

# BRICS *Newsletter*

Basic Research in Computer Science

No 5, July 1996

## In this Issue

<b>Welcome</b>	<b>1</b>
<b>Coming Events</b>	<b>1</b>
Set Constraints . . . . .	1
Competitive Online Algorithms . . . . .	2
Non-Interleaving Transition Systems . . . . .	2
Explicit Substitution . . . . .	2
Distributed Logics . . . . .	3
Verification '96 . . . . .	4
Autumn School . . . . .	4
<b>Reports on Events</b>	<b>4</b>
<b>Newly Appointed Researchers and Guests</b>	<b>5</b>
<b>Dissertation Abstracts</b>	<b>7</b>
Efficient External-Memory Data Structures and Applications . . . . .	7
Reasoning About Concurrent Computational Systems . . . . .	9
<b>New Reports</b>	<b>10</b>
<b>News and Technical Contributions</b>	<b>11</b>
Great(er) Expectations . . . . .	11
Some Surprising Connections between Theoretical Programming Languages and Algorithms . . . . .	14
MONA and FIDO . . . . .	14
<b>Calendar of Events</b>	<b>16</b>
<b>BRICS Address and World Wide Web</b>	<b>16</b>

## Welcome

Welcome to the fifth issue of the BRICS newsletter. Its purpose is to inform you of appointments,

publications, courses and other activities within BRICS. Further details can be obtained by contacting the addresses on the back page. 

## Coming Events

For details see the BRICS Activities web page<sup>1</sup>.

## Set Constraints

In mid August *Dexter Kozen*, Joseph Newton Pew, Professor of Computer Science, Cornell University, Ithaca, NY, USA, will give a short course of six lectures on Set Constraints. Set constraints are inclusions between expressions denoting sets of ground terms. They have been used extensively in program analysis and type inference. The course will give an introduction to the theory and applications of set constraints. The lectures are:

1. Basic definitions and applications.
2. Tree set automata and hypergraphs.
3. Complexity.
4. Duality of “soft” and “hard” typing.
5. Rational spaces.
6. Constraint logic programming with set constraints. 

<sup>1</sup><http://www.brics.dk/Activities>

## Competitive Online Algorithms

At the end of August *Susanne Albers*, Max-Planck-Institut für Informatik, Saarbrücken, Germany will give a mini-course on Competitive Online Algorithms.

Online algorithms represent a relatively new area of research that has developed in the past ten years. An online algorithm receives the input incrementally, one piece at a time, and must process each input portion without knowing future portions. An online algorithm  $A$  is called  $c$ -competitive if, for every input sequence, the cost incurred by  $A$  is at most  $c$  times the optimal off line cost for that sequence. In this mini-course we will present the most important concepts and techniques used in the study of deterministic and randomized online algorithms.

Research on online algorithms is generally motivated by real online problems that arise in practice. In the mini-course we will discuss in detail algorithms for the following applications.

1. Paging.
2. Self-organizing data structures; we will also show how algorithms in this field can be used to construct data compression schemes.
3. Scheduling.
4. Navigation and exploration.
5. Distributed computing. ☰

## Non-Interleaving Transition Systems

In September *Vladimiro Sassone*, Dipartimento di Informatica, Pisa, Italy, will give a mini-course on Non-Interleaving Transition Systems.

The operational semantics of concurrent systems is often specified by means of labelled transition systems. These simple structures provide a level of abstraction which is appropriate for many applications, and allow us to take an “extensional” view by focusing on the labels, meant

to represent the system's observable behaviour. They are, however, so-called *interleaving* models, meaning that they fail to draw natural distinctions between interleaved and concurrent execution of actions, so losing track of the casual dependencies between them.

Many authors have recently argued in favour of generalising transition systems, aiming at transition-based models in which the concurrent activity of several agents is explicitly represented by “*higher dimensional*” transitions. Besides any consideration about the semantic relevance of cause/effect relationships, such non-interleaving models can be helpful in more pragmatic tasks, such as model checking, where they can reduce considerably the state-space explosion problem.

We focus on a very recent model—actually still under development—which provides an elementary, set-theoretic formalisation of the idea of higher dimensional transition: the *Higher Dimensional Transition Systems* (HDTS). First, we review some of the proposals in the literature and compare them to HDTS, trying to show that they are rather natural structures. Then, we develop an extensional semantic theory for HDTS centered on a notion of history-preserving bisimulation that can be formulated abstractly in the “bisimulation-via-open-maps” paradigm. We conclude by presenting some of the latest developments, as well as open problems and directions for further research, aiming at demonstrating that HDTS are interesting mathematical structures, besides being a rather general model of concurrency. ☰

## Explicit Substitution

On October 10 and 11, *Kristoffer Høgsbro Rose*, BRICS, will a three lectures mini-course on “Explicit Substitution.” Topics:

1. Explicit Substitution Rules.

The basics of making the lambda-calculus \*syntactic\* by encoding substitution and de Bruijn indices and the rôle of substitution

concatenation and composition for proving confluence [1]. Why lambda-sigma-substitution is not terminating [2].

## 2. Explicit Substitution as a Basis for Implementations.

We explain how explicit substitution provides a technique for \*separating\* the issues of primitive operations and reduction strategies and show how an explicit substitution calculus equipped with a strategy can be used to \*derive\* an abstract machine. We show how this can be enriched to give formally understandable justifications for the techniques used for \*sharing\*, \*recursion\*, and \*parallelism\*, in “real” implementations [3].

## 3. Explicit Substitution in Higher-Order Rewriting.

We explain how explicit substitution can be used to facilitate reasoning about higher-order rewriting, both by translating higher-order rewrite systems into explicit substitution as well as defining higher-order rewriting through explicit substitution [4].

It is intended to collect a selection of extracts from Rose's thesis and papers, probably around 30 pp combining the essential results in explicit substitution with a bibliography; This will appear in the BRICS Notes Series.

## References

- [1] M. Abadi, L. Cardelli, P.-L. Curien, and J.-J. Lévy, Explicit substitutions, *Jour. Functional Progr.*, 1(4): 375–416, 1991.
- [2] P.-A. Mellis, Typed lambda-calculi with explicit substitution may not terminate, in M. Dezani, editor, *Int. Conf. on Typed Lambda Calculus and Applications*, Edinburgh, Scotland, LNCS 902, pp. 328–334, 1995
- [3] Z.-E.-A. Benaissa, K. H. Rose, and P. Lescanne, Modeling sharing and recursion for

weak reduction strategies using explicit substitution, in H. Kuchen and D. Swierstra, editors, *8th PLILP—Symposium on Programming Language Implementation and Logic Programming*, Aachen, Germany, September 1996.

- [4] R. Bloo and K. H. Rose, Combinatory reduction systems with explicit substitution that preserve strong normalisation, in H. Ganzinger, editor, *RTA '96*, Rutgers University, New Jersey, July 1996. ☐

## Distributed Logics

In November *P. S. Thiagarajan*, School of Mathematics, SPIC Science Foundation, Madras, India, will give a course of lectures (6–8 hours) on Distributed Logics.

A common feature of these logics is that their syntax and semantics—unlike in the case of conventional temporal logics—will directly reflect the fact that they are being interpreted over distributed behaviours.

Apart from an overview which will emphasize the motivation for considering such logics, the lectures will consider representative members of the following families of logics.

1. Dynamic logics for distributed transition systems.
2. Modal logics for prime event structures.
3. Linear time temporal logics for Mazurkiewicz traces.
4. Branching time temporal logics for trace structures (i.e. the branching time counterparts of Mazurkiewicz traces)

The logics in 4. are the subject of current research. ☐

## Verification '96

This year's BRICS theme is Verification. The activities will cover verification of computing systems in a broad sense. More specifically, we intend to investigate specification formalisms, proof principles, and technology of automated tools for verification. Events will take place during the autumn of '96. See e.g. the Autumn School below.

Guests of the theme include David A. Basin, Tom Melham, Nils Klarlund, Randy Pollack, Prakash Panangaden and P. S. Thiagarajan. ☰

## Autumn School

In connection with this year's BRICS theme on Verification, a one week autumn school will be held starting October 28 covering "Theorem Proving and Model Checking". The contents of the autumn school will be the presentation of a selected set of tools principles (theorem provers, model checkers and combinations) focussing on "Cutting-Edge Applications-Oriented Techniques", see the 4th issue of the BRICS Newsletter.

The deadline for application was June 15, and we have received about 100 applications.

The lectures to be given by the invited speakers are:

- Verification Based on Monadic Logic by *David A. Basin*, Max-Planck-Institut für Informatik.

## Reports on Events

### Course on Distributed Algorithms

*Richard B. Tan*, Department of Computer Science, University of Sciences & Arts of Oklahoma and University of Utrecht, visited BRICS during May and June and gave a mini-course in the area of

- Symbolic Model Checking by *Ed Clarke*, School of Computer Science, Carnegie Mellon University.
- Automatic Verification of Real-Time and Hybrid Systems by *Thomas A. Henzinger*, University of California at Berkeley.
- On-the-Fly Model Checking Tutorial by *Gerard Holzmann*, Computing Principles Research, Bell Laboratories, USA.
- Some Research Issues in Higher Order Logic Theorem Proving by *Tom Melham*, Department of Computing Science, University of Glasgow.
- What we learn from formal checking by *Randy Pollack*, Chalmers.
- A Combined Approach to Hardware Verification: Proof-checking, Rewriting with decision procedures and Model-checking by *Mandayam Srivas*, SRI International.

Detailed abstracts are accessible through the [www-url Verification theme Web page](http://www.urlverification.dk)<sup>2</sup>.

We plan on making the lecture notes available in the BRICS lecture notes series.

David A. Basin, Allan Cheng, Kim G. Larsen, Tom Melham, and Mogens Nielsen are responsible for the planning. Everybody potentially interested in taking part in the activities is welcome to contact us directly at [BRICS@brics.dk](mailto:BRICS@brics.dk). ☰

Distributed Algorithms. The course was organized into four basically self-contained lectures, each lasting approximately two hours:

1. Introduction and Models. Election on a Ring: Asynchronous algorithms, Syn-

<sup>2</sup><http://www.brics.dk/Activities/96/Verification>

chronous algorithms and Lower Bound results.

2. Election on complete graphs. Minimum Spanning Trees on general graphs.
3. Global Algorithms: Termination Detection and Snapshots.
4. Fault-Tolerance: Benign Failures, Byzantine Failures and Self-stabilization.

A list of references can be found on the Web<sup>3</sup>.

The course was attended by some 25 researchers and students from Aarhus. ☐

## BRICS Strategy Workshop

From mid-day Wednesday 5 June till early afternoon Thursday 6 June BRICS had a so called (in Danish) “internat” meeting—a BRICS retreat to discuss BRICS things technical, pedagogical and social, ranging from how things are going to how we might improve them in the future. The programme started with overviews of research achievements, and plans and hopes for the future. With an eye to broadening our concerns Richard B. Tan gave a talk on Distributed Algorithms and the relationship to Algorithmics, Mathematical Logic, and Semantics, while Christian N. S. Pedersen gave an introduction to basic concepts and fundamental questions in the area of Computational Biology. A lot of room was given to discussion. Figure 1 shows the participants of the workshop. ☐

## Newly Appointed Researchers and Guests

### *Klaus Havelund*

Klaus Havelund, Paris 6 University, France, will the start of September 1996 visit BRICS for one year. Klaus got his Ph.D. from DIKU, University of Copenhagen, in 1994. The thesis was entitled: “The Fork Calculus – Towards a Logic for Concurrent ML” and was supervised by Klaus Grue. Most of the work was carried out during a stay at Ecole Normale Supérieure, Paris, but regular visits were made to Aalborg University, where he collaborated with Kim Guldstrand Larsen. Current interests include the combination of theorem proving (PVS) and model checking to the verification of parallel and distributed algorithms.

### *Kristoffer Høgsbro Rose*

Kristoffer has joined us for the rest of this year to do research as well as teach a course in *rewriting systems*. He obtained his Ph.D. degree from DIKU, Computer Science, in February, doing research into how rewrite systems in general, and *explicit substitution* in particular, can be

used to model “difficult” aspects of real functional programming language evaluation such as *sharing* and *recursion*. From March to May he visited the rewriting group of INRIA Lorraine, Nancy (France), a collaboration that is continuing. Kristoffer's (scientific) hobby is the pragmatics of *diagram drawing* and his  $\TeX$  package  $\mathcal{X}\text{-pic}$  may be known to some.

### *Anna Ingólfssdóttir*

Anna Ingólfssdóttir's main research interest is within the area of semantic models for concurrency. She obtained her PhD in 1994 from Sussex University, supervised by Prof. Matthew Hennessy. She was an assistant professor at Aalborg University from February 1991 to December 1995 and joined BRICS in January 1996.

— ☐ —

### *Marcin Jurdziński*

Marcin Jurdziński is a student of Warsaw University. He is at the time working on his MSc thesis on the complexity of model checking for the *mu*-calculus under guidance of Dr. Damian

<sup>3</sup><http://www.brics.dk/Activities/96/DistAlg.html>



Figure 1: Participants in the BRICS Strategy Workshop. From left, row 1 (squatting): *Kim G. Larsen, Torben Braüner, Olivier Danvy, Uffe H. Engberg, Allan Cheng, Robert*; 2: *Arne Skou, Kristoffer H. Rose, Karen K. Møller, Martin Musicante*; 3: *Theis Rauhe, Thore Husfeldt, Glynn Winskel, Sven Skyum, Josva Kleist, Luca Aceto*; 4: *Ole I. Hougaard, Thomas T. Hildebrandt, Igor Walukiewicz, Andreas Podelski, Gerth S. Brodal, Mogens Nielsen, Anna Ingólfssdóttir*; 5: *David A. Basin, Jørgen H. Andersen, Ian Stark, Gudmund S. Frandsen, Christian N. S. Pedersen, Rune B. Lyngsø.*

Niwiński. His interests include verification of finite state concurrent systems, complexity theory and finite model theory. During his stay he will work on a problem of synthesis of parallel programs from games and hierarchy problems in fixpoint logics. He will be at BRICS in July as a BRICS summer student and then he will also stay for the autumn semester on the Polish-Danish exchange grant.

#### *Stefan Dziembowski*

Stefan Dziembowski is a student of the Warsaw University and from the next year will start to work as a teaching assistant there. He expects to get his MSc in Computer Science in August 1996 (with a dissertation titled "On the Complexity of the Fixpoint Database Queries"). His fields

of interest include theory of automata on infinite objects,  $\mu$ -calculus, monadic second-order logic and the functional programming. During his stay he will work on complexity issues of evaluating fixpoint queries. He will be visiting BRICS from the start of July to primo August as a BRICS summer student.

#### *Abdulwaheb Ayari*

Abdulwaheb Ayari is a PhD student of David A. Basin from Max-Planck-Institut für Informatik, Saarbrücken, Germany. He will visit BRICS in July and August and work on high-level notations for monadic second-order logic and related systems and their efficient translation into the MONA system. ☐

# Dissertation Abstracts

## Efficient External-Memory Data Structures and Applications

by Lars Arge

Traditionally when designing computer programs people have focused on the minimization of the internal computation time and ignored the time spent on Input/Output (I/O). Theoretically one of the most commonly used machine models when designing algorithms is the Random Access Machine (RAM) and one main feature of the RAM is that its memory consists of an (infinite) array, and that any entry in the array can be accessed at the same (constant) cost. Also in practice most programmers conceptually write programs on a machine model like the RAM. In an UNIX environment for example the programmer thinks of the machine as consisting of a processor and a huge (“infinite”) memory where the contents of each memory cell can be accessed at the same cost. The task of moving data in and out of the limited main memory is then entrusted to the operating system.

However, in practice there is a huge difference in access time of fast internal memory and slower external memory such as disks. While typical access time of main memory is measured in nano seconds, a typical access time of a disk is on the order of milli seconds. So roughly speaking there is a factor of a million in difference in the access time of internal and external memory, and therefore the assumption that every memory cell can be accessed at the same cost is questionable, to say the least!

In many modern large-scale applications the communication between internal and external memory, and not the internal computation time, is actually the bottleneck in the computation. Examples of large-scale applications can e.g. be found in database systems, spatial databases and geographic information systems (GIS), VLSI verification, constraint logic programming, computer graphics and virtual reality systems, computational biology, physics and geophysics and

in meteorology. The amount of data manipulated in such applications is too large to fit in main memory and must reside on disk, hence the I/O communication can become a very severe bottleneck. A good example is NASA's EOS project GIS system, which is expected to manipulate petabytes (thousands of terabytes, or millions of gigabytes) of data! The effect of the I/O bottleneck is getting more pronounced as internal computation gets faster, and especially as parallel computing gains popularity. Currently, technological advances are increasing CPU speeds at an annual rate of 40–60% while disk transfer rates are only increasing by 7–10% annually. Internal memory sizes are also increasing, but not nearly fast enough to meet the needs of important large-scale applications.

Modern operating systems try to minimize the effect of the I/O bottleneck by using sophisticated paging and prefetching strategies in order to assure that data is present in internal memory when it is accessed. However, these strategies are general purpose in nature and therefore they cannot take full advantage of the properties of a specific problem. Instead one could hope to design more efficient algorithms by explicitly considering the I/O communication when designing algorithms for specific problems. Such algorithms designed with I/O in mind are often called *external memory (or I/O) algorithms*.

## Contributions

In this thesis we study problems from several of the above mentioned areas in the *parallel disk model* defined by Aggarwal & Vitter and Vitter & Shriver (see Figure 2) and design efficient external-memory algorithms for them. In the parallel disk model the internal memory is capable of holding  $M$  data elements and the complexity measure is the number of I/Os used to solve a given problem. An I/O is defined as the process of simultaneously reading or writing a block of  $B$  contiguous data elements to or from

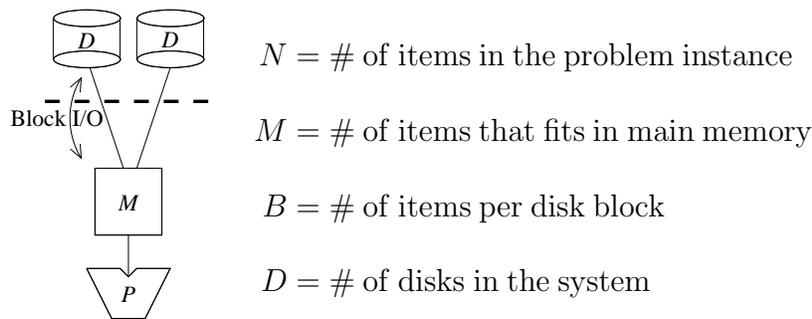


Figure 2: The parallel disk model

each of  $D$  disks. Internal computation is free in the model as I/O is our main concern.

A general theme in the thesis is to design I/O-efficient algorithms through the design of I/O-efficient data structures. One of our philosophies is to try to isolate all the I/O specific parts of an algorithm in the data structures, that is, to try to design I/O algorithms from internal memory algorithms by exchanging the data structures used in internal memory with their external memory counterparts. The results in the thesis include a technique for transforming an internal memory tree data structure into an external data structure which can be used in a *batched* dynamic setting, that is, a setting where we for example do not require that the result of a search operation is returned immediately. Using this technique we develop batched dynamic external versions of the (one-dimensional) range-tree and the segment-tree and we develop an external priority queue. Following our general philosophy we show how these structures can be used in standard internal memory sorting algorithms and algorithms for problems involving geometric objects. The latter have applications to VLSI design. Using the priority queue we improve upon known I/O algorithms for fundamental graph problems, and develop new efficient algorithms for the graph-related problem of ordered binary-decision diagram manipulation. Ordered binary-decision diagrams are the state-of-the-art data structure for boolean function manipulation and they are extensively used in large-scale applications like logic circuit verification.

Combining the batched dynamic segment tree with the novel technique of external-memory fractional cascading we develop I/O-efficient algorithms for a large number of geometric problems involving line segments in the plane, with applications to geographic information systems. Such systems frequently handle huge amounts of spatial data and thus they require good use of external-memory techniques.

We also manage to use the ideas in the batched dynamic segment tree to develop “on-line” external data structures for a special case of two-dimensional range searching with applications to databases and constraint logic programming. We develop an on-line external version of the segment tree, which improves upon the previously best known such structure, and an optimal on-line external version of the interval tree. The last result settles an important open problem in databases and I/O algorithms. In order to develop these structure we use a novel balancing technique for search trees which can be regarded as weight-balancing of B-trees.

Finally, we develop a technique for transforming internal memory lower bounds to lower bounds in the I/O model, and we prove that our new ordered binary-decision diagram manipulation algorithms are asymptotically optimal among a class of algorithms that include all know manipulation algorithms. ☐

# Reasoning About Concurrent Computational Systems

by Allan Cheng

This thesis contains three parts. The first part presents contributions in the field of verification of finite state concurrent systems, the second part presents contributions in the field of behavioural reasoning about concurrent systems based on the notions of behavioural preorders and behavioural equivalences, and, finally, the third part presents contributions in the field of set constraints. The main results are described below.

In the first part, we start by studying the computational complexity of several standard verification problems for a 1-safe Petri nets and some of its subclasses. Our results provide the first systematic study of the computational complexity of these problems for 1-safe nets.

We then investigate the computational complexity of a more general verification problem, model-checking, when an instance of the problem consists of a formula and a description of a system whose state space is at most exponentially larger than the description.

We continue by considering the problem of performing model-checking relative to a partial order semantics of concurrent systems, in which not all possible sequences of actions are considered relevant. It turns out that Mazurkiewicz trace theory provides a natural partial order semantics, in which the progress fairness assumptions can be formalised. We provide the first, to the best of our knowledge, set of sound and complete tableau rules for a CTL-like logic interpreted under progress fairness assumptions.

In the second part, we start by investigating Joyal, Nielsen, and Winskel's proposal of spans of open maps as an abstract category-theoretic way of adjoining a bisimulation equivalence,  $\mathcal{P}$ -bisimilarity, to a category of models of computation  $\mathcal{M}$ . We show that a representative selection of well-known bisimulations and be-

havioural equivalences can be captured in the setting of spans of open maps. We conclude, Joyal, Nielsen, and Winskel's proposed notion of open maps seems successful.

An issue left open by Joyal, Nielsen, and Winskel's work on open maps was the congruence properties of behavioural equivalences. We address the following fundamental question: given a category of models of computation  $\mathcal{M}$  and a category of observations  $\mathcal{P}$ , are there any conditions under which algebraic constructs viewed as functors preserve  $\mathcal{P}$ -bisimilarity? We define the notion of functors being  $\mathcal{P}$ -factorisable and show how this ensures that  $\mathcal{P}$ -bisimilarity is a congruence with respect to such functors. Guided by the definition of  $\mathcal{P}$ -factorisability we show how it is possible to parametrise proofs of functors being  $\mathcal{P}$ -factorisable with respect to the category of observations  $\mathcal{P}$ , i.e., with respect to a behavioural equivalence.

In the last part we then, almost, leave the field of concurrency to investigate set constraints.

Set constraints are inclusion relations between expressions denoting sets of ground terms over a ranked alphabet. They are the main ingredient in set-based program analysis. They are typically derived from the syntax of a program and solutions to them can yield useful information for, e.g., type inference, implementations, and optimisations.

In a brief section, we sketch how Milner's protocol can be verified for the absence of deadlocks using Kozen's CLP(SC), a constraint logic programming language over set constraints.

We provide a complete Gentzen-style axiomatisation for sequents  $\Phi \vdash \Psi$ , where  $\Phi$  and  $\Psi$  are finite sets of set constraints, based on the axioms of termset algebra. Sequents of the restricted form  $\Phi \vdash \perp$  correspond to positive set constraints, and those of the more general form  $\Phi \vdash \Psi$  correspond to systems of mixed positive and negative set constraints. We show that the deductive system is complete for the restricted sequents  $\Phi \vdash \perp$  over standard models, incomplete for general sequents  $\Phi \vdash \Psi$  over standard

models, but complete for general sequents over set-theoretic termset algebras.

We then continue by investigating Kozen's rational spaces. Rational spaces are topological spaces obtained as spaces of runs of topological  $\Sigma$ -hypergraphs. They were introduced by Kozen who showed how the topological structure of the spaces of solutions to systems of set constraints can be given in terms of rational

spaces. We give a Myhill-Nerode-like characterisation of rational points, which in turn is used to re-derive results about the rational points of finitary rational spaces. We show that the rational points in finitary rational spaces in some sense exactly capture the topological structure of the space. We define and investigate congruences on  $\Sigma$ -hypergraphs, and finally we determine the computational complexity of some decision problems related to rational spaces.  $\equiv$

## New in the BRICS Report Series

ISSN 0909-0878

- 24** Henrik Ejersbo Jensen, Kim G. Larsen, and Arne Skou. *Modelling and Analysis of a Collision Avoidance Protocol using SPIN and UP-PAAL*. July 1996. 20 pp.
- 23** Luca Aceto, Wan J. Fokkink, and Anna Ingólfssdóttir. *A Menagerie of Non-Finitely Based Process Semantics over BPA\*—From Ready Simulation Semantics to Completed Trace Semantics*. June 1996. 37 pp.
- 22** Luca Aceto and Wan J. Fokkink. *An Equational Axiomatization for Multi-Exit Iteration*. June 1996. 30 pp.
- 21** Dany Breslauer, Tao Jiang, and Zhigen Jiang. *Rotation of Periodic Strings and Short Superstrings*. June 1996. 14 pp.
- 20** Olivier Danvy and Julia L. Lawall. *Back to Direct Style II: First-Class Continuations*. June 1996. 36 pp. A preliminary version of this paper appeared in the proceedings of the 1992 ACM Conference on Lisp and Functional Programming, William Clinger, editor, LISP Pointers, Vol. V, No. 1, pages 299–310, San Francisco, California, June 1992. ACM Press.
- 19** John Hatcliff and Olivier Danvy. *Thunks and the  $\lambda$ -Calculus*. June 1996. 22 pp. To appear in *Journal of Functional Programming*.
- 18** Thomas Troels Hildebrandt and Vladimiro Sassone. *Comparing Transition Systems with Independence and Asynchronous Transition Systems*. June 1996. 14 pp. To appear in *Concurrency Theory: 7th International Conference, CONCUR '96 Proceedings, LNCS, 1996*.
- 17** Olivier Danvy, Karoline Malmkjær, and Jens Palsberg. *Eta-Expansion Does The Trick (Revised Version)*. May 1996. 30 pp. To appear in *ACM Transactions on Programming Languages and Systems (TOPLAS)*.
- 16** Lisbeth Fajstrup and Martin Raußen. *Detecting Deadlocks in Concurrent Systems*. May 1996. 10 pp.
- 15** Olivier Danvy. *Pragmatic Aspects of Type-Directed Partial Evaluation*. May 1996. 27 pp.
- 14** Olivier Danvy and Karoline Malmkjær. *On the Idempotence of the CPS Transformation*. May 1996. 15 pp.
- 13** Olivier Danvy and René Vestergaard. *Semantics-Based Compiling: A Case Study in Type-Directed Partial Evaluation*. May 1996. 28 pp. To appear in *8th International Symposium on Programming Languages, Implementations, Logics, and Programs, PLILP '96 Proceedings, LNCS, 1996*.

- 12 Lars Arge, Darren E. Vengroff, and Jeffrey S. Vitter. *External-Memory Algorithms for Processing Line Segments in Geographic Information Systems*. May 1996. 34 pp. A shorter version of this paper was presented at the *Third Annual European Symposium on Algorithms*, ESA '95.
- 11 Devdatt Dubhashi, David A. Grable, and Alessandro Panconesi. *Near-Optimal, Distributed Edge Colouring via the Nibble Method*. May 1996. 17 pp. Invited to be published in a special issue of *Theoretical Computer Science* devoted to the proceedings of ESA '95.
- 10 Torben Braüner and Valeria de Paiva. *Cut-Elimination for Full Intuitionistic Linear Logic*. April 1996. 27 pp. Also available as Technical Report 395, Computer Laboratory, University of Cambridge.
- 9 Thore Husfeldt, Theis Rauhe, and Søren Skyum. *Lower Bounds for Dynamic Transitive Closure, Planar Point Location, and Parentheses Matching*. April 1996. 11 pp. To appear in *Algorithm Theory: 5th Scandinavian Workshop, SWAT '96 Proceedings*, LNCS, 1996.
- 8 Martin Hansen, Hans Hüttel, and Josva Kleist. *Bisimulations for Asynchronous Mobile Processes*. April 1996. 18 pp. Appears in *Thilisi Symposium on Language, Logic, and Computation*, 1995.
- 7 Ivan Damgård and Ronald Cramer. *Linear Zero-Knowledge - A Note on Efficient Zero-Knowledge Proofs and Arguments*. April 1996. 17 pp.
- 6 Mayer Goldberg. *An Adequate Left-Associated Binary Numeral System in the  $\lambda$ -Calculus (Revised Version)*. March 1996. 19 pp. Accepted for *Information Processing Letters*. This report is a revision of the BRICS Report RS-95-38.
- 5 Mayer Goldberg. *Gödelisation in the  $\lambda$ -Calculus (Extended Version)*. March 1996. 10 pp.
- 4 Jørgen H. Andersen, Ed Harcourt, and K. V. S. Prasad. *A Machine Verified Distributed Sorting Algorithm*. February 1996. 21 pp. Abstract appeared in *7th Nordic Workshop on Programming Theory, NWPT '97 Proceedings*, 1995.

## News and Technical Contributions

### Great(er) Expectations

Devdatt Dubhashi<sup>4</sup> and Desh Ranjan<sup>5,6</sup>

Let  $(B, T, E)$  be a bipartite graph; by thinking of  $B$  as the “bottom” vertices, and  $T$  as the top vertices, and an edge  $(b, t)$  as standing for the relation  $b \leq t$ , we may regard it as a poset  $P$  of height 2. Let  $f$  be a real-valued function defined on  $P$  which is *non-decreasing* i.e.  $b \leq t$  implies  $f(b) \leq f(t)$ . Finally, let  $\mu$  be a measure defined

on  $P$ , i.e. a non-negative real valued function on  $P$  (not necessarily summing to 1).

It seemed attractive to us to conjecture that

$$\begin{aligned}
 E_B(f) &:= \frac{\sum_{b \in B} f(b)\mu(b)}{\sum_{b \in B} \mu(b)} \\
 &\leq \frac{\sum_{t \in T} f(t)\mu(t)}{\sum_{t \in T} \mu(t)} =: E_T(f).
 \end{aligned} \tag{1}$$

Intuitively, the expected value of  $f$  on the “top”

<sup>4</sup>BRICS, dubhashi@daimi.aau.dk.

<sup>5</sup>Work done while the author was visiting the Max-Planck-Institut für Informatik, and BRICS, University of Aarhus.

<sup>6</sup>Department of Computer Science, New Mexico State University, Las Cruces, New Mexico 88003, USA, dranjan@cs.nmsu.edu.

is greater than the expected value of  $f$  on the “bottom”.

One quickly discovers that (1) is false in general: take  $B := \{b_1, b_2\}, T := \{t_1, t_2\}$  with  $b_i \leq t_i$  for  $i = 1, 2$ . On  $B$ , set  $\mu(b_1) := 0, \mu(b_2) := 1$ , while on  $T$ , set  $\mu(t_1) = \frac{1}{2} = \mu(t_2)$ . Take  $f(x) := 1$  if  $x = b_2, t_2$  and 0 otherwise (note that this is a non-decreasing function). Then  $E_B(f) = 1 > \frac{1}{2} = E_T(B)$ .

Under what conditions on  $\mu$  does one obtain (1)? For  $b \in B$ , let  $N(b) := \{t \in T \mid b \leq t\}$  and for  $U \subseteq B$ , let  $N(U) := \bigcup_{b \in U} N(b)$  be the usual definition of neighbourhoods. For a subset  $S$ , set  $\mu(S) := \sum_{x \in S} \mu(x)$ .

**Theorem 1** *The following conditions are equivalent:*

- For any non-decreasing  $f$ , (1) holds.
- The subset property:

$$\mu(U) \leq \mu(N(U)), \quad \text{for any } U \subseteq B. \quad (2)$$

*Proof.* In one direction, take  $f$  to be the indicator function:

$$f(x) := \begin{cases} 1, & \text{if } x \in U \cup N(U); \\ 0, & \text{otherwise.} \end{cases}$$

Then

$$\mu(U) = E_B(f) \leq E_T(f) = \mu(N(U)).$$

Now let us show that the subset property yields (1). Number the elements of  $B$  so that  $f(b_1) \geq f(b_2) \geq \dots \geq f(b_m)$ . For  $i \in [m]$ , set  $S_i := N(\{b_1, \dots, b_i\}) \setminus N(\{b_1, \dots, b_{i-1}\})$ . Note that for each  $w \in S_i$ , we have  $f(w) \geq f(b_i)$ , hence we may as well assume that  $f(w) = f(b_i)$  for each  $w \in S_i$ . Then, we need to show that

$$\sum_{i \in [m]} f(b_i) \mu(b_i) \leq \sum_{i \in [m]} f(b_i) \mu(S_i).$$

Note that by the subset property, we have for each  $k \leq [m]$ ,

$$\sum_{i \in [k]} \mu(b_i) \leq \sum_{i \in [k]} \mu(S_i).$$

The result now follows from a standard majorisation argument quoted below. ■

**Lemma 2 (Majorisation)** *Let  $(a_i, i \in [m])$  and  $(b_i, i \in [m])$  be two sequences of reals such that for each  $k \leq m$ ,  $\sum_{i \in [k]} a_i \leq \sum_{i \in [k]} b_i$ . Then for any sequence  $f_1 \geq f_2 \geq \dots \geq f_m$ ,*

$$\sum_{i \in [m]} f_i a_i \leq \sum_{i \in [m]} f_i b_i.$$

We leave the proof of this lemma to the reader; see [3].

Let us consider now a rather special height 2 poset. Let  $m$  be a positive integer and let  $a < m$ . Consider the poset determined by the subsets of  $[m]$  of size  $a$  and  $a + 1$  ordered by set inclusion. Thus, the “bottom” elements are the subsets of size  $a$  and the “top” elements are the subsets of size  $a + 1$ . On subsets, a natural measure is the product measure: for a subset  $K \subseteq [m]$ ,

$$\mu(K) := \prod_{i \in K} p_i \prod_{i \notin K} q_i, \quad (3)$$

where  $(p_i, q_i \mid i \in [m])$  are arbitrary reals. Does (1) hold? In order for (1) to hold, Theorem 1 requires us to verify the subset condition for the product measure, a somewhat daunting task.

Instead we shall directly show that (1) holds. *The proof will not involve any calculations whatsoever!* The key will be the celebrated Marriage Theorem of Phillip Hall [2, Ch. 5]:

**Theorem 3 (Hall's Marriage Theorem)** *Let  $G := (A, B, E)$  be a bipartite graph. Then a necessary and sufficient condition for there to exist a matching in  $G$  saturating the vertices in  $A$  is that for all subsets  $U \subseteq A$ ,  $|N(U)| \geq |U|$ .*

What could Hall's Theorem possibly have to do with proving (1) for the case of product measures on subsets? A hint of the relationship is in the formal similarity of the condition in Hall's Theorem to the subset condition (2). Let us rewrite (1) as

$$\sum_K f(K) \sum_{K'} \mu(K') \leq \sum_{K'} f(K') \sum_K \mu(K),$$

or,

$$\sum_{K, K'} f(K)\mu(K)\mu(K') \leq \sum_{K, K'} f(K')\mu(K)\mu(K'). \quad (4)$$

Here  $K$  ranges over all subsets of size  $a$  and  $K'$  over all subsets of size  $a + 1$ . Think of  $(p_i, q_i, i \in [m])$  as independent indeterminates, and hence regard this an inequality over the polynomial ring  $N[p_i, q_i, i \in [m]]$ . Then, of course, it is natural to compare the two sides term-wise. Pick a fixed monomial  $t$ , and let

$$S_t := \{(K, K') \mid \mu(K)\mu(K') = t\},$$

be the set of pairs producing this monomial. Then, it suffices to prove that

$$\sum_{(K, K') \in S_t} f(K) \leq \sum_{(K, K') \in S_t} f(K'). \quad (5)$$

Let us take a closer look at the structure of the set  $S_t$ . Let  $(K, K')$  be a pair of sets producing the monomial  $t$ . Note that for each  $i \in [m]$ , the factor  $p_i^\alpha q_i^\beta$  occurs in  $t$  with exponent

- $\alpha = 2, \beta = 0$ , exactly if  $i$  is in both  $K$  and  $K'$ ;
- $\alpha = 1 = \beta$ , exactly if  $i$  is in one of  $K$  or  $K'$ .
- $\alpha = 0, \beta = 2$  exactly if  $i$  is in neither  $K$  nor  $K'$ .

Thus, the monomial  $t$  records exactly the multi-set  $K \cup K'$ . What other pairs of sets could produce the monomial  $t$ ? Exactly those that produce the same multiset as their multiset-union. Let  $U_t$  denote the multi-set  $K \cup K'$ ; note that this is of size  $2a + 1$  counting multiplicity. Let  $I_t$  denote the intersection  $K \cap K'$ . Then  $S_t$  consists exactly of the pairs  $(K, K')$  with  $K \cap K' = I_t$  and the remaining elements in  $U_t \setminus (I_t + I_t)$  partitioned in all possible ways into  $K$  and  $K'$  with exactly one more element in  $K'$ . Let  $U'_t$  denote the multi-set difference  $U_t \setminus (I_t + I_t)$ . Note that  $U'_t$  is a set of odd size. Note also that each  $K$  can be paired with exactly one  $K'$  and vice-versa to produce the monomial  $t$ .

Thus (5) reduces to showing:

$$\sum_{K \subseteq U'_t, |K|=a-|I_t|} f(K \cup I_t) \leq \sum_{K' \subseteq U'_t, |K'|=a-|I_t|+1} f(K' \cup I_t) \quad (6)$$

This follows from the following lemma with  $S := U'_t$  and  $g(K) := f(K \cup I_t)$ .

**Lemma 4** *Let  $S$  be a set of size  $2a + 1$  for a non-negative integer  $a$  let  $g$  be any real-valued function on sets such that  $K \subseteq K'$  implies  $g(K) \leq g(K')$ . Then,*

$$\sum_{K \subseteq S, |K|=a} g(K) \leq \sum_{K' \subseteq S, |K'|=a+1} g(K').$$

*Proof.* Consider the bipartite graph  $G := (A, B, E)$  where  $A := \{K \subseteq S \mid |K| = a\}$  and  $B := \{K' \subseteq S \mid |K'| = a + 1\}$  with an edge from  $K$  to  $K'$  exactly if  $K \subseteq K'$ . This is a regular graph of degree  $a + 1$ , hence by Hall's Marriage Theorem 3, there is a matching saturating  $A$ . For any  $K$  and the matching  $K'$ , we have  $g(K) \leq g(K')$ . Hence the result. ■

We invite the reader to prove directly, the following deduction from Theorem 1:

**Corollary 5** *The product measure on subsets has the subset property.*

More interesting consequences of this technique can be found in [1].

## References

- [1] D. Dubhashi and D. Ranjan, "Balls and Bins: A Study in Negative Dependence", submitted to *Random Structures and Algorithms*, 1995.
- [2] J. H. van Lint and R.M. Wilson, *A Course in Combinatorics*, Cambridge University Press, 1992.
- [3] A.W. Marshall and I. Olkin, *Inequalities: Theory of Majorization and its Applications*, Academic Press, New York, 1979.

## Some Surprising Connections between Theoretical Programming Languages and Algorithms

Bob Paige<sup>7</sup>

After spending a year at DIKU and most of a summer at BRICS, and having enjoyed teaching Formal Semantics at NYU for the first time using Glynn's book, I'm convinced that Europe is the New World, and America is the Old as far as PL is concerned. As a bridge person, I found several ways to make New World ideas palatable to people in my neck of the woods. In each of these ways language abstraction was used to obtain algorithmic results.

This last year I worked with a student Deepak Goyal on a compiler for the linear time fragment of Dan Willard's predicate calculus subset, a complex database query language that can be compiled into code guaranteed to run in linear time relative to the input/output space. Willard's model of computation assumes that hashing unit space data takes unit time. Goyal and I found, much to our surprise, that high level implementation code for any query in Willard's original language could be typed in a system developed by Henglein and me that guarantees real-time simulation of each of Willard's hash operations on a RAM with no unreasonable space overhead. We're aiming for a mechanical proof of correctness of the compiler together with the runtime complexity guarantees.

In another example, a student Chang and I made use of rule induction to design and prove the correctness of an algorithm to turn a regular expression of length  $r$  and with  $s$  alphabet symbol occurrences into a compressed NFA representation of size  $O(s)$  in  $O(r)$  time and  $O(s)$  space. The rule induction is based on a nonstandard grammar for regular expressions. This paper is in press at TCS, and can be obtained through the WWW at URL<sup>8</sup>. ☰

## MONA and FIDO

Nils Klarlund<sup>9</sup> and Michael I. Schwartzbach

*Regularity is everywhere, but is often difficult to capture.* This basic observation is the motivation for the MONA and FIDO project.

A number of recent papers have described how to exploit the Monadic Second-Order Logic (M2L) on finite strings and trees [7] to solve interesting and challenging problems. In each case, the results are obtained by exploiting an inherent regularity in the problem domain, thus reducing the problem to questions of regular string or tree languages. Successful applications today include verification of concurrent systems [5], hardware verification [1], and software engineering [4]. Work in progress include pointer verification and synthesis of finite state programs.

The rôle of M2L in this approach is to provide an extraordinarily succinct notation for complicated regular sets. The existing applications have demonstrated that this notation is sufficiently powerful to succeed where finite state automata, regular expressions, and grammars must fail. This is hardly surprising, since M2L is non-elementarily more succinct than other notations. Thus, some formulas in M2L describe regular sets for which the size of a corresponding DFA compared to the size of the formula is not bounded by any finite stack of exponentials.

The flip side of this impressive succinctness is that M2L correspondingly has a non-elementary lower bound on its decision procedure. Surprisingly, the MONA implementation of M2L [3] can handle non-trivial formulas, some as large as 100,000 characters. This is due to the application of BDD techniques [2], specialized algorithms on finite state automata, and careful tuning of the implementation [6].

The successful applications of M2L and MONA reside in a common, productive niche: they require the specification of regular sets that are too

<sup>7</sup>Courant Institute, New York University, USA.

<sup>8</sup><http://cs.nyu.edu/cs/faculty/paige/papers/cnnfa.ps>

<sup>9</sup>Bell Laboratories, USA

complicated to describe by other means, but not so complicated as to be infeasible for our tools.

Anyone interested in using the MONA tool is phased with an unprecedented task: writing huge formulas in M2L. While the basic logic is simple and quite intuitive, early experiences disclosed that the raw notation is a cumbersome and risky tool for “formula programming”. In fact, M2L specifications are uncomfortably similar to assembly code programs.

Driven by the needs of several emerging, non-trivial applications, we have concurrently developed a high-level notation, FIDO, that alleviates these short-comings. FIDO is fully implemented and provides, along with several supporting tools, an optimizing compiler into MONA formulas.

A significant number of BRICS staff members and students have been involved in this project, including: Nils Klarlund, Michael I. Schwartzbach, Theis Rauhe, Morten Biehl Christiansen, Kim Sunesen, Jesper G. Henriksen, Anders Sandholm, Michael Jørgensen, and Jakob Jensen, and the BRICS visitors David A. Basin, Bob Paige, and Abdelwaheb Ayari.

## References

- [1] David A. Basin and Nils Klarlund. Hardware verification using monadic second-order logic. In *Computer aided verification : 7th International Conference, CAV '95, LNCS 939*, 1995.
- [2] Randal E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, August 1986.
- [3] Jesper Gulmann Henriksen, Michael Jørgensen, Jakob Jensen, Nils Klarlund, Bob Paige, Theis Rauhe, and Anders Sandholm. Mona: Monadic second-order logic in practice. In *Proceedings TACAS'95, LNCS 1019*, May 1995.
- [4] Nils Klarlund, Jari Koistinen, and Michael I. Schwartzbach. Formal design constraints. In *Proceedings of OOPSLA'96*, October 1996.
- [5] Nils Klarlund, Mogens Nielsen, and Kim Sunesen. Automated logical verification based on trace abstraction. In *Proceedings of PODC'96*, 1996.
- [6] Nils Klarlund and Theis Rauhe. BDD algorithms and cache misses. Technical Report RS-96-05, BRICS, 1996. Submitted.
- [7] Wolfgang Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 133–191. MIT Press/Elsevier, 1990. ☐

# Calendar of Events

Date	Event
Jun-Aug 1996	Summer Student Program
Mid Aug 1996	<i>Dexter Kozen</i> , Cornell University, NY; mini-course on Set Constraints
End Aug 1996	<i>Susanne Albers</i> , Max-Planck-Institut für Informatik, Saarbrücken; mini-course on Competitive Online Algorithms
Sep 1996	<i>Vladimiro Sassone</i> , University of Pisa; mini-course on Higher Dimensional Automata
10–11 Oct 1996	<i>Kristoffer Høgsbro Rose</i> , BRICS; mini-course on Explicit Substitution
Nov 1996	<i>P. S. Thiagarajan</i> , SPIC Science Foundation, Madras; mini-course on Distributed Logics
Autumn 1996	BRICS theme on Verification
Week 44 Oct 1996	Autumn School on Theorem Proving and Model Checking
1997	CSL '97 (Computer Science Logic).

## BRICS Address and World Wide Web

Please **note** that the Centre now has got new (and shorter) net addresses! The old ones, however, will continue to be valid.

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, at

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:

<http://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
cd pub/BRICS
get README.
```



### BRICS Newsletter

ISSN 0909-6043

**Editors:** Glynn Winskel & Uffe H. Engberg

**Lay-out:** Uffe H. Engberg

**Publisher:** BRICS

**Print:** Departments of Mathematical Sciences  
University of Aarhus

© BRICS 1996