranteed worst case performance for a known set of scenarios. The method is demonstrated to improve worst case performance of the schedules when compared to ordinary scheduling; it substantially reduces program running times when compared to a more standard approach explicitly considering all scenarios. Schedules based on worst case performance measures often have suboptimal performance if no disruption happens, so the co-evolutionary algorithm is combined with a multi-objective algorithm which optimises worst case performance as well as performance without disruptions.

The co-evolutionary worst case algorithm is also combined with another algorithm to create schedules with a guaranteed level of *worst deviation performance*. In worst deviation performance the objective is to minimise the highest possible performance difference from the schedule optimal for the scenario that actually takes place. Minimising this kind of performance measure involves solving a large number of related scheduling problems in one run, so a new evolutionary algorithm for this kind of problem is suggested.

Other contributions of the thesis include a new co-evolutionary algorithm for minimax problems. The new algorithm is capable of solving problems with an asymmetric property that causes previously published algorithms to fail. Also, a new algorithm to solve the economic lot and delivery scheduling problem is presented. The new algorithm is guaranteed to solve the problem to optimality in polynomial time, something previously published algorithms have not been able to do. See [DS-01-3].

Reliable Real-Time Applications

Peter Krogsgaard Jensen

The task of verifying timing properties of critical computing systems during development is important, due to the serious or fatal consequences if such systems fail. The importance is growing because more and more software is involved in controlling the everyday life of humans. As we need to develop and verify more and more complex and larger systems, the software engineering process must be improved to maintain the quality of the end product. One way to improve is by making it more automatic so that trivial and error prone manual tasks can be handled by tools.

The development of software for industrial use will involve a testing phase, with the purpose of “verifying” the requirements of the product. Traditionally tests have targeted functional requirements, but the thesis in this work is that tests can to a higher degree than today be the foundation of analysis and verification of temporal properties. To support our thesis, we develop the Test Observation Paradigm, with tools to analyse temporal information from (instrumented) systems under test. An industrial sized control application is used as a case study, and all tools and techniques that we have developed are used in the case study. The case study works as a proof-of-concept, displaying the pros and cons of the tools and techniques.

During a test all paths of the implemented program text should be executed (in theory), but for a real-time application with a fast periodic task a particular path is executed many times. If the test is thorough this path is executed with different parameters, leading to different execution times due to conditional and loop statements. We suggest a Reliable Worst Case estimate, called RWC, which is an upper boundary of the statistical model of the variation of such execution times. This RWC is used for two things: 1) to visualise the structure of real-time applications, and 2) to evaluate deadlines in schedulability analysis. The RWC is a new approach to worst case execution time estimation for real-time programs.

One problem in many analytical analyses of a real-time program is that it can only handle a process model consisting of threads that serve one purpose. Much of the complexity in the work presented here originates from a larger process model where tasks are more complex. It is our aim to be able to analyse systems that use all these facilities. It is not desirable to use a smaller process model because sound software design principles for software engineering cannot be followed using simple models. Limitations cannot be completely avoided because we want a temporal predictable application.

As part of the work with the case study we have developed middle-ware and extensions for a real-time kernel so that we can observe the structure and timing of systems under test. It turns out that only 1.5 % of the CPU power is consumed by the extensions, and this convinces us that it is feasible for an industrial system.

Also to aid observation a trace language is suggested that can describe the structure of a large range of real-time applications. This language is a formal description of real-time applications, and we demonstrate its use as a part of the case study. The Observed Model, is the name we use for a model of an application described by a set of strings from the trace language, and all our analysis of the temporal behaviour is based on this model.

In the thesis, a task inspection technique is suggested. It is a postmortem traceability analysis which creates an end-to-end connection from input event to output event. With this analysis the structure of the real-time application can be visualised, based on worst case estimate of execution time of the program text.

Detailed timing information annotated to the trace language model makes it possible to do schedulability analysis on the involved tasks. The schedulability analysis is based on RWC values, which substitutes the WCET in the analysis. With an additional model that describes the real-time kernel, properties of the complete real-time system becomes verifiable. Part of the case study is a presentation of a technique where model checking is applied to an Observed Model including a model of the kernel. With this model checking, it is possible to verify deadlines and maximum jitter of events, by applying the real-time model checker Uppaal.

Games for Verification: Algorithmic Issues

Marcin Jurdziński

This dissertation deals with a number of algorithmic problems motivated by computer aided formal verification of finite state systems. The goal of formal verification is to enhance the design and development of complex systems by providing methods and tools for specifying and verifying correctness of designs. The success of formal methods in practice depends heavily on the degree of automation of development and verification process. This motivates development of efficient algorithms for problems underlying many verification tasks.

Two paradigmatic algorithmic problems motivated by formal verification that are in the focus of this thesis are model checking and bisimilarity checking. In the thesis game theoretic formulations of the problems are used to abstract away from syntactic and semantic peculiarities of formal models and specification formalisms studied. This facilitates a detailed algorithmic analysis, leading to two novel model checking algorithms with better theoretical or practical performance, and to an undecidability result for a notion of bisimilarity.

The original technical contributions of this thesis are collected in three research articles whose revised and extended versions are included.

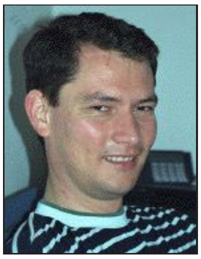
In the first two papers the computational complexity of deciding the winner in parity games is studied. The problem of solving parity games is polynomial time equivalent to the modal mu-calculus model checking. The modal mu-calculus plays a central role in the study of logics for specification and verification of programs. The model checking problem is extensively studied in literature on computer aided verification. The question whether there is a polynomial time algorithm for the modal mu-calculus model checking is one of the most challenging and fascinating open questions in the area.

In the first paper a new algorithm is developed for solving parity games, and hence for the modal mu-calculus model checking. The design and analysis of the algorithm are based on a semantic notion of a progress measure. The worst-case running time of the resulting algorithm matches the best worst-case running time bounds known so far for the problem, achieved by the algorithms due to Browne at al., and Seidl. Our algorithm has better space complexity: it works in small polynomial space; the other two algorithms have exponential worst-case space complexity.

In the second paper a novel approach to model checking is pursued, based on a link between parity games and discounted payoff and stochastic games, established and advocated by Puri. A discrete strategy improvement algorithm is given for solving parity games, thereby proving a new procedure for the modal mu-calculus model checking. Known strategy improvement algorithms, as proposed for stochastic games by Hoffman and Karp, and for discounted payoff games and parity games by Puri, work with real numbers and require solving linear programming instances involving high precision arithmetic. The present algorithm for parity games avoids these difficulties by efficient manipulation of carefully designed discrete valuations. A fast implementation is given for a strategy improvement step. Another advantage of the approach is that it provides a better conceptual understanding of the underlying discrete structure and gives hope for easier analysis of strategy improvement algorithms for parity games. However, so far it is not known whether the algorithm works in polynomial time. The long standing problem whether parity games can be solved in polynomial time remains open.

In the study of concurrent systems it is common to model concurrency by non-determinism. There are, however, some models of computation in which concurrency is represented explicitly; elementary net systems and asynchronous transition systems are well-known examples. History preserving and hereditary history preserving bisimilarities are behavioural equivalence notions taking into account causal relationships between events of concurrent systems. Checking history preserving bisimilarity is known to be decidable for finite labelled elementary nets systems and asynchronous transition systems. Its hereditary version appears to be only a slight strengthening and it was conjectured to be decidable too. In the third paper it is proved that checking hereditary history preserving bisimilarity is undecidable for finite labelled asynchronous transition systems and elementary net systems. This solves a problem open for several years. The proof is done in two steps. First an intermediate problem of deciding the winner in domino bisimulation games for origin constrained tiling systems is introduced and its undecidability is shown by a reduction from the halting problem for 2-counter machines. Then undecidability of hereditary history preserving bisimilarity is shown by a reduction from the problem of domino bisimulation games. See [DS-00-7].

Reasoning about Objects using Process Calculus Techniques



Josva Kleist

This thesis investigates the applicability of techniques known from the world of process calculi to reason about properties of object-oriented programs.

The investigation is performed upon a small object-oriented language - The Sigma-calculus of Abadi and Cardelli. The investigation is twofold:

- We investigate translations of Sigma-calculi into process calculi, with the idea that one should be able to show properties of Sigma-calculus program by showing properties about their translation. We present translations of two Sigma-calculi into Pi-calculi. A translation of the untyped functional Sigma-calculus turns out to be insufficient. Based on our experiences, we present a translation of a typed imperative Sigma-calculus, which looks promising. We are able to provide simple proofs of the equivalence of different Sigma-calculus objects using this translation.
- We use a labelled transition system adapted to the Sigma-calculus to investigate the use of process calculi techniques directly on the Sigma-calculus. The results obtained are of a fairly theoretical nature. We investigate the connection between the operational and denotational semantics for a typed functional Sigma-calculus. The result is that Abadi and Cardelli's denotational model is sound but not complete with respect to the operational semantics. We also construct a modal logic for the typed functional Sigma-calculus, provide a translation of types to a sub-logic and prove the translation is sound and complete.

The amount work required to perform these investigations indicate, that although it is perfectly possible to use process calculus techniques on object oriented languages, such techniques will not come to widespread use, but only be limited to reasoning about critical parts of a language or program design. See www.cs.auc.dk/~kleist/Thesis/thesis.pdf

A Configurable Process for Design of Object-Oriented Software Architectures

Birgitte Lønvig

When we design large complex software systems, such as systems in the telecommunications world, and we follow one of the standard object-oriented methods or processes, we end up with a system that fulfils the requirements of functionality. However, it is difficult to ensure that other requirements, such as modifiability and reusability, are fulfilled. Furthermore the architecture is not explicitly described and is therefore difficult to comprehend.

This PhD dissertation defines a configurable process for design of object-oriented software architectures. The process can be regarded as an extension to standard object-oriented methods and processes. Software architecture is in focus by making its design explicit in a process and the process is configured for a specific domain.

Configuring ensures that the process contains only relevant process elements, contrary to a general process that must cover all possible combinations of problems and solutions in a number of different domains. The workflow of how to configure a process for a domain is although applicable for different domains.

The software architecture design process is based on a general conceptual framework consisting of domain characteristics, requirements, architectural levels, architectural patterns, and architectural structures.

Specification and Test of Real-Time Systems

Brian Nielsen

Distributed real-time computer based systems are very complex and intrinsically difficult to specify and implement correctly; in part this is caused by the overwhelming number of possible interactions between system components, but especially by a lack of adequate methods and tools to deal with this complexity. This thesis proposes new specification and testing techniques.

We propose a real-time specification language which facilitates modular specification and programming of reusable components. A specification consists of a set of concurrent un-timed components describing the functional behaviour of the system, and a set of constraint patterns which describes and enforces the timing and synchronisation constraints among components.

We propose new techniques for automated black box conformance testing of real-time systems against densely timed specifications. A test generator tool examines a specification of the desired system behaviour and generates the necessary test cases. A main problem is to construct a reasonably small test suite that can be executed within allotted resources, while having a high likelihood of detecting unknown errors. Our goal has been to treat the time dimension of this problem thoroughly.

Based on a determinisable class of timed automata, Event Recording Automata, we show how to systematically and automatically generate tests in accordance with Hennessy's classical testing theory lifted to include timed traces. We select test cases from a coarse grained state space partitioning of the specification, and cover each partition with at least one test case, possibly selecting extreme clock values. In a partition, the system behaviour remains the same independently of the actual clock values.

We employ the efficient symbolic constraint solving techniques originally developed for model checking of real-time systems to compute the reachable parts of these equivalence classes, to synthesise the timed tests, and to guarantee a coverage of the equivalence class partitioning. We have implemented our techniques in the RT-CAT test case generation tool.

Through a series of examples we demonstrate how Event Recording Automata can specify nontrivial and practically relevant timing behaviour. Despite being theoretically less expressive than timed automata, it has proven sufficiently expressive for our examples, but sometimes causing minor inconveniences. Applying RTCAT to generate tests from these specifications, including the Philips Audio Protocol, resulted in encouragingly small test suites.

We conclude that our approach is feasible and deserves further work, but also that it should be generalised and allow timing uncertainty and modelling of the environment. Some implementation improvements are also necessary. See www.cs.auc.dk/~bnielsen/Published/bnielsenthesis070801.ps.

Time-Space Trade-Offs

Jakob Pagter

In this dissertation we investigate the complexity of the time-space trade-off for sorting, element distinctness, and similar problems. We contribute new results, techniques, tools, and definitions pertaining to time-space trade-offs for the RAM (Random Access Machine), including new tight upper bounds on the time-space trade-off for Sorting in a variety of RAM models and a strong time-space trade-off lower bound for a decision problem in the RAM (in fact the branching program) model. We thus contribute to the general understanding of a number of fundamental combinatorial problems.

Time and space are the two most fundamental resources studied in the areas of algorithms and computational complexity theory and it is clearly important to study their they are related. In 1966 Cobham founded the area of time-space trade-offs by formally proving the first time-space trade-off—for the language of palindromes, i.e., a formulae that relate time and space in the form of either an upper or a lower bound—in this case $T \cdot S \in \Theta(n^2)$. Over the last thirty five years the area of time-space trade-offs has developed significantly and now constitutes its own branch of algorithms and computational complexity.

The area of time-space trade-offs deals with both upper and lower bounds and both are interesting, theoretically as well as practically. The viewpoint of this dissertation is theoretical, but we believe that some of our results can find applications in practice as well.

The last four years has witnessed a number of significant breakthroughs on proving lower bounds for decision problems, including the first non-trivial lower bound on the unit-cost RAM. We generalise work of Ajtai and prove the quantitatively best known lower bound in this model, namely a $\Omega(n \lg n / \lg \lg n)$ bound on time for any RAM deciding the so-called Hamming distance problem in space $n^{1-\epsilon}$.

For non-decision problems, i.e., problems with multiple outputs, much stronger results are known. For Sorting, a fundamental problem in computer science, Beame proved that for any RAM the time-space product satisfies $T \cdot S \in \Omega(n^2)$. We prove that this bound is in fact tight (for most time functions), i.e., we show a $T \cdot S \in O(n^2)$ upper bound for time between $O(n(\lg \lg n)^2)$ and (roughly) $O(n^2)$. If we allow randomisation in the Las Vegas fashion the lower bound still holds, and in this case we can meet it down to $O(n \lg \lg n)$.

From a practical perspective hierarchical memory layout models are the most interesting. Such models are called external memory models, in contrast to the internal memory models discussed above. Despite the fact that space might be of great relevance when solving practical problems on real computers, no theoretical model capturing space (and time simultaneously) has been defined. We introduce such a model and use it to prove so-called IO-space trade-offs for Sorting. Building on the above-mentioned techniques for time-space efficient internal memory Sorting, we develop the first IO-space efficient external memory Sorting algorithms. Further, we prove that these algorithms are optimal using a transformation result which allows us to deduce external memory lower bounds (on IO-space trade-offs) from already known internal memory lower bounds (such as that of Beame). See [DS-01-2].

A Study of Defunctionalisation and Continuation-Passing Style

Lasse R. Nielsen

We show that defunctionalisation and continuations unify a variety of independently developed concepts in programming languages.

Defunctionalisation was introduced by Reynolds to solve a specific need—representing higher-order functions in a first-order definitional interpreter—but it is a general method for transforming programs from higher-order programs to first-order. We formalise this method and give it a partial proof of correctness. We use defunctionalisation to connect independent first-order and higher-order specifications and proofs by, e.g., defunctionalising continuations.

The canonical specification of a continuation-passing style (CPS) transformation introduces many administrative redexes. To show a simulation theorem, Plotkin used a so-called colon-translation to bypass administrative reductions and relate the reductions common to the transformed and the un-transformed program. This allowed him to make a proof by structural induction on the source program. We extend the colon-translation and Plotkin's proof to show simulation theorems for a higher-order CPS transformation, written in a two-level language, and for a selective CPS transformation, keeping parts of the source program in direct style.

Using, e.g., Plotkin's CPS transformation and then performing all possible administrative reductions gives a so-called two-pass CPS transformation. A one-pass CPS transformation gives the same result but without traversing the output. Several different one-pass CPS transformations exist, and we describe, evaluate, and compare three of these: (1) A higher-order CPS transformation à la Danvy and Filinski, (2) a syntactic-theory based CPS transformation derived from Sabry and Felleisen's CPS transformation, and (3) a look-ahead based CPS transformation which extends the colon-translation to a full one-pass CPS transformation and is new. This look-ahead based CPS transformation is not compositional, but it allows reasoning by structural induction, which we use to prove the correctness of a direct-style transformation.

Syntactic theories are a framework for specifying operational semantics that uses evaluation contexts to define a reduction relation. The two primary operations needed to implement syntactic theories are decomposing an expression into an evaluation context and a redex, and plugging an expression into an evaluation context to generate an expression. If implemented separately, these operations each take time proportional to the size of the evaluation context, which can cause a quadratic overhead on the processing of programs. However, in actual use of syntactic theories, the two operations mostly occur consecutively. We define a single function, 'refocus', that combines these operations and that avoids unnecessary overhead, thereby deforesting the composition of plugging and decomposition. We prove the correctness of refocusing and we show how to automatically generate refocus functions from a syntactic theory with unique decomposition.

We connect defunctionalisation and Church encoding. Church encoding represents data-structures as functions and defunctionalisation represents functions as data-structures, and we show to which extent they are inverses of each other.

We use defunctionalisation to connect a higher-order CPS transformation and a syntactic-theory based CPS transformation. Defunctionalising the static continuations of the higher-order transformation gives exactly the evaluation contexts of the syntactic theory.

We combine CPS transformation and defunctionalisation to automatically generate data structures implementing evaluation contexts.Defunctionalising the continuations of a CPS-transformed reduction function yields the evaluation contexts and their plug function—corresponding to the traditional view of continuations as a representation of the context. Defunctionalising the continuations of a CPS-transformed evaluation function yields the evaluation contexts and the refocus functions—corresponding to the traditional view of continuations as a representation of the remainder of the computation. See [DS-01-7].

Higher-Order Program Generation

Morten Rhiger

This dissertation addresses the challenges of embedding programming languages, specialising generic programs to specific parameters, and generating specialised instances of programs directly as executable code. Our main tools are higher-order programming techniques and automatic program generation. It is our thesis that they synergise well in the development of customisable software.

Recent research on domain-specific languages propose to embed them into existing general-purpose languages. Typed higher-order languages have proven especially useful as meta languages because they provide a rich infrastructure of higher-order functions, types, and modules. Furthermore, it has been observed that embedded programs can be restricted to those having simple types using a technique called “phantom types”. We prove, using an idealised higher-order language, that such an embedding is sound (i.e., when all object-language terms that can be embedded into the meta language are simply typed) and that it is complete (i.e., when all simply typed object-language terms can be embedded into the meta language). The soundness proof is shown by induction over meta-language terms using a Kripke logical relation. The completeness proof is shown by induction over object-language terms. Furthermore, we address the use of Haskell and Standard ML as meta-languages.

Normalisation functions, as embodied in type-directed partial evaluation, map a simply-typed higher-order value into a representation of its long beta-eta normal form. However, being dynamically typed, the original Scheme implementation of type-directed partial evaluation does not restrict input values to be typed at all. Furthermore, neither the original Scheme implementation nor the original Haskell implementation guarantee that type-directed partial evaluation preserves types. We present three implementations of type-directed partial evaluation in Haskell culminating with a version that restricts the input to typed values and for which the proofs of type-preservation and normalisation are automated.

Partial evaluation provides a solution to the disproportion between general programs that can be executed in several contexts and their specialised counterparts that can be executed efficiently. However, stand-alone partial evaluation is usually too costly when a program must be specialised at run time. We introduce a collection of byte-code combinators for OCaml, a dialect of ML, that provides run-time code generation for OCaml programs. We apply these byte-code combinators in semantics-directed compilation for an imperative language and in run-time specialisation using type-directed partial evaluation.

Finally, we present an approach to compiling goal-directed programs, i.e., programs that backtrack and generate successive results: We first specify the semantics of a goal-directed language using a monadic semantics and a spectrum of monads. We then compile goal-directed programs by specialising their interpreter (i.e., by using the first Futamura projection), using type-directed partial evaluation. Through various back ends, including a run-time code generator, we generate ML code, C code, and OCaml byte code. See [DS-01-4].

Compression with Fast Random Access

Flemming Friche Rodler

The main topic of this dissertation is the development and use of methods for space efficient storage of data combined with fast retrieval. By fast retrieval we mean that a single data element can be randomly selected and decoded efficiently. In particular, the focus will be on compression of volumetric data with fast random access to individual voxels, decoded from the compressed data.

Volumetric data is finding widespread use in areas such as medical imaging, scientific visualisation, simulations and fluid dynamics. Because of the size of modern volumes, using such data sets in uncompressed form is often only possible on computers with extensive amounts of memory. By designing compression methods for volumetric data that support fast random access this problem might be overcome. Since lossless compression of three-dimensional data only provides relatively small compression ratios, lossy techniques must be used. When designing compression methods with fast random access there is a choice between designing general schemes that will work for a wide range of applications or to tailor the compression algorithm to a specific application at hand. This dissertation we be concerned with designing general methods and we present two methods for volumetric compression with fast random access. The methods are compared to other existing schemes showing them to be quite competitive.

The first compression method that we present is suited for online compression. That is, the data can be compressed as it is downloaded utilising only a small buffer. Inspired by video coding the volume is considered as a stack of slices. To remove redundancies between slices a simple “motion estimation” technique is used. Redundancies are further reduced by wavelet transforming each slice using a two-dimensional wavelet transform. Finally, the wavelet data is thresholded and the resulting sparse representation is quantised and encoded using a nested block indexing scheme, which allows for efficient retrieval of coefficients. While being only slightly slower than other existing schemes the method improves the compression ratio by about 50%.

As a tool for constructing fast and efficient compression methods that support fast random access we introduce the concept of lossy dictionaries and show how to use it to achieve significant improvements in compressing volumetric data. The dictionary data structure is widely used in computer science as a tool for space efficient storage of sparse sets. The main performance parameters of dictionaries are space, lookup time and update time. In this dissertation we present a new and efficient dictionary scheme, based on hashing, called CUCKOO HASHING. CUCKOO HASHING achieves worst case constant lookup time, expected amortised constant update time and space usage of three words per element stored in the dictionary, i.e., the space usage is similar to that of binary search trees. Running extensive experiments we show CUCKOO HASHING to be very competitive to the most commonly used dictionary methods. Since these methods have nontrivial worst case lookup time CUCKOO HASHING is useful in time critical systems where such a guarantee is mandatory.

Even though, time efficient dictionaries with a reasonable space usage exist, the space usage of these are still too large to be of use in lossy compression. However, if the dictionary is allowed to store a slightly different set than intended, new and interesting possibilities originate. Very space efficient and fast data structures can be constructed by allowing the dictionary to discard some elements of the set (false negatives) and also allowing it to include elements not in the set (false positives). The lossy dictionary we present in this dissertation is a variant of CUCKOO HASHING which results in fast lookup time. We show that our data structure is nearly optimal with respect to space usage. Experimentally, we find that the data structure has very good behaviour with respect to keeping the most important elements of the set which is partially explained by theoretical considerations. Besides working well in compression our lossy dictionary data structure might find use in applications such as web cache sharing and differential files.

In the second volumetric compression method with fast random access that we present in this dissertation, we look at a completely different and rather unexploited way of encoding wavelet coefficients. In wavelet based compression it is common to store the coefficients of largest magnitude, letting all other coefficients be zero. However, the new method presented allows a slightly different set of coefficients to be stored. The foundation of the method is a three-dimensional wavelet transform of the volume and the lossy dictionary data structure that we introduce. Comparison to other previously suggested schemes in the literature, including the two-dimensional scheme mentioned above, shows an improvement of up to 80% in compression ratio while the time for accessing a random voxel is considerably reduced compared to the first method. See [DS-01-9].

New in the BRICS Report Series

ISSN 0909-0878

51 Ulrich Kohlenbach. *On Weak Markov's Principle*. December 2001. 10 pp.

50 Jiří Srba. *Note on the Tableau Technique for Commutative Transition Systems*. December 2001. To appear in the proceedings of FOSSACS '02.

49 Olivier Danvy and Lasse R. Nielsen. *A First-Order One-Pass CPS Transformation*. 2001. Extended version of a paper to appear in the proceedings of FOSSACS '02.

48 Mogens Nielsen and Frank D. Valencia. *Temporal Concurrent Constraint Programming: Applications and Behavior*. December 2001. 36 pp.

47 Jesper Buus Nielsen. *Non-Committing Encryption is Too Easy in the Random Oracle Model*. December 2001. 20 pp.

46 Lars Kristiansen. *The Implicit Computational Complexity of Imperative Programming Languages*. November 2001. 46 pp.

45 Ivan B. Damgård and Gudmund Skovbjerg Frandsen. *An Extended Quadratic Frobenius Primality Test with Average Case Error Estimates*. November 2001. 43 pp.

44 M. Oliver Möller, Harald Rueß, and Maria Sorea. *Predicate Abstraction for Dense Real-Time Systems*. November 2001. 27 pp.

43 Ivan B. Damgård and Jesper Buus Nielsen. *From Known-Plaintext Security to Chosen-Plaintext Security*. November 2001. 18 pp.

42 Zoltán Ésik and Werner Kuich. *Rationally Additive Semirings*. November 2001. 11 pp.

41 Ivan B. Damgård and Jesper Buus Nielsen. *Perfect Hiding and Perfect Binding Universally Composable Commitment Schemes with Constant Expansion Factor*. October 2001. 43 pp.

40 Daniel Damian and Olivier Danvy. *CPS Transformation of Flow Information, Part II: Administrative Reductions*. October 2001. 9 pp.

39 Olivier Danvy and Mayer Goldberg. *There and Back Again*. October 2001. 14 pp.

38 Zoltán Ésik. *Free De Morgan Bisemigroups and Bisemilattices*. October 2001. 13 pp.

37 Ronald Cramer and Victor Shoup. *Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption*. October 2001. 34 pp.

36 Gerth Stølting Brodal, Rolf Fagerberg, and Riko Jacob. *Cache Oblivious Search Trees via Binary Trees of Small Height*. October 2001. 20 pp.

35 Mayer Goldberg. *A General Schema for Constructing One-Point Bases in the Lambda Calculus*. September 2001. 6 pp.

34 Flemming Friche Rodler and Rasmus Pagh. *Fast Random Access to Wavelet Compressed Volumetric Data Using Hashing*. August 2001. 31 pp.

33 Rasmus Pagh and Flemming Friche Rodler. *Lossy Dictionaries*. August 2001. 14 pp. Short version appears in Meyer auf der Heide, editor, *9th Annual European Symposiumon on Algorithms*, ESA '01 Proceedings, LNCS 2161, 2001, pages 300–311.

32 Rasmus Pagh and Flemming Friche Rodler. *Cuckoo Hashing*. August 2001. 21 pp. Short version appears in Meyer auf der Heide, editor, *9th Annual European Symposiumon on Algorithms*, ESA '01 Proceedings, LNCS 2161, 2001, pages 121–133.

31 Olivier Danvy and Lasse R. Nielsen. *Syntactic Theories in Practice*. July 2001. 37 pp. Extended version of an article to appear in the informal proceedings of the *Second International Workshop on Rule-Based Programming*, RULE 2001 (Firenze, Italy, September 4, 2001).

30 Lasse R. Nielsen. *A Selective CPS Transformation*. July 2001. 24 pp. To appear in Brookes and Mislove, editors, *27th Annual Conference on the Mathematical Foundations of Programming Semantics*, MFPS '01 Proceedings, ENTCS 45, 2000. A preliminary version appeared in Brookes and Mislove, editors, *Preliminary Proceedings of the 17th Annual Conference on Mathematical Foundations of Programming Semantics, MFPS '01*, (Aarhus, Denmark, May 24–27, *2001)*, BRICS Notes Series NS-01-2, 2001, pages 201–222.

29 Olivier Danvy, Bernd Grobauer, and Morten Rhiger. *A Unifying Approach to Goal-Directed Evaluation*. July 2001. 23 pp. To appear in *New Generation Computing*, 20(1), November 2001. A preliminary version appeared in Taha, editor, *2nd International Workshop on Semantics, Applications, and Implementation of Program Generation*, SAIG '01 Proceedings, LNCS 2196, 2001, pages 108–125.

28 Luca Aceto, Zoltán Ésik, and Anna Ingólfsdóttir. *A Fully Equational Proof of Parikh's Theorem*. June 2001. 28 pp.

27 Mario Jose Cáccamo and Glynn Winskel. *A Higher-Order Calculus for Categories*. June 2001. 24 pp. Appears in Boulton and Jackson, editors, *Theorem Proving in Higher Order Logics: 14th International Conference*, TPHOLs '01 Proceedings, LNCS 2152, 2001, pages 136–153.

26 Ulrik Frendrup and Jesper Nyholm Jensen. *A Complete Axiomatization of Simulation for Regular CCS Expressions*. June 2001. 18 pp.

25 Bernd Grobauer. *Cost Recurrences for DML Programs*. June 2001. 51 pp. Extended version of a paper to appear in Leroy, editor, *Proceedings of the 6th ACM SIGPLAN International Conference on Functional Programming*, 2001.

24 Zoltán Ésik and Zoltán L. Németh. *Automata on Series-Parallel Biposets*. June 2001. 15 pp. To appear in Kuich, editor, *5th International Conference*, Developments in Language Theory DLT '01 Proceedings, LNCS, 2001.

23 Olivier Danvy and Lasse R. Nielsen. *Defunctionalization at Work*. June 2001. 45 pp. Extended version of an article to appear in Søndergaard, editor, *3rd International Conference on Principles and Practice of Declarative Programming*, PPDP '01 Proceedings, 2001.

22 Zoltán Ésik. *The Equational Theory of Fixed Points with Applications to Generalized Language Theory*. June 2001. 21 pp. To appear in Kuich, editor, *5th International Conference*, Developments in Language Theory DLT '01 Proceedings, LNCS, 2001.

21 Luca Aceto, Zoltán Ésik, and Anna Ingólfsdóttir. *Equational Theories of Tropical Semirings*. June 2001. 52 pp. Extended abstracts of parts of this paper have appeared in Honsell and Miculan, editors, *Foundations of Software Science and Computation Structures*, FoSSaCS '01 Proceedings, LNCS 2030, 2000, pages 42–56 and in Gaubert and Loiseau, editors, *Workshop on Max-plus Algebras and their Applications to Discrete-event Systems, Theoretical Computer Science, and Optimization*, MAX-PLUS '01 Proceedings, IFAC (International Federation of Automatic Control) IFAC Publications, 2001.

20 Catuscia Palamidessi and Frank D. Valencia. *A Temporal Concurrent Constraint Programming Calculus*. June 2001. 31 pp.

19 Jiří Srba. *On the Power of Labels in Transition Systems*. June 2001. 23 pp. Full and extended version of Larsen and Nielsen, editors, *Concurrency Theory: 12th International Conference*, CONCUR '01 Proceedings, LNCS, 2001.

18 Katalin M. Hangos, Zsolt Tuza, and Anders Yeo. *Some Complexity Problems on Single Input Double Output Controllers*. May 2001. 27 pp.

17 Claus Brabrand, Anders Møller, Steffan Olesen, and Michael I. Schwartzbach. *Language-Based Caching of Dynamically Generated HTML*. May 2001. 18 pp.

16 Olivier Danvy, Morten Rhiger, and Kristoffer H. Rose. *Normalization by Evaluation with Typed Abstract Syntax*. May 2001. 9 pp. To appear in *Journal of Functional Programming*.

15 Luigi Santocanale. *A Calculus of Circular Proofs and its Categorical Semantics*. May 2001. 30 pp.

14 Ulrich Kohlenbach and Paulo B. Oliva. *Effective Bounds on Strong Unicity in L_1-Approximation*. May 2001. 38 pp.

13 Federico Crazzolara and Glynn Winskel. *Events in Security Protocols*. April 2001. 30 pp.

12 Torben Amtoft, Charles Consel, Olivier Danvy, and Karoline Malmkjær. *The Abstraction and Instantiation of String-Matching Programs*. April 2001. 37 pp.

11 Alexandre David and M. Oliver Möller. *From* HUPPAAL *to* UPPAAL*: A Translation from Hierarchical Timed Automata to Flat Timed Automata*. March 2001. 40 pp.

10 Daniel Fridlender and Mia Indrika. *Do we Need Dependent Types?* March 2001. 6 pp. Appears in *Journal of Functional Programming*, 10(4):409–415, 2000. Superseeds BRICS Report RS-98-38.

9 Claus Brabrand, Anders Møller, and Michael I. Schwartzbach. *Static Validation of Dynamically Generated HTML*. February 2001. 18 pp.

8 Ulrik Frendrup and Jesper Nyholm Jensen. *Checking for Open Bisimilarity in the π-Calculus*. February 2001. 61 pp.

7 Gregory Gutin, Khee Meng Koh, Eng Guan Tay, and Anders Yeo. *On the Number of Quasi-Kernels in Digraphs*. January 2001. 11 pp.

6 Gregory Gutin, Anders Yeo, and Alexey Zverovich. *Traveling Salesman Should not be Greedy: Domination Analysis of Greedy-Type Heuristics for the TSP*. January 2001. 7 pp.

5 Thomas S. Hune, Judi Romijn, Mariëlle Stoelinga, and Frits W. Vaandrager. *Linear Parametric Model Checking of Timed Automata*. January 2001. 44 pp. To appear in Margaria and Yi, editors, *Tools and Algorithms for The Construction and Analysis of Systems: 7th International Conference*, TACAS '01 Proceedings, LNCS, 2001.

4 Gerd Behrmann, Ansgar Fehnker, Thomas S. Hune, Kim G. Larsen, Paul Pettersson, and Judi Romijn. *Efficient Guiding Towards Cost-Optimality in* UPPAAL. January 2001. 21 pp. To appear in Margaria and Yi, editors, *Tools and Algorithms for The Construction and Analysis of Systems: 7th International Conference*, TACAS '01 Proceedings, LNCS, 2001.

3 Gerd Behrmann, Ansgar Fehnker, Thomas S. Hune, Kim G. Larsen, Paul Pettersson, Judi Romijn, and Frits W. Vaandrager. *Minimum-Cost Reachability for Priced Timed Automata*. January 2001. 22 pp. Appears in Di Benedetto and Sangiovanni-Vincentelli, editors, *Hybrid Systems: Computation and Control, Fourth International Workshop*, HSCC '01 Proceedings, LNCS 2034, 1999, 147–161.

2 Rasmus Pagh and Jakob Pagter. *Optimal Time-Space Trade-Offs for Non-Comparison-Based Sorting*. January 2001. ii+20 pp.

1 Gerth Stølting Brodal, Rolf Fagerberg, Christian N. S. Pedersen, and Anna Östlin. *The Complexity of Constructing Evolutionary Trees Using Experiments*. January 2001. 28 pp.

52 Claude Crépeau, Frédéric Légaré, and Louis Salvail. *How to Convert a Flavor of Quantum Bit Commitment*. December 2000. 24 pp. To appear in Pfitzmann, editor, *Advances in Cryptology: International Conference on the Theory and Application of Cryptographic Techniques*, EUROCRYPT '01 Proceedings, LNCS, 2001.

51 Peter D. Mosses. *CASL for CafeOBJ Users*. December 2000. 25 pp. Appears in Futatsugi, Nakagawa and Tamai, editors, *CAFE: An Industrial-Strength Algebraic Formal Method*, 2000, chapter 6, pages 121–144.

50 Peter D. Mosses. *Modularity in Meta-Languages*. December 2000. 19 pp. Appears in Despeyroux, editor, *2nd Workshop on Logical Frameworks and Meta-Languages*, LFM '00 Proceedings, 2000, pages 1–18.

49 Ulrich Kohlenbach. *Higher Order Reverse Mathematics*. December 2000. 18 pp.

48 Marcin Jurdziński and Jens Vöge. *A Discrete Stratety Improvement Algorithm for Solving Parity Games*. December 2000. 31 pp. Extended abstract appears in Emerson and Sistla, editors, *Computer-Aided Verification: 11th International Conference*, CAV '00 Proceedings, LNCS 1855, 2000, pages 202–215.

47 Lasse R. Nielsen. *A Denotational Investigation of Defunctionalization*. December 2000. 50 pp. Presented at *16th Workshop on the Mathematical Foundations of Programming Semantics*, MFPS '00 (Hoboken, New Jersey, USA, April 13–16, 2000).

46 Zhe Yang. *Reasoning About Code-Generation in Two-Level Languages*. December 2000.

45 Ivan B. Damgård and Mads J. Jurik. *A Generalisation, a Simplification and some Applications of Paillier's Probabilistic Public-Key System*. December 2000. 18 pp. Appears in Kim, editor, *Fourth International Workshop on Practice and Theory in Public Key Cryptography*, PKC '01 Proceedings, LNCS 1992, 2001, pages 119–136. This revised and extended report supersedes the earlier BRICS report RS-00-5.

44 Bernd Grobauer and Zhe Yang. *The Second Futamura Projection for Type-Directed Partial Evaluation*. December 2000. To appear in *Higher-Order and Symbolic Computation*. This revised and extended report supersedes the earlier BRICS report RS-99-40 which in turn was an extended version of Lawall, editor, *ACM SIGPLAN Workshop on Partial Evaluation and Semantics-Based Program Manipulation*, PEPM '00 Proc., 2000, pages 22–32.

43 Claus Brabrand, Anders Møller, Mikkel Ricky Christensen, and Michael I. Schwartzbach. *PowerForms: Declarative Client-Side Form Field Validation*. December 2000. 21 pp. Appears in *World Wide Web Journal*, 4(3), 2000.

42 Claus Brabrand, Anders Møller, and Michael I. Schwartzbach. *The* `<bigwig>` *Project*. December 2000. 25 pp.

41 Nils Klarlund, Anders Møller, and Michael I. Schwartzbach. *The DSD Schema Language and its Applications*. December 2000. 32 pp. Shorter version appears in Heimdahl, editor, *3rd ACM SIGSOFT Workshop on on Formal Methods in Software Practice*, FMSP '00 Proceedings, 2000, pages 101–111.

40 Nils Klarlund, Anders Møller, and Michael I. Schwartzbach. *MONA Implementation Secrets*. December 2000. 19 pp. Shorter version appears in Yu and Păum, editors, *Fifth International Conference on Implementation and Application of Automata*, CIAA '00 Pre-Proceedings, LNCS 2088, 2001, pages 182–194.

39 Anders Møller and Michael I. Schwartzbach. *The Pointer Assertion Logic Engine*. December 2000. 23 pp. To appear in *ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '01 Proceedings, 2001.

38 Bertrand Jeannet. *Dynamic Partitioning in Linear Relation Analysis: Application to the Verification of Synchronous Programs*. December 2000. 44 pp.

37 Thomas S. Hune, Kim G. Larsen, and Paul Pettersson. *Guided Synthesis of Control Programs for a Batch Plant using* UPPAAL. December 2000. 29 pp. Appears in Lai, editor, *International Workshop in Distributed Systems Validation and Verification. Held in conjunction with 20th IEEE International Conference on Distributed Computing Systems (ICDCS '2000)*, DSVV '00 Proceedings, 2000, pages E15–E22.

36 Rasmus Pagh. *Dispersing Hash Functions*. December 2000. 18 pp. Preliminary version appeared in Rolim, editor, *4th International Workshop on Randomization and Approximation Techniques in Computer Science*, RANDOM '00, Proc. in Informatics, 2000, pages 53–67.

35 Olivier Danvy and Lasse R. Nielsen. *CPS Transformation of Beta-Redexes*. December 2000. 12 pp.

34 Olivier Danvy and Morten Rhiger. *A Simple Take on Typed Abstract Syntax in Haskell-like Languages*. December 2000. 25 pp. Appears in Kuchen and Ueda, editors, *Fifth International Symposium on Functional and Logic Programming*, FLOPS '01 Proceedings, LNCS 2024, 2001, pages 343–358.

33 Olivier Danvy and Lasse R. Nielsen. *A Higher-Order Colon Translation*. December 2000. 17 pp. Appears in Kuchen and Ueda, editors, *Fifth International Symposium on Functional and Logic Programming*, FLOPS '01 Proceedings, LNCS 2024, 2001, pages 78–91.

32 John C. Reynolds. *The Meaning of Types — From Intrinsic to Extrinsic Semantics*. December 2000. 35 pp.

31 Bernd Grobauer and Julia L. Lawall. *Partial Evaluation of Pattern Matching in Strings, revisited*. November 2000. 48 pp.

30 Ivan B. Damgård and Maciej Koprowski. *Practical Threshold RSA Signatures Without a Trusted Dealer*. November 2000. 14 pp. To appear in Pfitzmann, editor, *Advances in Cryptology: International Conference on the Theory and Application of Cryptographic Techniques*, EUROCRYPT '01 Proceedings, LNCS, 2001.

29 Luigi Santocanale. *The Alternation Hierarchy for the Theory of μ-lattices*. November 2000. 44 pp. Extended abstract appears in *Abstracts from the International Summer Conference in Category Theory*, CT2000, Como, Italy, July 16–22, 2000.

28 Luigi Santocanale. *Free μ-lattices*. November 2000. 51 pp. Short abstract appeared in *Proceedings of Category Theory 99*, Coimbra, Portugal, July 19–24, 1999. Full version to appear in a special conference issue of the *Journal of Pure and Applied Algebra*.

27 Zoltán Ésik and Werner Kuich. *Inductive *-Semirings*. October 2000. 34 pp.

26 František Čapkovič. *Modelling and Control of Discrete Event Dynamic Systems*. October 2000. 58 pp.

25 Zoltán Ésik. *Continuous Additive Algebras and Injective Simulations of Synchronization Trees*. September 2000. 41 pp.

24 Claus Brabrand and Michael I. Schwartzbach. *Growing Languages with Metamorphic Syntax Macros*. September 2000. 22 pp.

23 Luca Aceto, Anna Ingólfsdóttir, Mikkel Lykke Pedersen, and Jan Poulsen. *Characteristic Formulae for Timed Automata*. September 2000. 23 pp. To appear in *RAIRO, Theoretical Informatics and Applications*.

22 Thomas S. Hune and Anders B. Sandholm. *Using Automata in Control Synthesis — A Case Study*. September 2000. 20 pp. Appears in Maibaum, editor, *Fundamental Approaches to Software Engineering: First International Conference*, FASE '00 Proceedings, LNCS 1783, 2000, pages 349–362.

21 M. Oliver Möller and Rajeev Alur. *Heuristics for Hierarchical Partitioning with Application to Model Checking*. August 2000. 30 pp.

20 Luca Aceto, Willem Jan Fokkink, and Anna Ingólfsdóttir. *2-Nested Simulation is not Finitely Equationally Axiomatizable*. August 2000. 13 pp.

19 Vinodchandran N. Variyam. *A Note on* $\mathbf{NP} \cap \mathbf{coNP}/\mathrm{poly}$. August 2000. 7 pp.

18 Federico Crazzolara and Glynn Winskel. *Language, Semantics, and Methods for Cryptographic Protocols*. August 2000. ii+42 pp.

17 Thomas S. Hune. *Modeling a Language for Embedded Systems in Timed Automata*. August 2000. 26 pp. Earlier version entitled *Modelling a Real-Time Language* appeared in Gnesi and Latella, editors, *Fourth International ERCIM Workshop on Formal Methods for Industrial Critical Systems*, FMICS '99 Proceedings of the FLoC Workshop, 1999, pages 259–282.

16 Jiří Srba. *Complexity of Weak Bisimilarity and Regularity for BPA and BPP*. June 2000. 20 pp. To appear in Aceto and Victor, editors, *Expressiveness in Concurrency: 7th International Workshop EXPRESS '00 Proceedings*, ENTCS, 2000.

15 Daniel Damian and Olivier Danvy. *Syntactic Accidents in Program Analysis: On the Impact of the CPS Transformation*. June 2000. 26 pp. Extended version of an article appearing in Wadler, editor, *Proceedings of the fifth ACM SIGPLAN International Conference on Functional Programming*, 2000, pages 209–220.

14 Ronald Cramer, Ivan B. Damgård, and Jesper Buus Nielsen. *Multiparty Computation from Threshold Homomorphic Encryption*. June 2000. ii+38 pp. To appear in Pfitzmann, editor, *Advances in Cryptology: International Conference on the Theory and Application of Cryptographic Techniques*, EUROCRYPT '01 Proceedings, LNCS, 2001.

13 Ondřej Klíma and Jiří Srba. *Matching Modulo Associativity and Idempotency is NP-Complete*. June 2000. 19 pp. Appeared in Nielsen and Rovan, editors, *Mathematical Foundations of Computer Science: 25th International Symposium*, MFCS '00 Proceedings, LNCS 1893, 2000, pages 456–466.

12 Ulrich Kohlenbach. *Intuitionistic Choice and Restricted Classical Logic*. May 2000. 9 pp. To appear in *Mathematical Logic Quaterly*, 2001.

11 Jakob Pagter. *On Ajtai's Lower Bound Technique for R-way Branching Programs and the Hamming Distance Problem*. May 2000. 18 pp.

10 Stefan Dantchev and Søren Riis. *A Tough Nut for Tree Resolution*. May 2000. 13 pp.

9 Ulrich Kohlenbach. *Effective Uniform Bounds on the Krasnoselski-Mann Iteration*. May 2000. 34 pp.

8 Nabil H. Mustafa and Aleksandar Pekeč. *Democratic Consensus and the Local Majority Rule*. May 2000. 38 pp.

7 Lars Arge and Jakob Pagter. *I/O-Space Trade-Offs*. April 2000. Appears in Halldórsson, editor, *7th Scandinavian Workshop on Algorithm Theory*, SWAT '98 Proceedings, LNCS 1851, 2000, pages 448–461.

6 Ivan B. Damgård and Jesper Buus Nielsen. *Improved Non-Committing Encryption Schemes based on a General Complexity Assumption*. March 2000. 24 pp. Appears in Bellare, editor, *Advances in Cryptology: 20th Annual International Cryptology Conference*, CRYPTO '00 Proceedings, LNCS 1880, 2000, pages 433–451.

5 Ivan B. Damgård and Mads J. Jurik. *Efficient Protocols based on Probabilistic Encryption using Composite Degree Residue Classes*. March 2000. 19 pp.

4 Rasmus Pagh. *A New Trade-off for Deterministic Dictionaries*. February 2000. Appears in Halldórsson, editor, *7th Scandinavian Workshop on Algorithm Theory*, SWAT '98 Proceedings, LNCS 1851, 2000, pages 22–31.

3 Fredrik Larsson, Paul Pettersson, and Wang Yi. *On Memory-Block Traversal Problems in Model Checking Timed Systems*. January 2000. 15 pp. Appears in Graf and Schwartzbach, editors, *Tools and Algorithms for The Construction and Analysis of Systems: 6th International Conference*, TACAS '00 Proceedings, LNCS 1785, 2000, pages 127–141.

2 Igor Walukiewicz. *Local Logics for Traces*. January 2000. 30 pp.

1 Rune B. Lyngsø and Christian N. S. Pedersen. *Pseudoknots in RNA Secondary Structures*. January 2000. 15 pp. Appears in Shamir, editor, *Fourth Annual International Conference on Computational Molecular Biology*, RECOMB '00 Proceedings, 2000, 201–209 and in *Journal of Computational Biology*, 7(3/4):409–427, 2000.

57 Peter D. Mosses. *A Modular SOS for ML Concurrency Primitives*. December 1999. 22 pp.

56 Peter D. Mosses. *A Modular SOS for Action Notation*. December 1999. 39 pp. Full version of paper appearing in Mosses and Watt, editors, *Second International Workshop on Action Semantics*, AS '99 Proceedings, BRICS Notes Series NS-99-3, 1999, pages 131–142.

55 Peter D. Mosses. *Logical Specification of Operational Semantics*. December 1999. 18 pp. Invited paper. Appears in Flum, Rodríguez-Artalejo and Mario, editors, *European Association for Computer Science Logic: 13th International Workshop*, CSL '99 Proceedings, LNCS 1683, 1999, pages 32–49.

54 Peter D. Mosses. *Foundations of Modular SOS*. December 1999. 17 pp. Full version of paper appearing in Kutyłowski, Pacholski and Wierzbicki, editors, *Mathematical Foundations of Computer Science: 24th International Symposium*, MFCS '99 Proceedings, LNCS 1672, 1999, pages 70–80.

53 Torsten K. Iversen, Kåre J. Kristoffersen, Kim G. Larsen, Morten Laursen, Rune G. Madsen, Steffen K. Mortensen, Paul Pettersson, and Chris B. Thomasen. *Model-Checking Real-Time Control Programs — Verifying LEGO Mindstorms Systems Using UPPAAL*. December 1999. 9 pp. To appear in Toetenel, editor, *12th Euromicro Conference on Real-Time Systems*, ECRTS '00 Proceedings, 2000.

52 Jesper G. Henriksen, Madhavan Mukund, K. Narayan Kumar, and P. S. Thiagarajan. *Towards a Theory of Regular MSC Languages*. December 1999. 43 pp.

51 Olivier Danvy. *Formalizing Implementation Strategies for First-Class Continuations*. December 1999. Extended version of an article to appear in Smolka, editor, *Programming Languages and Systems: Ninth European Symposium on Programming*, ESOP '00 Proceedings, LNCS 1782, 2000.

50 Gerth Stølting Brodal and Srinivasan Venkatesh. *Improved Bounds for Dictionary Look-up with One Error*. December 1999. 5 pp. Appears in *Information Processing Letters* 75(1–2):57–59, 2000.

49 Alexander A. Ageev and Maxim I. Sviridenko. *An Approximation Algorithm for Hypergraph Max k-Cut with Given Sizes of Parts*. December 1999. 12 pp. Appears in Paterson, editor, *Eighteenth Annual European Symposiumon on Algorithms*, ESA '00 Proceedings, LNCS 1879, 2000, pages 32–49.

48 Rasmus Pagh. *Faster Deterministic Dictionaries*. December 1999. 14 pp. Appears in Shmoys, editor, *The Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '00 Proceedings, 2000, pages 487–493.

47 Peter Bro Miltersen and Vinodchandran N. Variyam. *Derandomizing Arthur-Merlin Games using Hitting Sets*. December 1999. 21 pp. Appears in Beame, editor, *40th Annual Symposium on Foundations of Computer Science*, FOCS '99 Proceedings, 1999, pages 71–80.

46 Peter Bro Miltersen, Vinodchandran N. Variyam, and Osamu Watanabe. *Super-Polynomial Versus Half-Exponential Circuit Size*

in the Exponential Hierarchy. December 1999. 14 pp. Appears in Asano, Imai, Lee, Nakano and Tokuyama, editors, *Computing and Combinatorics: 5th Annual International Conference*, COCOON '99 Proceedings, LNCS 1627, 1999, pages 210–220.

45 Torben Amtoft. *Partial Evaluation for Constraint-Based Program Analyses*. December 1999. 13 pp.

44 Uwe Nestmann, Hans Hüttel, Josva Kleist, and Massimo Merro. *Aliasing Models for Mobile Objects*. December 1999. ii+46 pp. To appear in a special FOOL6 issue of *Information and Computation*. An extended abstract of this revision, entitled *Aliasing Models for Object Migration*, appeared as Distinguished Paper in Amestoy, Berger, Daydé, Duff, Frayssé, Giraud and Daniel, editors, *5th International Euro-Par Conference*, EURO-PAR '99 Proceedings, LNCS 1685, 1999, pages 1353–1368, which in turn is a revised part of another paper called *Migration = Cloning ; Aliasing* that appeared in Cardelli, editor, *Foundations of Object-Oriented: 6th International Conference*, FOOL6 Informal Proceedings, 1999 and as such supersedes the corresponding part of the earlier BRICS report RS-98-33.

43 Uwe Nestmann. *What is a 'Good' Encoding of Guarded Choice?* December 1999. ii+34 pp. To appear in a special EXPRESS '97 issue of *Information and Computation*. This revised report supersedes the earlier BRICS report RS-97-45.

42 Uwe Nestmann and Benjamin C. Pierce. *Decoding Choice Encodings*. December 1999. ii+62 pp. To appear in *Journal of Information and Computation*. An extended abstract appeared in Montanari and Sassone, editors, *Concurrency Theory: 7th International Conference*, CONCUR '96 Proceedings, LNCS 1119, 1996, pages 179–194.

41 Nicky O. Bodentien, Jacob Vestergaard, Jakob Friis, Kåre J. Kristoffersen, and Kim G. Larsen. *Verification of State/Event Systems by Quotienting*. December 1999. 17 pp. Presented at *Nordic Workshop in Programming Theory*, Uppsala, Sweden, October 6–8, 1999.

40 Bernd Grobauer and Zhe Yang. *The Second Futamura Projection for Type-Directed Partial Evaluation*. November 1999. 44 pp. Extended version of an article appearing in Lawall, editor, *ACM SIGPLAN Workshop on Partial Evaluation and Semantics-Based Program Manipulation*, PEPM '00 Proc., 2000, pages 22–32.

39 Romeo Rizzi. *On the Steiner Tree $\frac{3}{2}$-Approximation for Quasi-Bipartite Graphs*. November 1999. 6 pp.

38 Romeo Rizzi. *Linear Time Recognition of P_4-Indifferent Graphs*. November 1999. 11 pp.

37 Tibor Jordán. *Constrained Edge-Splitting Problems*. November 1999. 23 pp. A preliminary version with the title *Edge-Splitting Problems with Demands* appeared in Cornujols, Burkard and Wöginger, editors, *Integer Programming and Combinatorial Optimization: 7th International Conference*, IPCO '99 Proceedings, LNCS 1610, 1999, pages 273–288.

36 Gian Luca Cattani and Glynn Winskel. *Presheaf Models for* **CCS***-like Languages*. November 1999. ii+46 pp.

35 Tibor Jordán and Zoltán Szigeti. *Detachments Preserving Local Edge-Connectivity of Graphs*. November 1999. 16 pp.

New in the BRICS Notes Series

ISSN 0909-3206

8 Anders Møller and Michael I. Schwartzbach. *The XML Revolution (Revised)*. December 2001. 186 pp. This revised and extended report superseds the earlier BRICS Report NS-00-8.

7 Patrick Cousot, Lisbeth Fajstrup, Eric Goubault, Jeremy Gunawardena, Maurice Herlihy, Martin Raussen, and Vladimiro Sassone, editors. *Preliminary Proceedings of the*

Workshop on Geometry and Topology in Concurrency Theory, GETCO '01, (Aalborg, Denmark, August 25, 2001), August 2001. vi+97 pp.

6 Luca Aceto and Prakash Panangaden, editors. *Preliminary Proceedings of the 8th International Workshop on Expressiveness in Concurrency, EXPRESS '01*, (Aalborg, Denmark, August 20, 2001), August 2001. vi+139 pp.

5 Flavio Corradini and Walter Vogler, editors. *Preliminary Proceedings of the 2nd International Workshop on Models for Time-Critical Systems, MTCS '01*, (Aalborg, Denmark, August 25, 2001), August 2001. vi+ 127pp.

4 Ed Brinksma and Jan Tretmans, editors. *Proceedings of the Workshop on Formal Approaches to Testing of Software, FATES '01*, (Aalborg, Denmark, August 25, 2001), August 2001. viii+156 pp.

3 Martin Hofmann, editor. *Proceedings of the 3rd International Workshop on Implicit Computational Complexity, ICC '01*, (Aarhus, Denmark, May 20–21, 2001), May 2001. vi+144 pp.

2 Stephen Brookes and Michael Mislove, editors. *Preliminary Proceedings of the 17th Annual Conference on Mathematical Foundations of Programming Semantics, MFPS '01*, (Aarhus, Denmark, May 24–27, 2001), May 2001. viii+279 pp.

1 Nils Klarlund and Anders Møller. MONA *Version 1.4 — User Manual*. January 2001. 83 pp.

8 Anders Møller and Michael I. Schwartzbach. *The XML Revolution*. December 2000. 149 pp.

7 Nils Klarlund, Anders Møller, and Michael I. Schwartzbach. *Document Structure Description 1.0*. December 2000. 40 pp.

6 Peter D. Mosses and Hermano Perrelli de Moura, editors. *Proceedings of the Third International Workshop on Action Semantics, AS 2000*, (Recife, Brazil, May 15–16, 2000), August 2000. viii+148 pp.

5 Claus Brabrand. `<bigwig>` *Version 1.3 — Tutorial*. September 2000. ii+92 pp.

4 Claus Brabrand. `<bigwig>` *Version 1.3 — Reference Manual*. September 2000. ii+56 pp.

3 Patrick Cousot, Eric Goubault, Jeremy Gunawardena, Maurice Herlihy, Martin Raussen, and Vladimiro Sassone, editors. *Preliminary Proceedings of the Workshop on Geometry and Topology in Concurrency Theory, GETCO '00*, (State College, USA, August 21, 2000), August 2000. vi+116 pp.

2 Luca Aceto and Björn Victor, editors. *Preliminary Proceedings of the 7th International Workshop on Expressiveness in Concurrency, EXPRESS '00*, (State College, USA, August 21, 2000), August 2000. vi+130 pp.

1 Bernd Gärtner. *Randomization and Abstraction — Useful Tools for Optimization*. February 2000. 106 pp.

BRICS Lecture Series

ISSN 1395-2048

1 Zsolt Tuza. *Unsolved Combinatorial Problems, Part I*. May 2001. viii+30 pp. **Abstract:**

Contents

11 Weighted edge covering with complete subgraphs

12 Strongly triangle-free subgraphs

13 Excluded cycle lengths, chromatic number, and orientations

14 The Acyclic Orientation Game

15 Transversals of uniform hypergraphs

16 Covering and coloring the maximal complete subgraphs

BRICS Dissertations Series

ISSN 1396-7002

10 Mikkel T. Jensen. *Robust and Flexible Scheduling with Evolutionary Computation*. November 2001. PhD thesis. xii+299 pp.

9 Flemming Friche Rodler. *Compression with Fast Random Access*. November 2001. PhD thesis. xiv+123 pp.

8 Niels Damgaard. *Using Theory to Make Better Tools*. October 2001. PhD thesis.

7 Lasse R. Nielsen. *A Study of Defunctionalization and Continuation-Passing Style*. August 2001. PhD thesis. iv+280 pp.

6 Bernd Grobauer. *Topics in Semantics-based Program Manipulation*. August 2001. PhD thesis. ii+x+186 pp.

5 Daniel Damian. *On Static and Dynamic Control-Flow Information in Program Analysis and Transformation*. August 2001. PhD thesis. xii+111 pp.

4 Morten Rhiger. *Higher-Order Program Generation*. August 2001. PhD thesis. xiv+146 pp.

3 Thomas S. Hune. *Analyzing Real-Time Systems: Theory and Tools*. March 2001. PhD thesis. xii+265 pp.

2 Jakob Pagter. *Time-Space Trade-Offs*. March 2001. PhD thesis. xii+83 pp.

1 Stefan Dziembowski. *Multiparty Computations — Information-Theoretically Secure Against an Adaptive Adversary*. January 2001. PhD thesis. 109 pp.

7 Marcin Jurdziński. *Games for Verification: Algorithmic Issues*. December 2000. PhD thesis. ii+112 pp.

6 Jesper G. Henriksen. *Logics and Automata for Verification: Expressiveness and Decidability Issues*. May 2000. PhD thesis. xiv+229 pp.

5 Rune B. Lyngsø. *Computational Biology*. March 2000. PhD thesis. xii+173 pp.

4 Christian N. S. Pedersen. *Algorithms in Computational Biology*. March 2000. PhD thesis. xii+210 pp.

3 Theis Rauhe. *Complexity of Data Structures (Unrevised)*. March 2000. PhD thesis. xii+115 pp.

2 Anders B. Sandholm. *Programming Languages: Design, Analysis, and Semantics*. February 2000. PhD thesis. xiv+233 pp.

1 Thomas Troels Hildebrandt. *Categorical Models for Concurrency: Independence, Fairness and Dataflow*. February 2000. PhD thesis. x+141 pp.

1 Gian Luca Cattani. *Presheaf Models for Concurrency (Unrevised)*. April 1999. PhD thesis. xiv+255 pp.

News

Short News

-

  Glynn Winskel has been appointed professor at the University of Cambridge Computer Laboratory, UK.

-

  Erik Meineche Schmidt has been elected as the new Dean of the Faculty of Science at the University of Aarhus from February 1, 2002.

-

  Kim G. Larsen has obtained an industrial CMG professorship at the University of Twente, Holland, including a grant for mutual visits.

-

  July 2000 *Mogens Nielsen* was elected Vice President for EATCS — the European Association for Theoretical Computer Science.

-

  Jotun Hein has been appointed professor at Department of Statistics, University of Oxford, UK.

-

  In May 2000 Maciej Koprowski presented his Master Degree thesis at IV National Conference of Cryptography Applications Enigma 2000 in Warsaw, Poland. He received 1st award for Polish MSc. thesis in the field of cryptology and information security.

- BRICS is now hosting the official EATCS web site: `www.eatcs.org`. The services of the site are powered by `<bigwig>`.

- New Release: UPPAAL2K 3.2.1 (released October 29, 2001) is now available. The new features include: liveness and deadlock properties in the query language, and a new—XML based—file format. For more information about UPPAAL2k and this new release of UPPAAL2K, follow the links on `www.cs.auc.dk/research/FS/research/uppaal/`.

- Version 2.0 of the `<bigwig>` compiler and runtime system has been released. This new version contains a program analysis for checking that programs always generate valid HTML, advanced caching of dynamically generated Web pages, an efficient Apache module for running `<bigwig>` services, ”seslets” for easy communication between applets and the Web server, and easier installation and configuration. Please see `www.brics.dk/bigwig/`.

- Recently a revised edition of *The XML Revolution*, a comprehensive online tutorial on XML and related technologies, has been published. So far, the online version has given the BRICS web server 1.2 million hits from 58,000 persons. See [NS-01-8] and `www.brics.dk/~amoeller/XML/`.

- `dk.brics.automaton` is a fast and compact Java implementation of finite-state automata and regular expressions (including complement) with Unicode alphabet. The implementation is developed by *Anders Møller* and available at `www.brics.dk/automaton/`.

Positions at BRICS

This is a call for applications for:

Research Positions,
PhD admission and PhD grants, and
Marie Curie Fellowships.

BRICS, Basic Research in Computer Science, is funded by the Danish National Research Foundation. It comprises an International PhD School with an associated Research Laboratory.

BRICS is based on a commitment to develop theoretical computer science, covering core areas such as:

- Semantics of Computation,
- Logic,
- Algorithms and Data Structures,
- Complexity Theory,
- Data Security and Cryptology, and
- Verification,

as well as a number of spin-off activities including

- Web Technology,
- Quantum Informatics,
- Bio Informatics, and
- Networks and Distributed Real-Time Systems.

BRICS has a number of new research positions, PhD grants, and fellowships available, starting in 2002. Applications can be submitted at any time. However, the application deadline for PhD grants and positions starting August 2002 is February 15, 2002.

Research positions: These positions are open to applicants already holding a PhD degree in computer science. At BRICS in Aalborg we particularly seek applications within the following areas:

- Verification and test of embedded real-time and hybrid systems,
- Semantics of concurrency,
- Modelling and reasoning about objects and mobile processes,
- Networks and firewalls, and
- Distributed and real-time systems.

PhD admission and grants: Any student with at least four years of studies in computer science is eligible to apply for PhD admission and grants.

Marie Curie Fellowships: These fellowships are offered within Interactive Computation - Methodology, Security, and Efficiency - and are open to PhD students who are nationals of a member state of the European Community or an associated state, and who wish to spend time (from three months to twelve months) at BRICS, as part of their PhD studies.

Further information, including instructions on how to apply, can be found at: www.brics.dk/Positions.

You are also welcome to contact one of the BRICS directors:

- *Mogens Nielsen*, Aarhus, mn@brics.dk
- *Kim G. Larsen*, Aalborg, kgl@brics.dk

ALCOM-FT: Future Technologies

Rolf Fagerberg

One of the many activities of BRICS is the participation in the project ALCOM-FT. This project is a collaboration between the algorithm groups of ten universities and research institutions in Europe, including the group of BRICS. The project runs from June 2000 to June 2003, and is supported by the EU.

The collaboration actually goes back quite some time, with ALCOM-FT being the fourth incarnation of the ALCOM consortium, and besides BRICS it includes the groups headed by *Josep Díaz* (Barcelona), *Michael Jünger* (Cologne), *Philippe Flajolet* (INRIA, Rocquencourt), *Kurt Mehlhorn* (Max-Planck-Institut für Informatik, Saarbrücken), *Friedhelm Meyer auf der Heide* and *Burkhard Monien* (Paderborn), *Paul Spirakis* (Computer Technology Institute, Patras), Giorgio Ausiello (Rome), *Jan van Leeuwen* (Utrecht), and *Mike Paterson* (Warwick). The acronym ALCOM itself stands for ALgorithms and COMplexity.

Over the past ten years, the consortium has contributed significantly to the high profile of European algorithms research, and has its particular strengths in the areas of long-term basic research and in transfer of knowledge from academia to industry. The large LEDA C++ library of data types and algorithms (www.mpi-sb.mpg.de/LEDA) is a good example of the latter effort.

The main themes of the current project are *massive data sets*, *networks and communication*, and *production and transportation planning*. These areas are attacked by the well-proven methods of theoretical analysis, as well as by newer methods from experimental algorithmics—a subject which is currently receiving increasing attention in the algorithms community. The aim is not only to improve the state of the art in the three areas mentioned, but also to develop the methodologies used. The main outcome (“deliverables” in EU terminology) of the project is research papers, software implementing advanced algorithms, and dissemination of algorithmic knowledge through summer schools and web sites. The productivity has been high, with 174 research papers produced during the first year of ALCOM-FT, the majority of which has already appeared at conferences and in journals.

Further information can be found at the website for ALCOM-FT at www.brics.dk/ALCOM-FT, where also the research reports and other deliverables of the project can be accessed on-line.

Automated Software Testing

Brian Nielsen

BRICS at Department of Computer Science at Aalborg University and Siemens Mobile Phones, Aalborg, have started a collaboration on automated software testing. Starting from may 2001, Brian Nielsen is employed as the primary researcher, and is in part sponsored by Siemens Mobile Phones.

The goal of automated testing is to increase its thoroughness by executing more tests and generating those systematically, and to reduce the time spent on testing by reducing the amount of laborious and tedious construction and validation of test cases. There are two aspects of test automatisation: The first is automatic synthesis of test cases; various degrees of tool support are possible ranging from completely automated to guidance of tools via engineerer specified partial event sequences or test purposes. The second aspect is automatic execution of test cases.

During the recent years new techniques and tools based on formal methods have been developed for automated (blackbox) conformance testing of subsystems like communication protocols and the interfaces of embedded systems. Conformance testing aims at checking whether the behaviour of the target system complies with that of its specification. Such test cases can be (completely and semi) automatically constructed from specifications of the required and desired behaviour of the target system. The specifications are given as state machine descriptions, or via formalism such as UML state machines or SDL descriptions that can be translated to state-machines.

Whereas reasonably mature (even commercially available) tools exist for non-real time systems, they lack explicit and systematic support for real-time. AAU possesses substantial expertise in specification, analysis and generation of tests for real-time systems. This project will challenge the automated testing techniques and tools—both in general and for real-time systems—with real applications from the area of mobile phones. The project will further show how such techniques can be applied in the context of mobile phones by attempting to apply and adapt them to this problem domain.

See also www.cs.auc.dk/~bnielsen/AutomatedTesting/.

BiRC: Bioinformatics Research Center

Jotun Hein, Erik M. Schmidt, and Christian N. S. Pedersen

The Bioinformatics Research Center (BiRC) at the University of Aarhus was established in January 2001 as a joint venture between the Faculty of Science, the Faculty of Health Sciences and Aarhus University Hospital. The activities of the Center will draw on a grant from the Aarhus University Research Foundation amounting to DKK 10 million over four years.

BRICS and the Department of Computer Science at the University of Aarhus (DAIMI) are actively involved in both the scientific and administrative aspects of BiRC.

Currently one BRICS employed researcher works full time at BiRC, and a number of students are employed as student programmers. In March 2001, BiRC moved into the Officers Building previously occupied by BRICS.

Figure 17: DNA, life's building-blocks, looking out across the campus.

DNA, proteins and the working of cells

Bioinformatics is an interdisciplinary area that can be described as the application of computer science, statistics, mathematics, chemistry and physics to problems in the fields of biology and medicine. To a great extent, the work consists of developing and applying models, algorithms and computer programs for the gathering, processing and analysis of molecular data. Sequence data is particularly important in this context.

The genetic information for an organism is stored in the chemical structure of its DNA. This can be described in terms of a simple sequence of the letters A, C, G and T, which clearly makes it amenable to computer-based analysis. The DNA sequence encodes the molecular structure of the proteins, among other things, since it can be read as a sequence of amino-acids, the building-blocks of the protein. Other types of data studied include molecular structures, for example the three-dimensional structure of proteins, and information on the functioning of cells, such as gene expression data, which can tell us about the proteins produced by the cell.

More data than computational power

Biological data, especially sequence and gene expression data, is being collected so fast that

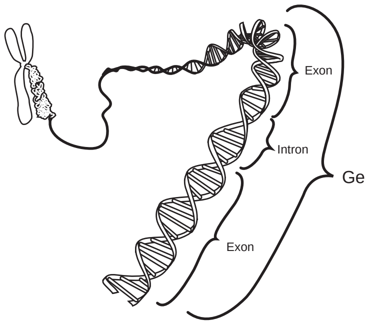
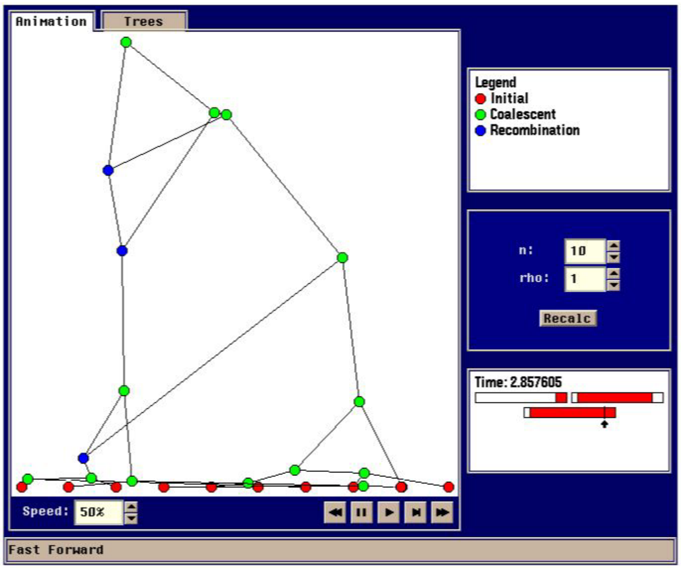


Figure 18: A DNA molecule consists of two strings intertwined in a double helix. The chemical structure of DNA can be described in terms of a sequence of As, Cs, Gs and Ts.

within a few years it will be difficult to use all the available information that is relevant to a particular problem. Even with the very fastest computer, it will only be possible to examine a minute part of the data available within a reasonable amount of time. The processing and analysis of the huge amounts of data is thus not simply a problem within the field of biology, but also represents a computational and conceptual challenge, and the solutions can be useful outside the field of bioinformatics.

The demand for expertise in bioinformatics has greatly increased in recent years. One important reason for this is the speed at which it is now possible to collect sequence data, i.e. to 'read' DNA sequences. The mapping of the human genome has been a milestone in this work. It has involved 'reading' the DNA sequence of human hereditary material, which consists of over three billion As, Cs, Gs and Ts. This hereditary material is estimated to contain approximately 50,000 genes, divided up into small regions of the complete DNA sequence.

Analysing and understanding the human genome is thus a huge bioinformatics task. The work will take many years and will in a sense never be completed. The complex of problems involves describing of the structure, function and action of the genes - in other words, gaining an understanding of what roles the coded proteins play in the human organism, and under which circumstances the instructions contained in the gene are carried out.



n: 10
rho: 1
Time: 2.857605
Speed: 50%
Fast Forward

Figure 19: Software development is extremely important in connection with the implementation of new methods of analysis in the fields of biology and medicine. The illustration shows a screen dump from a program designed to simulate sequential evolution.

Bioinformatics in Aarhus

The University of Aarhus has a long tradition for research and teaching in the field of bioinformatics. One of the local pioneers of bioinformatics is *Jotun Hein*, who until September 1 this year was an associate professor at the Department of Ecology and Genetics, and now is Professor in Bioinformatics at Oxford University. Over the past ten years Jotun Hein and colleagues have been doing bioinformatics research and have offered a fair number of bioinformatics courses which have been taken by students from biology, molecular biology, statistics and computer science. The result has been the establishment of an interdisciplinary group of researchers and students whose shared interest in bioinformatics cuts across departmental boundaries. At BRICS and DAIMI there has been close collaboration with Jotun Hein through student programmers and a number of master and PhD projects.

The establishment of BiRC in January 2001 has gathered together the interdisciplinary group of researchers and students who share an interest in bioinformatics both physically and “in spirit”. During the past year local bioinformatics activities have been further strengthened through the creation of new research positions and through input from suitable partners, both within and outside the University. Two problems currently in focus at BiRC are gene finding and gene expression analysis.

Gene finding

Searching for the regions of a DNA sequence which make up a gene, is a central problem in bioinformatics. Constructing a good computational gene finder requires a substantial knowledge of the biological mechanisms which are believed to be the reason why some regions of a DNA sequence are genes and others are not. But it also requires a substantial knowledge of algorithmic theory and computer science in general to analyze large DNA sequences efficiently.

At BiRC we are developing methods for comparative gene finding. Finding genes by comparing whole, or large parts, of genomes is motivated by evolutionary biology, which suggests that highly similar regions in genomes of different organisms, e.g. man and mouse, are likely candidates for being genes. The comparative approach to gene finding includes many computational challenges. For example, many of the algorithms previously used for sequence comparison have a quadratic time complexity which using todays technology make them unsuitable for comparison of sequences consisting of millions of characters.

Gene expression analysis

Finding the genes is an important task. The next step is to understand the function and action of the genes. Measuring gene expression levels using DNA chips is an experimental technique which makes it possible to gather efficiently information about which genes (from a known set of genes) are expressed in the cells from a tissue sample. This makes it possible to study the connections between gene expression levels and other observable characteristics of the cell, e.g. cancer. Mathematically, a DNA chip experiment yields a vector of values each describing the expression level of a particular gene. A typical DNA chip measures the expression levels of 10,000 to 50,000 genes. DNA chip experiments are fairly cheap, and huge amounts of gene expression data are being collected.

At BiRC we collaborate with molecular biologists who perform DNA chip experiments with self designed chips, and who need bioinformatics tools for analysing their data. The analysis of gene expression data includes many theoretical questions related to clustering and pattern recognition. A promising approach to gene expression analysis is to employ Bayesian networks and graphical models in general. We are also involved in the development of a WWW based database for storing the large amounts of gene expression data being collected at the Aarhus University Hospital in Skejby.

Conclusion

Bioinformatics and the activities at BiRC include many problems not mentioned above. For example structure prediction, modelling of sequence families, and reconstruction of evolutionary history. The aim of the Bioinformatics Research Center is to support the theoretical work that has been carried out so far, but also to be oriented toward the application of bioinformatics methods in practice. BiRC will also provide the basis for a new Master’s degree course in bioinformatics, which has been approved by the Ministry of Education. For more information about the activities at BiRC you are welcome to contact *Christian Nørgaard Storm Pedersen*, cstorm@brics.dk, or visit the BiRC WWW-site at www.birc.dk.

BRICS Appointed Marie Curie Training Site

Under the Fifth Framework Programme the European Community appointed BRICS in the autumn 2000 to be a Marie Curie Training Site. The grant covers 114 months for visiting PhD students.

BRICS as a Marie Curie Training Site offers doctoral training in the specific area of Interactive Computation. More concretely, training activities will be provided aiming at research in

- new programming methodologies required for the concept of global
- computing made possible by the web and its successors,
- solutions to the important security problems arising in this context, and
- efficient algorithms and data structures in the setting of interactive computation.

The training activities are aimed at fellows with a solid background in computer science (in particular semantics and algorithms). It is expected that BRICS will host 4–5 fellows annually, and in the selection it will apply equal opportunity regulations.

The first fellows were already successfully completed their stays at BRICS. Currently *Pablo Arrighi* (see page 17) is visiting BRICS and new fellows are lined up.

For further information on criteria, deadlines etc., please see page 47 and in particular `www.brics.dk/MarieCurie`.

CISS: Center for Embedded Software Systems[2]

Kim Guldstrand Larsen

Tomorrow's consumer electronics, and electronic products in general, will have to meet increasing demands on user friendliness, flexibility, internal control functionality, low physical size and weight as well as low power consumption. To achieve these goals there is a need for combining compact electronic modules with complex software realization of most functionality. In particular, mobile and wireless communication products as well as medico-technical equipment contain increasingly amount of software. In addition, seamless network access will be taken for granted in the future.

The newly started Center for Embedded Software Systems, CISS, aims at strengthening industrial competence, research and education within the area of embedded software systems. Particular attention will be given to products and devices whose individual components must typically be able to communicate and cooperate with other systems over networks. This is in accordance with the preparation of the 6th framework of the Research Program of the European Union, which recognises a strong need for intensified research and development within the area.

The CISS centre is established at Aalborg University based on three existing, internationally recognised research groups within the Institutes of Electronic Systems and Computer Science. One of the three units is the Aalborg wing of BRICS, contributing to the activities of CISS with expertise on modelling, validation and verification of embedded and real-time systems. The two other research units are the Distributed Real-Time Systems group with expertise in real-time control, field-busses, protocols and control

[2] 'Embedded'='Indlejrede' in Danish. For more information about CISS see `ciss.auc.dk`.

theory and the Embedded Systems group specialising HW/SW co-design, DSP and in low power implementations based on compiler techniques. All involved research groups have substantial experience with industrially collaboration on utilisation and development of technology.

CISS will create a new common forum, where engineers from industry and researchers from the university work on different multidisciplinary and industrial relevant R&D projects with the purpose of increasing the maturity level of the current practice within software development for embedded systems. The list of companies committed to take active part in CISS activities is at present (in random order): Grundfos A/S, S-Card A/S, Siemens MP A/S, Amplex A/S, Analog Devices A/S, LM Ericsson A/S, Maxon Cellular Systems, RTX Telecom A/S, Nokia MP A/S, Telital R&S Denmark A/S, Alpha Shipmate A/S, Danfoss A/S, Digianswer A/S, Bang&Olufsen, Magneti-Marelli/FIAT, ETI A/S, Simrad Shipmate A/S, Terma Elektronik A/S, and we hope more will join in the near future.

Typical activities within CISS will be i) collaborative projects between CISS (and BRICS) and industry ii) focused educational activities for industry iii) temporary exchange of staff. Also, CISS will draw on existing exchange agreements of BRICS, as well as coming agreements with EE&CS at Berkeley and Embedded Systems Lab in Eindhoven. Moreover, BRICS at Aalborg will serve as the danish node in the newly funded EU Network of Excellence, ARTIST (Advanced Real TIme SysTems) which covers all major sites in Europe within the area.

A generous donation by the regional council of Northern Jutland and the city council of Aalborg (700.000 DKK) has enabled CISS to start already in August 2001. Thus, a number of student projects are in action as well as more detailed specifications of several collaborative projects. A main CISS event was the CISS Trends & Visions Day on Development of Communicating Embedded Software Systems taking place on November 26. The visions were presented by four keynote speakers: *Robert Binder*, President of RBSC Corporation (Achieving High Reliability for Ubiquitous Information Technology), *Rolf Ernst*, Professor at Braunsweig University (Timing Analysis of Heterogeneous HW/SW Systems), *Gérard Berry*, Cheif Scientific Officer Esterel Technologies (Formal Design and Formal Verification of Embedded Applications), *Martin Astradsson*, Leader of Architecture Group, Nokia MP A/S (Software Design Methods for DSP Applications). The full-day meeting attracted some 90 people from industry indicating the high industrial interest in the area.

Figure 20: *Kim G. Larsen* and *Gérard Berry* at the CISS Trends &Visions Day.

Even though CISS is now in action and a reality, full scale operation still awaits the National Budget for 2002 and the level by which 7 selected IT-projects (including CISS) of the 'Jysk-Fynske Erhverssamarbejde' will be funded. So far the regional councils of Northern Jutland and Aalborg City has (conditionally) guaranteed 12M DKK out of a total remaining budget of 44M DKK. ▤

JWIG: Java Web Interface Generator

Aske S. Christensen, Anders Møller, and Michael I. Schwartzbach

JWIG is a Java-based development system for making high-level web services. It integrates the central features of the <bigwig> language into Java by providing explicit support for web service sessions and safe XHTML dynamic document construction. To support program development we provide a suite of program analyses that at compile-time verify for a given program that no runtime errors can occur while building documents or receiving form input, and that all documents being shown are valid according to their document type definition, for instance XHTML, WML, or VoiceXML. A prototype implementation is udner development. See www.brics.dk/JWIG/.

PALE: Pointer Assertion Logic Engine

Anders Møller and Michael I. Schwartzbach

Pointer Assertion Logic is a notation for expressing assertions about the heap structure of imperative languages. It allows programmers to specify pre- and post-conditions of procedures, loop invariants, and other assertions in Weak Monadic Second-order Logic of Graph Types. The main target applications are safety critical data type algorithms. PALE - the Pointer Assertion Logic Engine - is an experimental implementation of the technique, based on the MONA tool. It analyses an annotated program and reports null-pointer dereferences, memory leaks, and violations of assertions and graph type errors. See www.brics.dk/PALE/.

The Metafront System

Claus Brabrand and Michael I. Schwartzbach

Metafront is a domain specific language for extensible syntax processing. It takes a well-formed syntactic specification of a (programming) language and turns it into a top-down parser that always selects the most specific production applicable. Programmers are subsequently allowed to transparently extend this language through syntactic transformations (metamorphic syntax macros) by specifying how their new (meta) syntax is transformed into base language syntax. These syntax transformations are checked at definition time, guaranteeing that no parse errors can occur as a direct consequence of the syntax transformations (macro expansion).

The tool can serve either as an extensible *preprocessor* or *front-end*. As a preprocessor, it takes a base language specification and any number of meta language extensions and transforms meta language into base language by applying all syntax transformations. As a front-end, it executes any associated (Java) code actions in a programmer-specified order.

The first application of the tool will be to make Java syntax extensible.

The metafront tool is currently being developed at BRICS. Check out www.brics.dk/metafront/ for more information.

QAIP: Quantum Algorithms and Information Processing

Ivan B. Damgård

QAIP is an EU supported research program with the goal of doing new work in three main areas: Quantum Algorithms and Complexity, Quantum Cryptography and Fault Tolerant Quantum Computing. This covers all the main areas of computer science based research in quantum information processing.

Quantum Information (QI) generally means information encoded in the state of very small physical systems. So small, in fact, that the laws of quantum mechanics govern its behaviour. This means that QI, unlike its classical counterpart, cannot be copied or be measured completely reliably, but can on the other hand exist in superposition of several different states. This opens the way to forms of communication and computing that are not possible with classical means: a quantum computer can efficiently solve problems that are exceedingly difficult with conventional methods, and quantum cryptography makes it possible to exchange secret keys efficiently under the nose of an all powerful adversary.

The partners in QAIP are CWI, Amsterdam, the Netherlands. University of Latvia, Riga, Latvia. Oxford University, UK. University of Bristol, UK. University of Aarhus, Denmark. LRI, Paris, France. Hebrew University, Jerusalem, Israel. Weizmann Institute, Rehovot, Israel. Technion, Haifa, Israel. University of Waterloo, Canada. McGill University, Montreal, Canada. Université de Montréal, Canada. University of Calgary, Canada. University of California at Berkeley, USA.

See `www.cwi.nl/projects/QAIP/`.

SECURE: Secure Environments for Collaboration among Ubiquitous Roaming Entities

The SECURE project is funded by the EU IST FET programme — the Future and Emerging Technologies action of the major theme on Information Society Technologies within the European Union's Fifth RTD Framework Programme. The purpose of the FET action is to promote research that is of a longer-term nature.

The project was formed in response to the FET 2001 proactive initiative on Global Computation (Co-operation of Autonomous and Mobile Entities in Dynamic Environments) and will start at the beginning of 2002.

The SECURE project will investigate a new approach to security founded on the notion of trust. The project aims to develop a model in which trust relationships are established from the record of interaction between entities, and a security mechanism expressed in terms of such trust. SECURE will also investigate how to specify access control policy based on trust. The project will formally define a computational trust model and a collaboration model capturing the dynamic aspects of the trust model; means to specify and to enforce security policies based on trust; means to evaluate security policies and implementations based on trust; and algorithms for trust management.

The partners of the Secure project are: Trinity College Dublin, Ireland (*Vinny Cahill*, *Christian Jensen*), University of Cambridge, UK (*Jean Bacon*, *Ken Moody*), University of Geneva, Switzerland (*Ciaran Bryce*), University of Strathclyde, UK (*Paddy Nixon*), and University of Aarhus, Denmark (*Ivan B. Damgård*, *Michael I. Schwartzbach*, *Mogens Nielsen*)

The SECURE project is part of a Cross-programme Cluster, the Mobile Computing Cluster, of FET projects with common topics and objectives. Apart from SECURE, the cluster includes the MyTHs project (Models and Types for Security in Mobile Distributed Systems) with the sites Sussex, UK (*Vladimiro Sassone*) and ENS Paris, Ca Foscari Venezia, France and the MRG project (Mobile Resource Guarantees) with the sites Edinburgh, UK (*Donald Sannella*) and LMU Munich, Germany.

YakYak

Michael I. Schwartzbach, Nils Klarlund, and Niels Damgaard

YakYak is a preprocessor for the parser generator Bison. It allows you to write logical formulas in an extended Bison specification file. These logical formulas express properties of the parse tree. The truth value of the formulas can be used in the action code of the productions.

YakYak is especially useful when writing a parser where it is cumbersome to fit a grammar into the class accepted by parser generators. In this case, programmers usually make a too loose grammar and then check a number of side constraints by traversing the parse tree afterwards. However, it takes time to make these tree traversals correct and they are difficult to maintain when the underlying grammar is changing. YakYak can handle the side constraints for the programmer: the looser grammar is simply enriched with a number of side constraints specified in a simple and intuitive logic.

Get YakYak from `www.brics.dk/yakyak`.

Technical Contributions

Improving Firewalls using BRIC(K)S

Mikkel Christiansen and Emmanuel Fleury

Introduction

The Internet Firewall is a central element in the security of any organizations network-infrastructure. The firewall provides administrators with the ability to police the network traffic between the Internet and the organizations intranet. To fulfill such a rigorous task firewalls should be theoretically well founded to provide a strong platform for improving network security. Our observation is, on the contrary, that current firewalls are based on a fair design, but with little or no regard to using any form of a formalized framework[3]. Therefore we have set as our goal to provide a formalized framework for key elements of the firewall and thus increasing performance and providing a foundation that overall allows improved network security.

In the following sections we first give a brief introduction to Internet firewalls. We then describe two specific projects of BRICS in Aalborg that are currently in progress. Finally we conclude and describe potential future work. We should remark that the work described in this brief presentation can be described as "work in progress".

Background

Figure 21 shows an illustration of an Internet firewall. The firewall is used for controlling the flow of traffic between an protected and unprotected network, in this case an intranet and the Internet. Inside the firewall we organize filters on an information based hierarchy. In this order, we consider the *packet* as the smallest atomic information unit. The more the filter needs to remember about previous packets, the higher the filter is placed in the hierarchy.

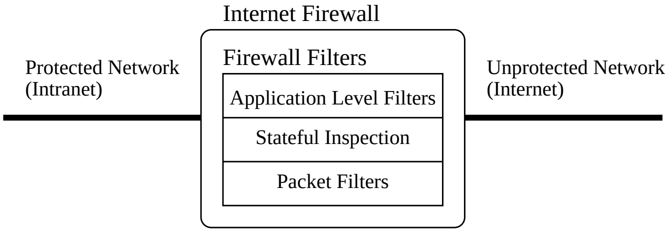


Figure 21: An illustration of an Internet Firewall

At the lowest level the firewall filter provides its primary service, that is *packet filtering*. A packet filter is a set of rules describing which packets are allowed to pass through the firewall and which are not. In essence, this allows an administrator to describe which hosts and protocols are accessible from a protected network, and thus the firewall can be used as a centralized control mechanism for managing access to hosts on a protected network.

By definition, the packet filter maintains no memory over a sequence of related packets. As a consequence packet filters cannot apply filtering rules based on previous events in the packet stream. To access this problem modern firewalls have been extended with a higher order filter referred to as *stateful inspection*.

Stateful inspection, as implemented on Linux 2.4, provides a filtering mechanism that operates on network connections rather than at the level of individual packets. This allows an administrator to refine the filtering policy further, because stateful inspection remembers the state of connections. For instance, the filter can now specify that connections may only be initiated from inside the protected network and out. As a consequence the network becomes more secure due to more accurate filter description and the overall filtering rules are simplified due to the *connection* abstraction provided with stateful inspection.

Both packet filters and stateful inspection filters does only classify based on the informa-

[3] mainly based on an analysis of the Netfilter firewall in the Linux 2.4 kernel [3] and ipfilter [5] for FreeBSD and OpenBSD

tion available in the packet headers, and thus the payload of the packets is ignored. Filtering based on payload is generally done in *application level filters*. A wide range of application level filters exists for filtering based on mail, web or other content. The main strength of these filters is protection against viruses and worms that spread through holes in mail readers and Web browsers. Furthermore, these filters are also often used for monitoring and controlling network usage from inside the protected network and out. Note that, as this filter need to reassemble the packets payloads, they are taking place in the top of the information based hierarchy previously defined.

Hence, we have introduced here the three main existing filters in modern firewalls. However, we should also mention that firewalls generally include other functionality such as *network address translation* (*aka* NAT), handling of encrypted connection, caching of Web data and so forth. Anyway, as our main goal is to strengthen the basis of firewalls, we focus on stateful inspection as well as packet filtering aspects. In the following we describe the two projects that we have initiated.

Improving Stateful Inspection

The principle idea of stateful inspection is to track how a stream of related packets passes through different protocol states such as initial handshakes for connection establishment. Using this technique a firewall can filter out any stream of packet that deviates from the protocol specification.

Through our study of the Linux 2.4.x implementation of stateful inspection we have observed that the implementation uses a simplified model of TCP. The use of a simplified connection model means that connection tracking will allow a wider range of protocol deviations than stateful inspection. Thus security improvements with connection tracking can be improved by using a more exact model of TCP connections.

An other problem in Linux 2.4.x implementation of stateful inspection is related to the way the kernel memorizes to active connections. Connections are stored in a table along with information describing the current state of the connection. The only way a connection can be removed from the connection table is on timeout. However, since this timeout period is close to two minutes the table can easily be filled, thus causing any new connections to be blocked. Consequently stateful inspection in Linux 2.4.x is very sensitive to overload or denial-of-service attacks where someone intentionally monopolize the connection table.

The goal of this project is first of all to improve the current implementation of stateful inspection module in Linux 2.4.x. The main focus with this project is to improve the protocol model used so that it matches that of other stateful implementations such as IP filter [5], and secondly, develop algorithms that improves the administration of the connection table, making the firewall less sensitive to high loads or denial-of-service attacks. In the long run the ambition of this project is to provide a generalized framework for stateful inspection. That is, a framework where new stateful protocols can be adapted though the use of an abstract protocol description.

High Performance Packet Classification

In this project our ambition is to provide both a well defined language description for filtering and a data structure that minimize the computational cost of packet filtering. To describe these goals we first need to look at how packet are specified and how they are currently classified.

Current approach

The actual filter is a disjunction of rules. Each rule describes a policy that should be applied on a specific set of packets. The actual packet set is described by specifying possible values of packet header fields such as addresses and port numbers. An example of such a filter is shown in Figure 22. Here the filter consists of three rules. Rule number one specifies that packet with a

Rule	Filter	Policy
1	-s 192.168.*.* –dport 22	accept
2	–dport 80	accept
3	*	deny

Figure 22: A packet filter.

source address beginning with 192.168 *and* with the destination port of 22 should be accepted by the filter. The second rule describes that any traffic with destination port 80 should be accepted. And finally, the third rule specifies that all non matching traffic should be denied access.

The sequence of the rules in a filter plays a significant role for what is specified. The reason for this is that when a packet is processed, the header is compared against the rules in the filter in the sequence that they are stated, and when a rule is matched then the policy of that rule is applied. A key reason for this philosophy is to allow the administrator to consider the optimal order of rules, e.g. most popular rules first, in order to minimize classification time of a packet.

This approach for describing firewall packet filters has several problems. Overall the current approach has linear complexity in the number of specified rules. So, as the number of services provided though a firewall increases, so does the number of rules. This fact along with the fact that we are seeing exponential growth in available bandwidth shows that current practice does not scale well enough to hold for future requirements.

Also, the language used for describing packet filters is poorly designed. Filters can be ambiguous. For instance, if the filter of rules intersect and their policy differ, then the order of the rules defines semantics of the filter. There is no standard syntax for firewall rules, and the expressive power of the language is poor. Moreover, this language is defined at a very low abstraction level and forces the user to be very explicit. Therefore, this language makes it difficult to express even simple formulas [2].

As a final note we should mention that we have seen no tools for developing packet filters, that is, no simulator, no debugger, and no checker. The reason is likely to be that the rule based language design is a weak platform to base such tools.

Using DDs

The packet classification problem can easily be reduced to the evaluation of a logic formula on bounded integer variables (the set of rules) given a set of variables (the header). This problem is known to be constant time with the number of variables. In order to solve this problem in the most efficient way we are using decision diagrams such as Binary Decision Diagrams (BDDs) [1] or Interval Decision Diagrams (IDDs) [4]. We have chosen to focus on IDDs because comparing integers rather than bits on a generic CPU is more efficient.

Introducing these structures requires that we more formally define the system we are working on. Let H be the set of all the possible headers and Π the set of all the policies. A rule is then a couple composed by a set of headers and a policy:

$$r=(\varphi,\pi), \text{ where } \varphi\in H \text{ and } \pi\in\Pi. \qquad (1)$$

A filter is described by a set of rules that defines a partition over $H\times\Pi$. The partition ensure consistent data structures as it do not allow intersection between any set of headers:

$$\begin{aligned}&F=((\varphi_1,\pi_1),(\varphi_2,\pi_2),\ldots,(\varphi_n,\pi_n)),\\&\text{where } \textstyle\bigcup_{i\le n}\varphi_i=H,\\&\text{and } \varphi_i\cap\varphi_j=\emptyset, \forall i,j \text{ with } i\neq j.\end{aligned}$$

IDDs or BDDs are used as an representation of any filter F. In our case each node in the tree represents a variable from a header, and a leaf represents a policy. The policy to apply on a packet is found by matching the nodes of the IDD with the header fields of the packet thus leading to a policy. As an example Figure 23 shows the filter specified in Figure 22. Notice that because the edges from a node defines a partition of a header field any packet in H can be matched with either

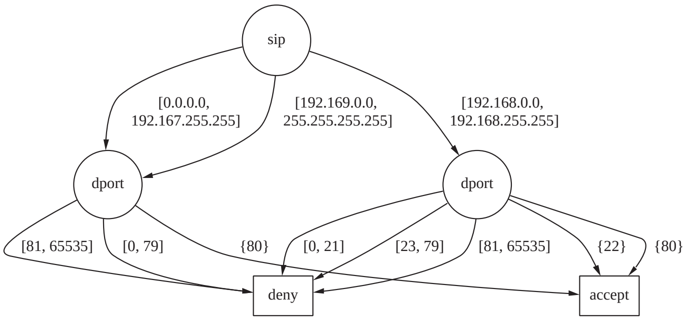


Figure 23: The IDD representing the filter described in figure 22

`accept` or `deny`, this is important since this ensures that a policy exists for any packet that is classified using instance of F.

An interesting property of BDDs and IDDs is that we can minimize the number of nodes of these structures by removing redundant or superfluous nodes. This corresponds to minimizing the number of tests that we have to perform in order to evaluate the formula, and thus, minimizing the number of tests necessary to determine which policy to apply to a given packet.

Conclusion

In this brief overview we have described our current work to improve Internet firewalls.

With the project on stateful inspection we aim at improving the methods used for tracking protocol failures, with the long term goal of providing a framework which abstract the protocol from the implementation of the stateful inspection module in the firewall.

The high performance packet filter project aims at defining and formalizing a framework for optimum packet filtering. Through the use of decision diagram structures we will provide both a compact and well defined language, based on logic, and an efficient structure in matter of space and computational time to represent the filters.

In a long term view, we are planning to provide tools based on model-checking techniques, that can help system administrators for checking and improving the security of their networks.

References

[1] Randal E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, August 1986.

[2] Brent D. Chapman. Network (in)security through ip packet filtering. In *Proceedings of the Third USENIX Security Symposium*, 1992.

[3] The Netfilter Project Homepage. `netfilter.samba.org`

[4] Karsten Strehl and Lothar Thiele. Symbolic model checking using interval diagram techniques. Technical Report 40, Computer Engineering and Networks Lab Swiss Federal Institute of Technology, Gloriastrasse 35, 8092 Zurich, Switzerland, 1998.

[5] Guido van Rooij. Real stateful tcp packet filtering in ip filter. In *proceedings of 10th USENIX Security Symposium*, 2001.

Calendar of Events

Date	Event
Feb '02	Mini-course on Mobile Calculi, Aarhus
Mar '02	Mini-course on Computable Analysis and its Applications, Aarhus
Apr '02	Mini-course on Automata and Logic, Aarhus
Jun 27–Jul 1 '02	EFF Summer School on Massive Data Sets, Aarhus
Jul 20 – Aug 1 '02	FloC, Federated Logic Conference, Copenhagen
Summer '03	18th IEEE Conference on Computational Complexity, Aarhus

BRICS Address and World Wide Web

For further information, hard copies of information material and reports of the BRICS Series as well as enrolment into the BRICS newsgroup, please contact **BRICS** at

Telephone:	+45 8942 3360
Telefax:	+45 8942 3255
Internet:	BRICS@brics.dk

or, in writing, to

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK - 8000 Aarhus C
Denmark.

You can get access to information from BRICS through World Wide Web (WWW) and anonymous FTP. To connect to the BRICS WWW entry, open the URL:

```
www.brics.dk
```

The BRICS WWW entry contains updated information about most of the topics covered in the newsletter as well as access to electronic copies of information material and reports of the BRICS Series (look under Publications).

To access the information material and reports of the BRICS Series via anonymous FTP do the following:

```
ftp ftp.brics.dk
get README.
```

BRICS *Newsletter*

ISSN 0909-6043

Editors: Mogens Nielsen & Uffe H. Engberg
Lay-out: Uffe H. Engberg
Publisher: BRICS

Print: Faculty of Science
University of Aarhus