BRICS Research Series, Abstracts, 2004

May 20, 2005

This document is also available as *PostScript* and *DVI.*

References

[RS-04-41] *PostScript*, *PDF*, *DVI.*
Olivier Danvy.
Sur un Exemple de Patrick Greussay.
December 2004.
14 pp.
Abstract: This note was written at the occasion of the retirement of Jean-Francois Perrot at the Universite Pierre et Marie Curie (Paris VI). In an attempt to emulate his academic spirit, we revisit an example proposed by Patrick Greussay in his doctoral thesis: how to verify in sublinear time whether a Calder mobile is well balanced. Rather than divining one solution or another, we derive a spectrum of solutions, starting from the original specification of the problem. We also prove their correctness.

[RS-04-40] *PostScript*, *PDF.*
Mads Sig Ager, Olivier Danvy, and Henning Korsholm Rohde.
Fast Partial Evaluation of Pattern Matching in Strings.
December 2004.
22 pp. To appear in TOPLAS. Supersedes BRICS report RS-03-20.
Abstract: We show how to obtain all of Knuth, Morris, and Pratt's linear-time string matcher by specializing a quadratic-time string matcher with respect to a pattern string. Although it has been known for 15 years how to obtain this linear matcher by partial evaluation of a quadratic one, how to obtain it in linear time has remained an open problem.

Obtaining a linear matcher by partial evaluation of a quadratic one is achieved by performing its backtracking at specialization time and memoizing its results. We show (1) how to rewrite the source matcher such that its static intermediate computations can be shared at specialization time and (2) how to extend the memoization capabilities of a partial evaluator to static functions. Such an extended partial evaluator, if its memoization is implemented efficiently, specializes the rewritten source matcher in linear time.

Finally, we show that the method also applies to a variant of Boyer and Moore's string matcher.

[RS-04-39] *PostScript*, *PDF*, *DVI*.
Olivier Danvy and Lasse R. Nielsen.
CPS Transformation of Beta-Redexes.
December 2004.
ii+11 pp. Superseedes an article to appear in *Information Processing Letters* and BRICS report RS-00-35.

Abstract: The extra compaction of the most compacting CPS transformation in existence, which is due to Sabry and Felleisen, is generally attributed to (1) making continuations occur first in CPS terms and (2) classifying more redexes as administrative. We show that this extra compaction is actually independent of the relative positions of values and continuations and furthermore that it is solely due to a context-sensitive transformation of beta-redexes. We stage the more compact CPS transformation into a first-order uncurrying phase and a context-insensitive CPS transformation. We also define a context-insensitive CPS transformation that provides the extra compaction. This CPS transformation operates in one pass and is dependently typed.

[RS-04-38] Olin Shivers and Mitchell Wand.
Bottom-Up β-Substitution: Uplinks and λ-DAGs.
December 2004.

Abstract: Terms of the lambda-calculus are one of the most important data structures we have in computer science. Among their uses are representing program terms, advanced type systems, and proofs in theorem provers. Unfortunately, heavy use of this data structure can become intractable in time and space; the typical culprit is the fundamental operation of beta reduction.

If we represent a lambda-calculus term as a DAG rather than a tree, we can efficiently represent the sharing that arises from beta reduction, thus avoiding combinatorial explosion in space. By adding uplinks from a child to its parents, we can efficiently implement beta reduction in a bottom-up manner, thus avoiding combinatorial explosion in time required to search the term in a top-down fashion.

We present an algorithm for performing beta reduction on lambda terms represented as uplinked DAGs; describe its proof of correctness; discuss its relation to alternate techniques such as Lamping graphs, the suspension lambda-calculus (SLC) and director strings; and present some timings of an implementation.

Besides being both fast and parsimonious of space, the algorithm is particularly suited to applications such as compilers, theorem provers, and type-manipulation systems that may need to examine terms in-between reductions – i.e., the "readback" problem for our representation is trivial.

Like Lamping graphs, and unlike director strings or the suspension lambda-calculus, the algorithm functions by side-effecting the term containing the redex; the representation is not a "persistent" one.

The algorithm additionally has the charm of being quite simple; a complete implementation of the core data structures and algorithms is 180 lines of fairly straightforward SML.

[RS-04-37] *PostScript*, *PDF*.
Jørgen Iversen and Peter D. Mosses.
Constructive Action Semantics for Core ML.
December 2004.
68 pp. To appear in a special *Language Definitions and Tool Generation* issue of the journal *IEE Proceedings Software*.

Abstract: Usually, the majority of language constructs found in a programming language can also be found in many other languages, because language design is based on reuse. This should be reflected in the way we give semantics to programming languages. It can be achieved by making a language description consist of a collection of modules, each defining a single language construct. The description of a single language construct should be language independent, so that it can be reused in other descriptions without any changes. We call a language description framework "constructive" when it supports independent description of individual constructs.

We present a case study in constructive semantic description. The case study is a description of Core ML, consisting of a mapping from it to BAS (Basic Abstract Syntax) and action semantic descriptions of the individual BAS constructs. The latter are written in ASDF (Action Semantics Definition Formalism), a formalism specially designed for writing action semantic descriptions of single language constructs. Tool support is provided by the ASF+SDF Meta-Environment and by the Action Environment, which is a new extension of the ASF+SDF Meta-Environment.

[RS-04-36] *PostScript*, *PDF*.
Mark van den Brand, Jørgen Iversen, and Peter D. Mosses.
An Action Environment.
December 2004.
27 pp. Appears in Hedin and Van Wyk, editors, *Fourth ACM SIGPLAN Workshop on Language Descriptions, Tools and Applications*, LDTA '04, 2004, pages 149–168.

Abstract: Some basic programming constructs (e.g., conditional statements) are found in many different programming languages, and can often be included without change when a new language is designed. When writing a semantic description of a language, however, it is usually not possible to reuse parts of previous descriptions without change.

This paper introduces a new formalism, ASDF, which has been designed specifically for giving reusable action semantic descriptions of individual language constructs. An initial case study in the use of ASDF has already provided reusable descriptions of

all the basic constructs underlying Core ML.

The paper also describes the Action Environment, a new environment supporting use and validation of ASDF descriptions. The Action Environment has been implemented on top of the ASF+SDF Meta-Environment, exploiting recent advances in techniques for integration of different formalisms, and inheriting all the main features of the Meta-Environment.

[RS-04-35] *PostScript*, *PDF*, *DVI*.
Jørgen Iversen.
Type Checking Semantic Functions in ASDF.
December 2004.
29 pp.

Abstract: When writing semantic descriptions of programming languages, it is convenient to have tools for checking the descriptions. With frameworks that use inductively defined semantic functions to map programs to their denotations, we would like to check that the semantic functions result in denotations with certain properties. In this paper we present a type system for a modular style of the action semantic framework that, given signatures of all the semantic functions used in a semantic equation defining a semantic function, performs a soft type check on the action in the semantic equation.

We introduce types for actions that describe different properties of the actions, like the type of data they expect and produce, whether they can fail or have side effects, etc. A type system for actions which uses these new action types is presented. Using the new action types in the signatures of semantic functions, the language describer can assert properties of semantic functions and have the assertions checked by an implementation of the type system.

The type system has been implemented for use in connection with the recently developed formalism ASDF. The formalism supports writing language definitions by combining modules that describe single language constructs. This is possible due to the inherent modularity in ASDF. We show how we manage to preserve the modularity and still perform specialised type checks for each module.

[RS-04-34] *PostScript*, *PDF*, *DVI*.
Anders Møller and Michael I. Schwartzbach.
The Design Space of Type Checkers for XML Transformation Languages.
December 2004.
21 pp. Appears in Eiter and Libkin, editors, *Database Theory: 10th International Conference*, ICDT '05 Proceedings, LNCS 3363, 2005, pages 17–36.

Abstract: We survey work on statically type checking XML transformations, covering a wide range of notations and ambitions. The concept of *type* may vary from idealizations of DTD to full-

blown XML Schema or even more expressive formalisms. The notion of *transformation* may vary from clean and simple transductions to domain-specific languages or integration of XML in general-purpose programming languages. Type annotations can be either explicit or implicit, and type checking ranges from exact decidability to pragmatic approximations.

We characterize and evaluate existing tools in this design space, including a recent result of the authors providing practical type checking of full unannotated XSLT 1.0 stylesheets given general DTDs that describe the input and output languages.

[RS-04-33] *PostScript*, *PDF*.
Aske Simon Christensen, Christian Kirkegaard, and Anders Møller.
A Runtime System for XML Transformations in Java.
December 2004.
15 pp. Appears in Bellahsene, Milo, Rys, Suciu and Unland, editors, *Database and XML Technologies: Second International XML Database Symposium*, XSym '04 Proceedings, LNCS 3186, 2004, pages 143–157. Supersedes the earlier BRICS report RS-03-29.

Abstract: We show that it is possible to extend a general-purpose programming language with a convenient high-level data-type for manipulating XML documents while permitting (1) precise static analysis for guaranteeing validity of the constructed XML documents relative to the given DTD schemas, and (2) a runtime system where the operations can be performed efficiently. The system, named Xact, is based on a notion of immutable XML templates and uses XPath for deconstructing documents. A companion paper presents the program analysis; this paper focuses on the efficient runtime representation.

[RS-04-32] *PostScript*, *PDF*, *DVI*.
Philipp Gerhardy.
A Quantitative Version of Kirk's Fixed Point Theorem for Asymptotic Contractions.
December 2004.
9 pp.

Abstract: In [J.Math.Anal.App.277(2003) 645-650], W.A.Kirk introduced the notion of asymptotic contractions and proved a fixed point theorem for such mappings. Using techniques from proof mining, we develop a variant of the notion of asymptotic contractions and prove a quantitative version of the corresponding fixed point theorem.

[RS-04-31] *PostScript*, *PDF*, *DVI*.
Philipp Gerhardy and Ulrich Kohlenbach.
Strongly Uniform Bounds from Semi-Constructive Proofs.
December 2004.
31 pp.

Abstract: In 2003, the second author obtained metatheorems for the extraction of effective (uniform) bounds from classical, prima facie non-constructive proofs in functional analysis. These metatheorems for the first time cover general classes of struc-

tures like arbitrary metric, hyperbolic, CAT(0) and normed linear spaces and guarantee the independence of the bounds from parameters raging over metrically bounded (not necessarily compact!) spaces. The use of classical logic imposes some severe restrictions on the formulas and proofs for which the extraction can be carried out. In this paper we consider similar metatheorems for semi-intuitionistic proofs, i.e. proofs in an intuitionistic setting enriched with certain non-constructive principles. Contrary to the classical case, there are practically no restrictions on the logical complexity of theorems for which bounds can be extracted. Again, our metatheorems guarantee very general uniformities, even in cases where the existence of uniform bounds is not obtainable by (ineffective) straightforward functional analytic means. Already in the purely intuitionistic case, where the existence of effective bounds is implicit, the metatheorems allow one to derive uniformities that may not be obvious at all from a given constructive proofs. Finally, we illustrate our main metatheorem by an example from metric fixed point theory.

[RS-04-30] *PostScript*, *PDF*, *DVI*.
Olivier Danvy.
From Reduction-Based to Reduction-Free Normalization.
December 2004.
27 pp. Invited talk at the *4th International Workshop on Reduction Strategies in Rewriting and Programming*, WRS 2004 (Aachen, Germany, June 2, 2004). To appear in ENTCS.

Abstract: We present a systematic construction of a reduction-free normalization function. Starting from a reduction-based normalization function, i.e., the transitive closure of a one-step reduction function, we successively subject it to refocusing (i.e., deforestation of the intermediate reduced terms), simplification (i.e., fusing auxiliary functions), refunctionalization (i.e., Church encoding), and direct-style transformation (i.e., the converse of the CPS transformation). We consider two simple examples and treat them in detail: for the first one, arithmetic expressions, we construct an evaluation function; for the second one, terms in the free monoid, we construct an accumulator-based flatten function. The resulting two functions are traditional reduction-free normalization functions.

The construction builds on previous work on refocusing and on a functional correspondence between evaluators and abstract machines. It is also reversible.

[RS-04-29] *PostScript*, *PDF*, *DVI*.
Małgorzata Biernacka, Dariusz Biernacki, and Olivier Danvy.
An Operational Foundation for Delimited Continuations in the CPS Hierarchy.
December 2004.
iii+45 pp. A preliminary version appeared in Thielecke, editor, *4th ACM SIGPLAN Workshop on Continuations*, CW '04 Proceedings, Association for Computing Machinery (ACM) SIGPLAN

Technical Reports CSR-04-1, 2004, pages 25–33.

Abstract: We present an abstract machine and a reduction semantics for the λ-calculus extended with control operators that give access to delimited continuations in the CPS hierarchy. The abstract machine is derived from an evaluator in continuation-passing style (CPS); the reduction semantics (i.e., a small-step operational semantics with an explicit representation of evaluation contexts) is constructed from the abstract machine; and the control operators are the shift and reset family. At level n of the CPS hierarchy, programs can use the control operators shift_i and reset_i for $1 \leq i \leq n$, the evaluator has $n + 1$ layers of continuations, the abstract machine has $n+1$ layers of control stacks, and the reduction semantics has $n + 1$ layers of evaluation contexts.

We also present new applications of delimited continuations in the CPS hierarchy: finding list prefixes and normalization by evaluation for a hierarchical language of units and products.

[RS-04-28] *PostScript*, *PDF*, *DVI*.

Mads Sig Ager, Olivier Danvy, and Jan Midtgaard.

A Functional Correspondence between Monadic Evaluators and Abstract Machines for Languages with Computational Effects.

December 2004.

44 pp. Extended version of an article to appear in *Theoretical Computer Science*.

Abstract: We extend our correspondence between evaluators and abstract machines from the pure setting of the lambda-calculus to the impure setting of the computational lambda-calculus. We show how to derive new abstract machines from monadic evaluators for the computational lambda-calculus. Starting from (1) a generic evaluator parameterized by a monad and (2) a monad specifying a computational effect, we inline the components of the monad in the generic evaluator to obtain an evaluator written in a style that is specific to this computational effect. We then derive the corresponding abstract machine by closure-converting, CPS-transforming, and defunctionalizing this specific evaluator. We illustrate the construction first with the identity monad, obtaining the CEK machine, and then with a lifting monad, a state monad, and with a lifted state monad, obtaining variants of the CEK machine with error handling, state and a combination of error handling and state.

In addition, we characterize the tail-recursive stack inspection presented by Clements and Felleisen as a lifted state monad. This enables us to combine this stack-inspection monad with other monads and to construct abstract machines for languages with properly tail-recursive stack inspection and other computational effects. The construction scales to other monads—including one more properly dedicated to stack inspection than the lifted state monad—and other monadic evaluators.

[RS-04-27] *PostScript*, *PDF*.

Gerth Stølting Brodal, Rolf Fagerberg, and Gabriel Moruz.

On the Adaptiveness of Quicksort.
December 2004.
23 pp. To appear in Demetrescu and Tamassia, editors, *Seventh Workshop on Algorithm Engineering and Experiments*, ALENEX '05 Proceedings, 2005.

Abstract: Quicksort was first introduced in 1961 by Hoare. Many variants have been developed, the best of which are among the fastest generic sorting algorithms available, as testified by the choice of Quicksort as the default sorting algorithm in most programming libraries. Some sorting algorithms are adaptive, i.e. they have a complexity analysis which is better for inputs which are nearly sorted, according to some specified measure of presortedness. Quicksort is not among these, as it uses $\Omega(n \log n)$ comparisons even when the input is already sorted. However, in this paper we demonstrate empirically that the actual running time of Quicksort *is* adaptive with respect to the presortedness measure Inv. Differences close to a factor of two are observed between instances with low and high Inv value. We then show that for the randomized version of Quicksort, the number of element *swaps* performed is *provably* adaptive with respect to the measure Inv. More precisely, we prove that randomized Quicksort performs expected $O(n(1 + \log(1 + \mathrm{Inv}/n)))$ element swaps, where Inv denotes the number of inversions in the input sequence. This result provides a theoretical explanation for the observed behavior, and gives new insights on the behavior of the Quicksort algorithm. We also give some empirical results on the adaptive behavior of Heapsort and Mergesort.

[RS-04-26] *PostScript*, *PDF*, *DVI*.
Olivier Danvy and Lasse R. Nielsen.
Refocusing in Reduction Semantics.
November 2004.
iii+44 pp. This report supersedes BRICS report RS-02-04. A preliminary version appears in the informal proceedings of the *Second International Workshop on Rule-Based Programming*, RULE 2001, Electronic Notes in Theoretical Computer Science, Vol. 59.4.

Abstract: The evaluation function of a reduction semantics (i.e., a small-step operational semantics with an explicit representation of the reduction context) is canonically defined as the transitive closure of (1) decomposing a term into a reduction context and a redex, (2) contracting this redex, and (3) plugging the contractum in the context. Directly implementing this evaluation function therefore yields an interpreter with a worst-case overhead, for each step, that is linear in the size of the input term.

We present sufficient conditions over the constituents of a reduction semantics to circumvent this overhead, by replacing the composition of (3) plugging and (1) decomposing by a single "refocus" function mapping a contractum and a context into a new context and a new redex, if any. We also show how to construct

such a refocus function, we prove the correctness of this construction, and we analyze the complexity of the resulting refocus function.

The refocused evaluation function of a reduction semantics implements the transitive closure of the refocus function, i.e., a "pre-abstract machine." Fusing the refocus function with the trampoline function computing the transitive closure gives a state-transition function, i.e., an abstract machine. This abstract machine clearly separates between the transitions implementing the congruence rules of the reduction semantics and the transitions implementing its reduction rules. The construction of a refocus function therefore shows how to mechanically obtain an abstract machine out of a reduction semantics, which was done previously on a case-by-case basis.

We illustrate refocusing by mechanically constructing Felleisen et al.'s CK machine from a call-by-value reduction semantics of the lambda-calculus, and by constructing a substitution-based version of Krivine's machine from a call-by-name reduction semantics of the lambda-calculus. We also mechanically construct three one-pass CPS transformers from three quadratic context-based CPS transformers for the lambda-calculus.

[RS-04-25] *PostScript*, *PDF*, *DVI*.
Mayer Goldberg.
On the Recursive Enumerability of Fixed-Point Combinators.
November 2004.
7 pp. Superseded by the later report BRICS RS-05-1.

Abstract: We show that the set of fixed-point combinators forms a recursively-enumerable subset of a larger set of terms that is (A) not recursively enumerable, and (B) the terms of which are observationally equivalent to fixed-point combinators in any computable context.

[RS-04-24] *PostScript*, *PDF*, *DVI*.
Luca Aceto, Willem Jan Fokkink, Anna Ingólfsdóttir, and Sumit Nain.
Bisimilarity is not Finitely Based over BPA with Interrupt.
October 2004.
30 pp.

Abstract: This paper shows that bisimulation equivalence does not afford a finite equational axiomatization over the language obtained by enriching Bergstra and Klop's Basic Process Algebra with the interrupt operator. Moreover, it is shown that the collection of closed equations over this language is also not finitely based.

[RS-04-23] *PostScript*, *PDF*, *DVI*.
Hans Hüttel and Jiří Srba.
Recursion vs. Replication in Simple Cryptographic Protocols.
October 2004.

26 pp. To appear in Vojtas, editor, *31st Conference on Current Trends in Theory and Practice of Informatics*, SOFSEM '05 Proceedings, LNCS, 2005.

Abstract: We use some recent techniques from process algebra to draw several conclusions about the well studied class of ping-pong protocols introduced by Dolev and Yao. In particular we show that all nontrivial properties, including reachability and equivalence checking wrt. the whole van Glabbeek's spectrum, become undecidable for a very simple recursive extension of the protocol. The result holds even if no nondeterministic choice operator is allowed. We also show that the extended calculus is capable of an implicit description of the active intruder, including full analysis and synthesis of messages in the sense of Amadio, Lugiez and Vanackere. We conclude by showing that reachability analysis for a replicative variant of the protocol becomes decidable.

[RS-04-22] *PostScript*, *PDF*.
Gian Luca Cattani and Glynn Winskel.
Profunctors, Open Maps and Bisimulation.
October 2004.
64 pp. To appear in *Mathematical Structures in Computer Science.*

Abstract: This paper studies fundamental connections between profunctors (i.e., distributors, or bimodules), open maps and bisimulation. In particular, it proves that a colimit preserving functor between presheaf categories (corresponding to a profunctor) preserves open maps and open map bisimulation. Consequently, the composition of profunctors preserves open maps as 2-cells. A guiding idea is the view that profunctors, and colimit preserving functors, are linear maps in a model of classical linear logic. But profunctors, and colimit preserving functors, as linear maps, are too restrictive for many applications. This leads to a study of a range of pseudo-comonads and how non-linear maps in their co-Kleisli bicategories preserve open maps and bisimulation. The pseudo-comonads considered are based on finite colimit completion, "lifting", and indexed families. The paper includes an appendix summarising the key results on coends, left Kan extensions and the preservation of colimits. One motivation for this work is that it provides a mathematical framework for extending domain theory and denotational semantics of programming languages to the more intricate models, languages and equivalences found in concurrent computation. But the results are likely to have more general applicability because of the ubiquitous nature of profunctors.

[RS-04-21] *PostScript*, *PDF*, *DVI*.
Glynn Winskel and Francesco Zappa Nardelli.
New-HOPLA—A Higher-Order Process Language with Name Generation.
October 2004.
38 pp. Appears in Lévy, Mayr and Mitchell, editors, *3rd IFIP International Conference on Theoretical Computer Science*, TCS '04

Proceedings, 2004, pages 521–534.

Abstract: This paper introduces new-HOPLA, a concise but powerful language for higher-order nondeterministic processes with name generation. Its origins as a metalanguage for domain theory are sketched but for the most part the paper concentrates on its operational semantics. The language is typed, the type of a process describing the shape of the computation paths it can perform. Its transition semantics, bisimulation, congruence properties and expressive power are explored. Encodings are given of well-known process algebras, including pi-calculus, Higher-Order pi-calculus and Mobile Ambients.

[RS-04-20] *PostScript*, *PDF*, *DVI.*
Mads Sig Ager.
From Natural Semantics to Abstract Machines.
October 2004.
21 pp. Presented at the *International Symposium on Logic-based Program Synthesis and Transformation*, LOPSTR 2004, Verona, Italy, August 26–28, 2004.

Abstract: We describe how to construct correct abstract machines from the class of L-attributed natural semantics introduced by Ibraheem and Schmidt at HOOTS 1997. The construction produces stack-based abstract machines where the stack contains evaluation contexts. It is defined directly on the natural semantics rules. We formalize it as an extraction algorithm and we prove that the algorithm produces abstract machines that are equivalent to the original natural semantics. We illustrate the algorithm by extracting abstract machines from natural semantics for call-by-value, call-by-name, and call-by-need evaluation of lambda terms.

[RS-04-19] *PostScript*, *PDF.*
Bolette Ammitzbøll Madsen and Peter Rossmanith.
Maximum Exact Satisfiability: NP-completeness Proofs and Exact Algorithms.
October 2004.
20 pp.

Abstract: Inspired by the Maximum Satisfiability and Exact Satisfiability problems we present two Maximum Exact Satisfiability problems. The first problem called Maximum Exact Satisfiability is: given a formula in conjunctive normal form and an integer k, is there an assignment to all variables in the formula such that at least k clauses have exactly one true literal. The second problem called Restricted Maximum Exact Satisfiability has the further restriction that no clause is allowed to have more than one true literal. Both problems are proved NP-complete restricted to the versions where each clause contains at most two literals. In fact Maximum Exact Satisfiability is a generalisation of the well-known NP-complete problem MaxCut. We present an exact algorithm for Maximum Exact Satisfiability where each clause contains at most two literals with time complexity $O(poly(L)\cdot 2^{m/4})$, where m is the number of clauses and L is the length of the formula. For the second version we give an algorithm with time complexity $O(poly(L) \cdot 1.324718^n)$, where n is the number of variables. We note that when restricted to the versions where each clause contains exactly two literals and there are no negations both problems are fixed parameter tractable. It is an open question if this is also the case for the general problems.

[RS-04-18] *PostScript*, *PDF*, *DVI*.
Bolette Ammitzbøll Madsen.
An Algorithm for Exact Satisfiability Analysed with the Number of Clauses as Parameter.
September 2004.
4 pp.

Abstract: We give an algorithm for Exact Satisfiability with polynomial space usage and a time bound of $poly(L) \cdot m!$, where m is the number of clauses and L is the length of the formula. Skjernaa has given an algorithm for Exact Satisfiability with time bound $poly(L) \cdot 2^m$ but using exponential space. We leave the following problem open: Is there an algorithm for Exact Satisfiability using only polynomial space with a time bound of c^m, where c is a constant and m is the number of clauses?

[RS-04-17] *PostScript*, *PDF*, *DVI*.
Mayer Goldberg.
Computing Logarithms Digit-by-Digit.
September 2004.
6 pp.

Abstract: In this work, we present an algorithm for computing logarithms of positive real numbers, that bares structural resemblance to the elementary school algorithm of long division. Using this algorithm, we can compute successive digits of a logarithm using a 4-operation pocket calculator. The algorithm makes no use of Taylor series or calculus, but rather exploits properties of the radix-d representation of a logarithm in base d. As such, the algorithm is accessible to anyone familiar with the elementary properties of exponents and logarithms.

[RS-04-16] *PostScript*, *PDF*.
Karl Krukow and Andrew Twigg.
Distributed Approximation of Fixed-Points in Trust Structures.
September 2004.
25 pp.

Abstract: Recently, developments of sophisticated formal models of trust in large distributed environments, incorporate aspects of partial information, important e.g. in global-computing scenarios. Specifically, the framework based on the notion of trust structures, introduced by Carbone, Nielsen and Sassone, deals well with the aspect of partial information. The framework is "denotational" in the sense of giving meaning to the global trust-state as a unique, abstract mathematical object (the least fixed-point of a continuous function). We complement the denotational

framework with "operational" techniques, addressing the practically important problem of approximating and computing the semantic objects. We show how to derive from the setting of the framework, a situation in which one may apply a well-established distributed algorithm, due to Bertsekas, in order to solve the problem of computation and approximation of least fixed-points of continuous functions on cpos. We introduce mild assumptions about trust structures, enabling us to derive two theoretically simple, but highly useful propositions (and their duals), which form the basis for efficient protocols for sound approximation of the least fixed-point. Finally, we give dynamic algorithms for safe reuse of information between computations, in face of dynamic trust-policy updates.

[RS-04-15] *PostScript*, *PDF*, *DVI*.
Jesús Fernando Almansa.
The Full Abstraction of the UC Framework.
August 2004.
ii+24 pp.
Abstract: Two different approaches for general protocol security are proved equivalent. Concretely, we prove that security in the Universal Composability framework (UC) is equivalent to security in the probabilistic polynomial time calculus ppc. Security is defined under active and adaptive adversaries with synchronous and authenticated communication. In detail, we define an encoding from machines in UC to processes in ppc and show UC is fully abstract in ppc, i.e., we show the soundness and completeness of security in ppc with respect to UC. However, we restrict security in ppc to be quantified not over all possible contexts, but over those induced by UC-environments under encoding. This result is not overly-restricting security in ppc, since the threat and communication models we assume are meaningful in both practice and theory.

[RS-04-14] *PostScript*, *PDF*, *DVI*.
Jesper Makholm Byskov.
Maker-Maker and Maker-Breaker Games are PSPACE-Complete.
August 2004.
5 pp.
Abstract: We show that the problems of deciding the outcome of Maker-Maker and Maker-Breaker games played on arbitrary hypergraphs are PSPACE-complete. Maker-Breaker games have earlier been shown PSPACE-complete by Schaefer (1978); we give a simpler proof and show a reduction from Maker-Maker games to Maker-Breaker games.

[RS-04-13] *PostScript*, *PDF*, *DVI*.
Jens Groth and Gorm Salomonsen.
Strong Privacy Protection in Electronic Voting.
July 2004.
12 pp. Preliminary abstract presented at Tjoa and Wagner, editors, *13th International Workshop on Database and Expert Systems*

Applications, DEXA '02 Proceedings, 2002, page 436.

Abstract: We give suggestions for protection against adversaries with access to the voter's equipment in voting schemes based on homomorphic encryption. Assuming an adversary has complete knowledge of the contents and computations taking place on the client machine we protect the voter's privacy in a way so that the adversary has no knowledge about the voter's choice. Furthermore, an active adversary trying to change a voter's ballot may do so, but will end up voting for a random candidate.

To accomplish the goal we assume that the voter has access to a secondary communication channel through which he can receive information inaccessible to the adversary. An example of such a secondary communication channel is ordinary mail. Additionally, we assume the existence of a trusted party that will assist in the protocol. To some extent, the actions of this trusted party are verifiable.

[RS-04-12] *PostScript*, *PDF*, *DVI.*
Olivier Danvy and Ulrik P. Schultz.
Lambda-Lifting in Quadratic Time.
June 2004.
34 pp. Appears in *Journal of Functional and Logic Programming*, 1:43, 2004. This report supersedes the earlier BRICS report RS-03-36 which was an extended version of a paper appearing in Hu and Rodríguez-Artalejo, editors, *Sixth International Symposium on Functional and Logic Programming*, FLOPS '02 Proceedings, LNCS 2441, 2002, pages 134–151.

Abstract: Lambda-lifting is a program transformation that is used in compilers, partial evaluators, and program transformers. In this article, we show how to reduce its complexity from cubic time to quadratic time, and we present a flow-sensitive lambda-lifter that also works in quadratic time.

Lambda-lifting transforms a block-structured program into a set of recursive equations, one for each local function in the source program. Each equation carries extra parameters to account for the free variables of the corresponding local function *and of all its callees.* It is the search for these extra parameters that yields the cubic factor in the traditional formulation of lambda-lifting, which is due to Johnsson. This search is carried out by computing a transitive closure.

To reduce the complexity of lambda-lifting, we partition the call graph of the source program into strongly connected components, based on the simple observation that *all functions in each component need the same extra parameters and thus a transitive closure is not needed.* We therefore simplify the search for extra parameters by treating each strongly connected component instead of each function as a unit, thereby reducing the time complexity of lambda-lifting from $O(n^3)$ to $O(n^2)$, where n is the size of the program.

Since a lambda-lifter can output programs of size $O(n^2)$, our

algorithm is asympotically optimal.

[RS-04-11] *PostScript*, *PDF*.
Vladimiro Sassone and Paweł Sobociński.
Congruences for Contextual Graph-Rewriting.
June 2004.
29 pp.
Abstract: We introduce a comprehensive operational semantic theory of graph rewriting. The central idea is recasting rewriting frameworks as Leifer and Milner's reactive systems. Consequently, graph rewriting systems are associated with canonical labelled transition systems, on which bisimulation equivalence is a congruence with respect to arbitrary graph contexts (cospans of graphs). This construction is derived from a more general theorem of much wider applicability. Expressed in abstract categorical terms, the central technical contribution of the paper is the construction of groupoidal relative pushouts, introduced and developed by the authors in recent work, in suitable cospan categories over arbitrary adhesive categories. As a consequence, we both generalise and shed light on rewriting via borrowed contexts due to Ehrig and König.

[RS-04-10] *PostScript*, *PDF*, *DVI*.
Daniele Varacca, Hagen Völzer, and Glynn Winskel.
Probabilistic Event Structures and Domains.
June 2004.
41 pp. Extended version of an article appears in Gardner and Yoshida, editors, *Concurrency Theory: 15th International Conference*, CONCUR '04 Proceedings, LNCS 3170, 2004, pages 481–496.
Abstract: This paper studies how to adjoin probability to event structures, leading to the model of probabilistic event structures. In their simplest form probabilistic choice is localised to cells, where conflict arises; in which case probabilistic independence coincides with causal independence. An application to the semantics of a probabilistic CCS is sketched. An event structure is associated with a domain—that of its configurations ordered by inclusion. In domain theory probabilistic processes are denoted by continuous valuations on a domain. A key result of this paper is a representation theorem showing how continuous valuations on the domain of a confusion-free event structure correspond to the probabilistic event structures it supports. We explore how to extend probability to event structures which are not confusion-free via two notions of probabilistic runs of a general event structure. Finally, we show how probabilistic correlation and probabilistic event structures with confusion can arise from event structures which are originally confusion-free by using morphisms to rename and hide events.

[RS-04-9] *PostScript*, *PDF*, *DVI*.
Ivan B. Damgård, Serge Fehr, and Louis Salvail.

Zero-Knowledge Proofs and String Commitments Withstanding Quantum Attacks.
May 2004.
22 pp. Appears in Franklin, editor, *Advances in Cryptology: 24th Annual International Cryptology Conference*, CRYPTO '04 Proceedings, LNCS 3152, 2003, pages 254–272.

Abstract: The concept of zero-knowledge (ZK) has become of fundamental importance in cryptography. However, in a setting where entities are modeled by quantum computers, classical arguments for proving ZK fail to hold since, in the quantum setting, the concept of rewinding is not generally applicable. Moreover, known classical techniques that avoid rewinding have various shortcomings in the quantum setting.

We propose new techniques for building *quantum* zero-knowledge (QZK) protocols, which remain secure even under (active) quantum attacks. We obtain computational QZK proofs and perfect QZK arguments for any NP language in the common reference string model. This is based on a general method converting an important class of classical honest-verifier ZK (HVZK) proofs into QZK proofs. This leads to quite practical protocols if the underlying HVZK proof is efficient. These are the first proof protocols enjoying these properties, in particular the first to achieve perfect QZK.

As part of our construction, we propose a general framework for building unconditionally hiding (trapdoor) string commitment schemes, secure against quantum attacks, as well as concrete instantiations based on specific (believed to be) hard problems. This is of independent interest, as these are the first unconditionally hiding string commitment schemes withstanding quantum attacks.

Finally, we give a partial answer to the question whether QZK is possible in the plain model. We propose a new notion of QZK, *non-oblivious verifier* QZK, which is strictly stronger than honest-verifier QZK but weaker than full QZK, and we show that this notion can be achieved by means of efficient (quantum) protocols.

[RS-04-8] *PostScript*, *PDF*, *DVI.*
Petr Jančar and Jiří Srba.
Highly Undecidable Questions for Process Algebras.
April 2004.
25 pp. Appears in Lévy, Mayr and Mitchell, editors, *3rd IFIP International Conference on Theoretical Computer Science*, TCS '04 Proceedings, 2004, pages 507–520.

Abstract: We show Σ^1_1-completeness of weak bisimilarity for PA (process algebra), and of weak simulation preorder/equivalence for PDA (pushdown automata), PA and PN (Petri nets). We also show Π^1_1-hardness of weak omega-trace equivalence for the (sub)classes BPA (basic process algebra) and BPP (basic parallel processes).

Comments: http://pauillac.inria.fr/ levy/TCS2004/.

[RS-04-7] *PostScript*, *PDF*, *DVI*.
Mojmír Křetínský, Vojtěch Řehák, and Jan Strejček.
On the Expressive Power of Extended Process Rewrite Systems.
April 2004.
18 pp.

Abstract: We provide a unified view on three extensions of Process rewrite systems (PRS) and compare their and PRS's expressive power. We show that the class of Petri Nets is less expressible up to bisimulation than the class of Process Algebra extended with finite state control unit. Further we show our main result that the reachability problem for PRS extended with a so called weak finite state unit is decidable.

[RS-04-6] *PostScript*, *PDF*, *DVI*.
Gudmund Skovbjerg Frandsen and Igor E. Shparlinski.
On Reducing a System of Equations to a Single Equation.
March 2004.
11 pp. Appears in Schicho and Singer, editors, *ACM SIGSAM International Symposium on Symbolic and Algebraic Computation*, ISSAC '04 Proceedings, 2004, pages 163–166.

Abstract: For a system of polynomial equations over Q_p we present an efficient construction of a single polynomial of quite small degree whose zero set over Q_p coincides with the zero set over Q_p of the original system. We also show that the polynomial has some other attractive features such as low additive and straight-line complexity.

The proof is based on a link established here between the above problem and some recent number theoretic result about zeros of p-adic forms.

[RS-04-5] *PostScript*, *PDF*, *DVI*.
Dariusz Biernacki and Olivier Danvy.
From Interpreter to Logic Engine by Defunctionalization.
March 2004.
20 pp. Appears in Bruynooghe, editor, *International Symposium on Logic Based Program Development and Transformation, LOPSTR '03 Proceedings*, Revised Selected Papers, LNCS 3018, 2004, pages 143–159. This report supersedes the earlier BRICS report RS-03-25.

Abstract: Starting from a continuation-based interpreter for a simple logic programming language, propositional Prolog with cut, we derive the corresponding logic engine in the form of an abstract machine. The derivation originates in previous work (our article at PPDP 2003) where it was applied to the lambda-calculus. The key transformation here is Reynolds's defunctionalization that transforms a tail-recursive, continuation-passing interpreter into a transition system, i.e., an abstract machine. Similar denotational and operational semantics were studied by de Bruin and de Vink (their article at TAPSOFT 1989), and we compare their study with our derivation. Additionally, we present a direct-style interpreter of propositional Prolog expressed with control operators for delimited continuations.

[RS-04-4] *PostScript*, *PDF*.

Patricia Bouyer, Franck Cassez, Emmanuel Fleury, and Kim G. Larsen.

Optimal Strategies in Priced Timed Game Automata.

February 2004.

32 pp.

Abstract: Priced timed (game) automata extends timed (game) automata with costs on both locations and transitions. In this paper we focus on reachability games for priced timed game automata and prove that the optimal cost for winning such a game is computable under conditions concerning the non-zenoness of cost. Under stronger conditions (strictness of constraints) we prove in addition that it is decidable whether there is an optimal strategy in which case an optimal strategy can be computed. Our results extend previous decidability result which requires the underlying game automata to be acyclic. Finally, our results are encoded in a first prototype in HyTech which is applied on a small case-study.

[RS-04-3] *PostScript*, *PDF*, *DVI*.

Mads Sig Ager, Olivier Danvy, and Jan Midtgaard.

A Functional Correspondence between Call-by-Need Evaluators and Lazy Abstract Machines.

February 2004.

17 pp. This report supersedes the earlier BRICS report RS-03-24. Extended version of an article appearing in *Information Processing Letters*, 90(5):223–232, 2004.

Abstract: We bridge the gap between compositional evaluators and abstract machines for the lambda-calculus, using closure conversion, transformation into continuation-passing style, and defunctionalization of continuations. This article is a followup of our article at PPDP 2003, where we consider call by name and call by value. Here, however, we consider call by need.

We derive a lazy abstract machine from an ordinary call-by-need evaluator that threads a heap of updatable cells. In this resulting abstract machine, the continuation fragment for updating a heap cell naturally appears as an 'update marker', an implementation technique that was invented for the Three Instruction Machine and subsequently used to construct lazy variants of Krivine's abstract machine. Tuning the evaluator leads to other implementation techniques such as unboxed values. The correctness of the resulting abstract machines is a corollary of the correctness of the original evaluators and of the program transformations used in the derivation.

[RS-04-2] *PostScript*, *PDF*.

Gerth Stølting Brodal, Rolf Fagerberg, Ulrich Meyer, and Norbert Zeh.

Cache-Oblivious Data Structures and Algorithms for Undirected Breadth-First Search and Shortest Paths.

February 2004.

19 pp. Appears in Hagerup and Katajainen, editors, *9th Scandinavian Workshop on Algorithm Theory*, SWAT '04 Proceedings, LNCS 3111, 2004, pages 480–492.

Abstract: We present improved cache-oblivious data structures and algorithms for breadth-first search (BFS) on undirected graphs and the single-source shortest path (SSSP) problem on undirected graphs with non-negative edge weights. For the SSSP problem, our result closes the performance gap between the currently best *cache-aware* algorithm and the *cache-oblivious* counterpart. Our cache-oblivious SSSP-algorithm takes nearly full advantage of block transfers for *dense* graphs. The algorithm relies on a new data structure, called *bucket heap*, which is the first cache-oblivious priority queue to efficiently support a weak DecreaseKey operation. For the BFS problem, we reduce the number of I/Os for *sparse* graphs by a factor of nearly $\sqrt{B}$, where B is the cache-block size, nearly closing the performance gap between the currently best *cache-aware* and *cache-oblivious* algorithms.

[RS-04-1] *PostScript*, *PDF*, *DVI*.

Luca Aceto, Willem Jan Fokkink, Anna Ingólfsdóttir, and Bas Luttik.

Split-2 Bisimilarity has a Finite Axiomatization over CCS with Hennessy's Merge.

January 2004.

16 pp.

Abstract: This note shows that split-2 bisimulation equivalence (also known as timed equivalence) affords a finite equational axiomatization over the process algebra obtained by adding an auxiliary operation proposed by Hennessy in 1981 to the recursion free fragment of Milner's Calculus of Communicating Systems. Thus the addition of a single binary operation, viz. Hennessy's merge, is sufficient for the finite equational axiomatization of parallel composition modulo this non-interleaving equivalence. This result is in sharp contrast to a theorem previously obtained by the same authors to the effect that the same language is not finitely based modulo bisimulation equivalence.