Proceedings of the
6th Nordic Workshop on Programming Theory
(Aarhus, Denmark, 17–19 October, 1994)

December 1994

This document is also available as *PostScript*, *DVI*, `Text`.

References

[Hen] *PostScript*, *DVI*.
Matthew Hennessy.
Higher-order processes and their models.
In 6th NWPT, page 1.
vi+483 pp.

Abstract: A higher-order process algebra in which processes can be sent and received as data along channels is investigated. Using a simple operational semantics two behavioural preorders are defined. The first, based on may testing, is in terms of the ability of processes to offer communications on channels while the second, based on must testing, depends on the communications which processes can guarantee.

The first behavioural preorder can be modelled by a denotational semantics which uses a notion of higher-order traces while for the second we develop a denotational model using higher-order Acceptance Trees.

Comments: University of Sussex, School of Math. & Phys. Sciences, Falmer, Brighton BN1 9QH, United Kingdom. Email: `matthewh@cogs.susx.ac.uk`.

[Ste] *PostScript*.
Bernhard Steffen.
Finite model checking and beyond.
In 6th NWPT, pages 2–17.
vi+483 pp.

Abstract: This paper reviews finite and infinite-state model checking form two pragmatic perspectives: the application to the static analysis of imperative programs or data flow analysis, and the implementational aspect. Data flow analysis provides an interesting area of application for model checking, which is particularly challenging in the interprocedural case, since infinite state spaces need to be considered. Moreover, as it is a typical case of a structurally unrestricted problem, one should use global iterative methods for solution. This can be exploited for a uniform and efficient treatment based on a *fixpoint analysis machine* that covers all kinds of logics and model structures important for the considered application.

Comments: Lehrstuhl für Informatik, Universität Passau, Innstr. 33, D-94032 Passau (Germany), tel: +49 851 509.3090, fax: +49 851 509.3092, `steffen@fmi.uni-passau.de`.

[NS] *PostScript*, *DVI.*
Dave S. Neilson and Ib Holm Sørensen.
The B-Technologies: A system for computer aided programming.
In 6th NWPT, pages 18–35.
vi+483 pp.

Abstract: The paper introduces the B-Technologies, a mathematically based formal method and a tool-set for computer aided software engineering.

The *B-Technologies* (comprising three components - the *B-Method*, the *B-Tool* and the *B-Toolkit*) have been designed to scale up formal methods for practical application. The *B-Method* and the *B-Toolkit* are described in this paper.

The *B-Method* is designed to provide a notation and a methodology for the formal specification, design, implementation and maintenance of industrial-scale software systems. The features of incremental construction of layered software as well as its incremental verification have been guiding principles in the development of the B-Method. The method uses J-R Abrial's *Abstract Machine Notation* (AMN) as the language for specification, design and implementation within the software process. AMN is is based on an extension of Dijkstra's guarded command notation, with built-in structuring mechanisms for the construction of large systems.

The *B-Toolkit* supports the method over the entire spectrum of activities from specification through design and implementation into maintenance. The B-Toolkit comprises automatic and interactive theorem-proving assistants, a proof printer and a set of software development tools: an AMN syntax & type checker, a specification animator and generators promoting an object oriented approach at all stages of development, and the re-use of specification models/software modules. All tools are integrated with the proof assistants into a window-based development environment.

Comments: B-Core Limited, Magdalen Centre, Oxford Science Park, Oxford OX4 4GA, United Kingdom. Email: `{Dave.Neilson,Ib.Sorensen}@comlab.ox.ac.uk`.

[AHS] *PostScript*, *DVI.*
Hosein Askari, Ole I. Hougaard, and Michael I. Schwartzbach.
A framework for ad-hoc type inference.
In 6th NWPT, pages 36–50.
Extended Abstract.

Abstract: Languages based on variations of the lambda calculus are designed to permit the slick, unification-based technique for type inference, which is by now a well-established discipline.

Other widely used languages have been created less by design and more by coincidence and compromise. It seems therefore that the question of type inference for such languages could be infeasible or should at least permit only ad-hoc solutions.

In this paper we argue the existence of a uniform framework for developing type inference algorithms, even for seemingly ad-hoc languages. This framework provides useful generalizations of well-known concepts

related to type inference. It is, however, by no means a universal algorithm. In fact, the essential algorithmic problem must generally be solved from scratch.

We demonstrate the viability of our approach by developing a type inference algorithm for a full version of the Turbo Pascal language.

Comments: BRICS, Department of Computer Science, University of Aarhus, 8000 Aarhus C, Denmark.

[AI] *PostScript, DVI.*
Luca Aceto and Anna Ingólfsdóttir.
CPO models for a class of GSOS languages.
In 6th NWPT, pages 51–66.
vi+483 pp.

Abstract: In this paper, we present a general way of giving denotational semantics to a class of languages equipped with an operational semantics that fits the GSOS format of Bloom, Istrail and Meyer. The canonical model used for this purpose will be Abramsky's domain of synchronization trees, and the denotational semantics automatically generated by our methods will be guaranteed to be fully abstract with respect to the finitely observable part of the bisimulation preorder. In the process of establishing the full abstraction result, we also obtain several general results on the bisimulation preorder (including a complete axiomatization for it), and give a novel operational interpretation of GSOS languages.

Comments: BRICS, Department of Mathematics and Computer Science, Aalborg University, Denmark.

[AK] *PostScript.*
Parosh Aziz Abdullah and Mats Kindahl.
On decidability of simulation and bisimulation for lossy channel systems.
In 6th NWPT, pages 67–77.
vi+483 pp.

Abstract: By using an infinite-state system consisting of finite-state processes communicating via unbounded FIFO channels it is possible to model link protocols like the Alternating Bit protocol , or the HDLC protocol. It is well known that most interesting verification problems are undecidable for this class of systems. We have previously shown that by altering the behaviour of the channels so that they become *lossy*, several verification problems become decidable. In this paper we develop algorithms for deciding strong bisimulation equivalence and simulation preorder. Furthermore, we show that weak bisimulation equivalence is undecidable.

Comments: Uppsala, Sweden.

[AM] *PostScript, DVI,*
Relevant papers.
Henrik Reif Andersen and Michael Mendler.
PMC: A process algebra for real-time systems.
In 6th NWPT, pages 78–79.
vi+483 pp.

Abstract: Most timed process algebras view a real-time system as operating under the regime of a global time parameter constraining the occurrence of actions. By the use of quantitative timing constraints they aim at describing completely the global real-time behaviour of timed systems in a fairly detailed fashion. Based on an industrial case study we believe that these approaches are often overly realistic with disadvantages for both the specification and the modelling of real-time systems. We propose a rather different, abstract approach to the specification and modelling of real-time systems that captures the *qualitative* aspects of timing constraints through the use of *multiple clocks*. Clocks enforce global synchronization of actions without compromising the abstractness of time by referring to a concrete time domain. Technically, we present the process algebra PMC as a non-trivial extension of CCS by *multiple clocks* with associated *timeout* and *clock ignore* operators.

We describe the object of investigation in the industrial case study, a highly sophisticated instrument – the Brüel & Kjær 2145 Frequency Analyzer – and show how the central real-time constraints are expressed concisely in PMC. Focus is on the actual specification of the instrument and the technical results that have been obtained for PMC is only touched briefly.

Comments: Department of Computer Science, Technical University of Denmark, Building 344, DK-2800 Lyngby, Denmark. Email: {`hra,mvm`}`@id.dtu.dk`.

[ANN] *PostScript*, *DVI*.
Torben Amtoft, Flemming Nielson, and Hanne Riis Nielson.
Type and behaviour reconstruction for higher-order concurrent programs.
In 6th NWPT, pages 80–95.
vi+483 pp.

Abstract: In this paper we develop a sound and complete type and behaviour inference algorithm for a fragment of CML (Standard ML with primitives for concurrency). Behaviours resemble terms of a process algebra and yield a concise presentation of the communications taking place during execution; types are mostly as usual except that function types and "delayed communication types" are labelled by behaviours expressing the communications that will eventually take place. The development of the present paper improves a previously published algorithm in not only achieving a sound algorithm for type inference but also one that is complete, due to an alternative strategy for generalising over types and behaviours. Although we show how to solve special kinds of constraints the problem of a general solution procedure for the constraints generated remains; this is related to an open problem concerning whether all programs typable in Standard ML (with concurrency primitives) are also typable in our system.

Comments: {`tamtoft,fn,hrn`}`@daimi.aau.dk`.

[BB] *PostScript*, *DVI*.
Ralph-Johan Back and Michael J. Butler.
Applications of summation and product operators in the refinement calculus.
In 6th NWPT, pages 96–111.
vi+483 pp.

Abstract: Product and summation operators for predicate transformers were introduced by Naumann and by Martin using category theoretic considerations. In this paper, we formalise these operators in the higher order logic approach to the refinement calculus, look at various algebraic properties of these operators, and examine several of their applications. We look at how the product operator provides a model of simultaneous execution of statements, while the summation operator provides a simple model of late binding. We also generalise

the product operator slightly to form an operator that corresponds to conjunction of specifications. We show how the product operator may be used to model extension and modification operators for programs, and how a combination of the product and summation operators may be used to model inheritance in an object-oriented programming language.

Comments: Dept. of Computer Science, Åbo Akademi, Finland.

[CH] *PostScript*, *DVI*.
Vytautas Čyras and Magne Haveraaen.
Programming with data dependencies: a comparison of two approaches.
In 6th NWPT, pages 112–126.
Some proofs are omitted due to lack of space.

Abstract: We present two methods for expressing computations based on recurrence relations and discuss their relative merits. One method, the *structural blanks* approach, is built on top of traditional programming languages like Fortran or Pascal. The other method, the *constructive recursive* approach, is based on recursive relations over graphs.

Comments: Vilnius University, Lithuania and University of Bergen, Norway.

[Cer] *PostScript*, *DVI*.
Kārlis Čerāns.
A calculus of timed refinement.
In 6th NWPT, pages 127–141.
vi+483 pp.

Abstract: This paper presents CTR - a process algebraic framework for loose specification of time quantity sensitive operational behaviour of reactive systems.

Comments: Institute of Mathematics and Computer Science, University of Latvia, Rainis blvd. 29, Riga, Latvia. Email: `karlis@mii.lu.lv`.

[Che] *PostScript*.
Allan Cheng.
Petri nets, traces, and local model checking.
In 6th NWPT, pages 142–156.
vi+483 pp.

Abstract: It has been observed that the behavioural view of concurrent systems that all possible sequences of actions are relevant is too generous; Not all sequences should be considered as likely behaviours. By taking progress fairness assumptions into account one obtains a more realistic behavioural view of the systems. In this paper we consider the problem of performing model checking relative to this behavioural view. We present a *CTL*-like logic which is interpreted over the model of concurrent systems labeled 1-safe nets. It turns out that Mazurkiewicz trace theory provides a useful setting in which the progress fairness assumptions can be formalized in a natural way. We provide the first, to our knowledge, set of sound and complete tableau rules for a *CTL*-like logic interpreted under progress fairness assumptions.

Comments: BRICS, Department of Computer Science, University of Aarhus, Ny Munkegade, DK-8000 Aarhus C, Denmark. Email: `acheng@daimi.aau.dk`.

[Coq] *PostScript*, *DVI*.
Catarina Coquand.

Introduction to ALF — a background.
In 6th NWPT, page 157.
vi+483 pp.

Abstract: We will give an introduction to type theory, focussing on its use as a logical framework for proofs and programs. We will also discuss the use of inductive definition as specifications and proofs by pattern matching. Many examples will be presented.

The basic idea behind using type theory for developing proofs and programs is the Curry-Howard isomorphism between propositions and types. Take for example the proposition (A implies B implies A). Looking at this proposition as a type we see that this is the type of the K combinator. In type theory we then say that the K combinator proofs the proposition (A implies B implies A).

The language that is used is a functional language with dependent types. Hence programs and proofs can be described in the same language. We use inductive definitions for our specifications. Introduction rules in logic corresponds to defining a type by its constructors. Proofs are functions defined by pattern matching over these types. We shall also mention how this can be extended to proofs about infinite objects.

Comments: Chalmers Technical University, Inst. for Computer Science, Göteborg, Sweden. E-mail: `catarina@cs.chalmers.se`.

[Fal] *PostScript*, *DVI*.
Göran Falkman.
Program separation as a basis for definitional higher order programming.
In 6th NWPT, pages 158–172.
vi+483 pp.

Abstract: We describe a program separation scheme based on a notion of form and content of an algorithm. We show that this type of program separation is one way of making algorithms an explicit part of declarative programming. We also show how this type of program separation can be used as a basis for definitional higher order programming.

Comments: Department of Computing Science, Chalmers University of Technology, S-412 96 Göteborg, Sweden.

[Fre] *PostScript*, *DVI*.
Øyvind Bolme Fredriksen.
General algebras.
In 6th NWPT, pages 173–187.
vi+483 pp.

Abstract: We study categories of algebras (for a given signature) constructed from arbitrary categories C with specific finite products. We show how a left adjoint $Set \rightarrow C$ may be utilized to construct an initial algebra satisfying a given set of equations. Finally, we give a condition under which satisfaction of equations is independent of the set of variables considered.

Comments: University of Bergen, Department of Informatics, Høyteknologisenteret, N–5020 BERGEN, NORWAY.

[HHK] *PostScript*, *DVI*.
Martin Hansen, Hans Hüttel, and Josva Kleist.
Bisimulations for asynchronous mobile processes.
In 6th NWPT, pages 188–189.

vi+483 pp.

Abstract: In this paper we consider Plain LAL, a mobile process calculus which differs from the π-calculus in the sense that the communication of data values happens *asynchronously*. Previously, Honda and Tokoro, and Boudol have described so-called asynchronous π-calculi. However, they have not studied the notion of equivalence within this setting. The surprising result is that in the presence of asynchrony, the open, late and early bisimulation equivalences *coincide*. As the equivalences are one and the same, this leads us to formulate an *equational theory* which is sound for finite terms of Plain LAL.

Comments: BRICS, Department of Mathematics and Computer Science, Aalborg University, Fredrik Bajersvej 7E, 9220 Aalborg Ø, Denmark. Email: `hans@iesd.auc.dk`.

[HK] *PostScript*, *DVI*.
Claus Hintermeier and Hélène Kirchner.
G^2-algebras.
In 6th NWPT, pages 190–191.
vi+483 pp.

Abstract: G^2-algebras are polymorphic, order-sorted algebras where sorts are terms in an equational theory. G^2-algebras are still first order and have the classical quotient term algebra as initial model. They extend G-algebras conservatively and are close to a two level, hierarchical fragment of unified algebra. The presented framework has meanwhile been superseded by an n-level Horn clause approach called G^n-logics, having a sound and complete deduction system and initial models.

Comments: CRIN-CNRS & INRIA-Lorraine, BP239, F-54506 Vandœuvre-lès-Nancy Cedex, France. Email: {`hinterme,hkirchne`}`@loria.fr`.

[HP] *PostScript*, *DVI*.
Leszek Holenderski and Axel Poigné.
Boolean automata: A compact representation of synchronous reactive systems.
In 6th NWPT, pages 192–193.
vi+483 pp.

Abstract: Boolean automata are compact representations of synchronous reactive systems. The compactness manifests in avoiding the well-known "state explosion" problem which arises from representing parallel composition by a standard product construction.

In the computational model for synchronous reactive systems, time is assumed to be *discrete*, i.e. divided into enumerable number of non-overlapping segments, called *instances of time*. In every instance of time, a synchronous reactive system performs a so-called *reaction step*, which consists in fetching inputs, computing, and emitting outputs. The fetch-compute-emit sequence is considered to be an atomic action which takes exactly one instance of time. A system may consist of several reactive components, running in parallel and communicating with each other. The components perform their reaction steps simultaneously, i.e. in the same instance of time, and the outputs emitted in one component are immediately available to all other components. Thus, communication is based on instantaneous broadcasting, used in the synchronous programming languages Esterel, Lustre and Argos, as opposed to instantaneous hand-shaking, used in synchronous CCS.

In fact, we only consider *pure* synchronous reactive systems, i.e. those in which all inputs and outputs are pure signals. A pure signal does not carry a value. It is either present or absent, in every instance of time.

It is convenient to represent a *pure* synchronous reactive system (with signals S), by a Mealy automaton whose transitions are labelled by pairs e/e', where $e, e' \subset S$. A transition $sigma \xrightarrow{e/e'} \sigma'$, where σ and σ' are states, describes a possible reaction step of the system. It has the following intuitive reading: provided the system starts an instance in state σ and e is the set of all signals present during the instance, the system emits signals e' and finishes the instance in state σ'. In such a setting, parallel composition of reactive systems is represented by a product of Mealy automata, which leads to the well-known "state explosion" problem.

It turns out that the transition relation of a pure synchronous reactive system can be coded by two sets of Boolean functions. The first one represents a set of (possibly recursive) Boolean equations whose solutions represent pairs e/e'. The second one represents a set of simultaneous Boolean assignments which represent changes of states.

In the paper, we show how to compose such Boolean automata using typical operations, like sequential and parallel composition, choice, iteration and preemption. The main result is that a Boolean automaton grows linearly with the growth of a reactive synchronous system it represents.

Comments: GMD-SET, Schloss Birlinghoven, D-53757 Sankt Augustin, Germany.

[Har] *PostScript.*

Mait Harf.

Structural synthesis of programs using regular data structures.

In 6th NWPT, pages 194–202.

vi+483 pp.

Abstract: This paper discusses a specific feature of structural synthesis of programs. Using regular data structures enables to optimize the method of structural synthesis of programs for several particular cases. The method for program construction described here is implemented in the NUT programming system.

Comments: Dept. of Computer Software, Institute of Cybernetics, Akadeemia tee 21, EE 0026 Tallinn, Estonia. Email: `mait@cs.ioc.ee`.

[Ing] *PostScript, DVI.*

Anna Ingólfsdóttir.

A semantic theory for value-passing processes, late approach, part I: A denotational model and a complete axiomatization.

In 6th NWPT, pages 203–219.

vi+483 pp.

Abstract: A general class of languages and denotational models for value-passing calculi based on the late semantic approach is defined. A concrete instantiation of the general syntax is given. This is a modification of the standard CCS according to the late approach. A denotational model for the concrete language is given, an instantiation of the general class. An equationally based proof system is defined and shown to be sound and complete with respect to the model.

Comments: BRICS, Institute for Electronic Systems, Department of

Mathematics and Computer Science, Aalborg University, Denmark.

[Jen] *PostScript*, *DVI*.
Claus Torp Jensen.
Interpreting broadcast communication in CCS with priority choice.
In 6th NWPT, pages 220–236.
vi+483 pp.

Abstract: *CBS* (Calculus of Broadcasting Systems) is a process calculus in which *broadcast* is the fundamental communication paradigm. In a *CBS* system every parallel subprocess participates in any action that the system may perform. In that sense *CBS* is a synchronous calculus of the same type as *SCCS*.

CCS on the other hand is an asynchronous calculus. At most two subprocesses participate in any action of a particular parallel system.

We show that by adding a priority choice operator to *CCS*, these two different views of communication and parallellism may be reconciled in the sense that any *CBS* system may be translated to a term in *CCS* with priority choice. A *CBS* synchronization must necessarily correspond to a *sequence* of transitions in the *CCS* model, and the priority choice operator enables us to ensure that such sequences terminate correctly. The translation of *CBS* terms is correct in the sense that two *CBS* terms are strongly bisimilar iff their translations are weakly bisimilar. Due to the multiway synchronizations in *CBS* this is the simplest relation between a term and its translation that we can hope for.

Comments: Computer Science Department, Chalmers University of Technology.

[Jeu] *PostScript*, *DVI*.
Johan Jeuring.
Polytypic programming — abstract.
In 6th NWPT, page 237.
vi+483 pp.

Abstract: In this talk I discuss polytypic functions. I combine the polytypic functions that have been defined in the Bird-Meertens calculus, a calculus for transformational programming developed over the last decade, with inductive definitions of natural transformations to build new polytypic functions.

Comments: Chalmers University of Technology and University of Göteborg, S-412 96 Göteborg, Sweden. Email: `johanj@cs.chalmers.se`.

[LSW] *PostScript*, *DVI*.
Kim G. Larsen, Bernhard Steffen, and Carsten Weise.
A constraint oriented proof methodology based on modal transition systems.
In 6th NWPT, pages 238–250.
vi+483 pp.

Abstract: In this paper, we present a constraint-oriented state-based proof methodology for concurrent software systems which exploits compositionality and abstraction for the reduction of the verification problem under investigation. Formal basis for this methodology are Modal Transition Systems allowing loose state-based specifications, which can be refined by successively adding constraints. Key concepts of our method are *projective views*, *separation of proof obliga-*

tions, *Skolemization* and *abstraction*. The method is even applicable to real time systems.

Comments: Weise: Lehrstuhl für Informatik I, Aachen University of Technology, Germany. Email: `carsten@informatik.rwth-aachen.de`.

[Las] *PostScript*, *DVI.*
Søren Bøgh Lassen.
Reasoning with actions.
In 6th NWPT, pages 251–265.
vi+483 pp.

Abstract: Action semantics is a semantic description framework with very good pragmatic properties but a rather weak theory for reasoning about programs. A strong action theory would be of great practical use, however. It would make it possible to reason about the large class of programming languages that can be described in action semantics.

This paper develops the foundations for a richer action theory, by bringing together concepts and techniques from testing theory for processes and from work on operational reasoning about functional programs. Semantic preorders and equivalences in the action semantics setting are studied and a useful operational technique for establishing testing equivalences is presented.

Comments: BRICS, Department of Computer Science, University of Aarhus, DK-8000 Aarhus C, Denmark.

[MD] *PostScript*, *DVI.*
Karoline Malmkjær and Olivier Danvy.
Preprocessing by program specialization.
In 6th NWPT, pages 266–268.
vi+483 pp.

Abstract: We describe an alternative approach to preprocessing that uses partial evaluation and the properties of a simple two-arguments matching program to determine structures yielding linear matching.

Comments: Computer Science Department, Aarhus University.

[Mag] *PostScript*, *DVI.*
Lena Magnusson.
Introduction to ALF — an interactive proof editor.
In 6th NWPT, page 269.
vi+483 pp.

Abstract: We will give a description of incremental type checking, which is the main tool for doing interactive proof development in ALF.

ALF is based on Martin-Löf's type theory, which is a logical framework in the sense that the language can be extended with new definitions. The language is a functional language with dependent function types, which can be extended with data types, functions and constants as new definitions, in the spirit of traditional functional languages. The dependent function type gives us a richer type system, in which propositions and specifications of programs can be represented. Thus, the relation $e : T$ can denote a program e of type T or a proof e of proposition T. This relation is decidable and the type (proof) checking algorithm is the core of the proof editor ALF. However, the objective of ALF is to interactively build proof terms or programs, in a flexible and safe way. For this purpose we have extended the language with the notion of placeholders (meta variables) which denotes sub-terms intended to be filled in. Hence, an incomplete proof term is a term containing placeholders, and proof refinement corresponds to successively replacing a placeholder by a (possibly incomplete) sub-term.

The type checking algorithm is extended to handle incomplete terms which gives as result a unification problem, i.e. a set of equations and typing restrictions of the placeholders occurring in the incomplete term. This set corresponds to the requirements which must be satisfied in future refinements. The unification problem can not always be solved, due to higher order placeholders, and unsolved equations are left as constraints restricting future instantiations of the placeholders. Correctness of a refinement is established by type checking the refined (incomplete) proof term. Thus, proof refinement is reduced to type checking incomplete proof terms.

Comments: Chalmers Technical University, Inst. for Computer Science, Göteborg, Sweden. E-mail: `lena@cs.chalmers.se`.

[NN] *PostScript*, *DVI*.

Hanne Riis Nielson and Flemming Nielson.

Static and dynamic processor allocation for higher-order concurrent languages.

In 6th NWPT, pages 270–285.

vi+483 pp.

Abstract: Starting from the process algebra for Concurrent ML we develop two program analyses that facilitate the intelligent placement of processes on processors. Both analyses are obtained by augmenting an inference system for counting the number of channels created, the number of input and output operations performed, and the number of processes spawned by the execution of a Concurrent ML program. One analysis provides information useful for making a static decision about processor allocation; to this end it accumulates the communication cost for all processes with the same label. The other analysis provides information useful for making a dynamic decision about processor allocation; to this end it determines the maximum communication cost among processes with the same label. We prove the soundness of the inference system and the two analyses and demonstrate how to implement them; the latter amounts to transforming the syntax-directed inference problems to instances of syntax-free equation solving problems.

Comments: Computer Science Department, Aarhus University, Denmark.

[Olv] *PostScript*, *DVI*.

Peter C. Ølveczky.

Termination of order-sorted rewriting.

In 6th NWPT, pages 286–299.

vi+483 pp.

Abstract: We present a method for proving termination of order-sorted rewrite systems by transforming an order-sorted rewrite system into an unsorted one such that termination of the latter implies termination of the order-sorted system. The method is inspired by ideas of Gnaedig and Ganzinger and contains as special cases the method presented in [Gn92a] and the method that simply ignores sort information.

[Gn92a] *Termination of order-sorted rewriting*. Proc. Algebraic and Logic Programming. Third International Conference, Springer-Verlag, 1992.

Comments: Department of Informatics, University of Oslo, Norway. Email: `peterol@ifi.uio.no`.

[PO] *PostScript*, *DVI.*

Jens Palsberg and Patrick O'Keefe.

A type system equivalent to flow analysis.

In 6th NWPT, pages 300–316.

Extended abstract of a paper in Proc. POPL '95, 22nd Annual SIGPLAN–SIGACT Symposium on Principles of Programming Languages.

Abstract: Flow-based safety analysis of higher-order languages has been studied by Shivers, and Palsberg and Schwartzbach. Open until now is the problem of finding a type system that accepts exactly the same programs as safety analysis.

In this paper we prove that Amadio and Cardelli's type system with subtyping and recursive types accepts the same programs as a certain safety analysis. The proof involves mappings from types to flow information and back. As a result, we obtain an inference algorithm for the type system, thereby solving an open problem.

Comments: Palsberg: BRICS, Department of Computer Science, University of Aarhus, DK-8000 Aarhus C, Denmark. Email: `palsberg@daimi.aau.dk`. O'Keefe: 151 Coolidge Avenue # 211, Watertown, MA 02172, USA. Email: `pmo@world.std.com`.

[PPS] *PostScript*, *DVI.*

Wiesław Pawłowski, Paweł Pączkowski, and Stefan Sokołowski.

Specifying and verifying parametric processes.

In 6th NWPT, pages 317–331.

vi+483 pp.

Abstract: A framework in which processes parametrized with other processes can be specified, defined and verified is introduced. Higher order process-parameters are allowed. The formalism resembles typed lambda calculus built on top of a process algebra, where specifications play the role of types. A proof system for deriving judgements "parametric process meets a specification" is given.

Comments: Institute of Computer Science, Polish Academy of Sciences, Gdańsk, / Department of Computing Science, Chalmers University of Technology, Göteborg, / Institute of Mathematics, University of Gdańsk.

[RSZ] *PostScript*, *DVI.*

Rimvydas Rukšėnas, Kaisa Sere, and Yi Zhao.

On the formal derivation of a FEAL microprocessor.

In 6th NWPT, pages 332–345.

vi+483 pp.

Abstract: We present an outline of a method for *formal* derivation of asynchronous VLSI circuits. The proposed method focuses on transformational style of the design and it uses techniques familiar from the construction of parallel programs. *Refinement calculus* and *action systems* are used as a framework for the design process. As a case study we look at the derivation of an asynchronous encryption/decryption microprocessor.

Comments: Åbo Akademi University, Department of Computer Science, FIN-20520 Turku, Finland, emails: `{rruksena,zyi}@aton.abo.fi`, University of Kuopio, Department of

Computer Science and Applied Mathematics, FIN-70211 Kuopio, Finland, email: `Kaisa.Sere@uku.fi`, fax: +358-71-162595.

[SNN] *PostScript*, *DVI*.
Kirsten Lackner Solberg, Hanne Riis Nielson, and Flemming Nielson.
Strictness and totality analysis.
In 6th NWPT, page 346.
vi+483 pp.
Abstract: We define a novel inference system for strictness and totality analysis for the simply-typed lazy lambda-calculus with constants and fixpoints. Strictness information identifies those terms that definitely denote bottom (i.e. do not evaluate to WHNF) whereas totality information identifies those terms that definitely do not denote bottom (i.e. do evaluate to WHNF). The analysis is presented as an annotated type system allowing conjunctions only at "top-level". We give examples of its use and prove the correctness with respect to a natural-style operational semantics.
Comments: Solberg: Dept. of Math. and Computer Science, Odense University, Denmark. Nielson: Computer Science Dept., Aarhus University, Denmark. Email: {`kls,hrn,fn`}`@daimi.aau.dk`.

[SW] *PostScript*, *DVI*.
K. Sere and Marina Waldén.
Backward refinement for verifying distributed algorithms.
In 6th NWPT, page 347.
vi+483 pp.
Abstract: We present a new verification method for distributed algorithms. The basic idea is that an algorithm to be verified is stepwise transformed into a high level specification through a number of correctness-preserving steps. At each step some mechanism of the algorithm is identified and abstracted away while the basic computation in the original algorithm is preserved. In this way the algorithm becomes more coarse-grained. Only the essential parts of the algorithm are then left for final verification.
Comments: Sere: University of Kuopio, Department of Computer Science and Applied Mathematics, P.O.Box 1627, SF-70211 Kuopio, Finland. Email: `Kaisa.Sere@uku.fi`. Waldén: Åbo Akademi University, Department of Computer Science, SF-20520 Turku, Finland. Email: `mwalden@abo.fi`.

[Sak] *PostScript*, *DVI*.
Jūratė Sakalauskaitė.
Nonclausal resolution system for branching temporal logic.
In 6th NWPT, pages 348–358.
vi+483 pp.
Abstract: We present a proof system for branching propositional temporal logic. The system is based on nonclausal resolution.

The system is proved to be complete. The proof of completeness uses tableau construction for the logic.
Comments: Institute of Mathematics and Informatics, Akademijos 4, 2600 Vilnius, Lithuania.

[Sana] *PostScript*, *DVI*.
David Sands.
Composed reduction systems.
In 6th NWPT, pages 359–376.

vi+483 pp.

Abstract: This paper studies *composed reduction systems*: a system of programs built up from the reduction relations of some reduction system, by means of parallel and sequential composition operators. The trace-based compositional semantics of composed reduction systems is considered, and a new graph-representation is introduced as an alternative basis for the study of compositional semantics, refinement, true concurrency (in the case of composed rewriting systems) and program logics.

Comments: DIKU, University of Copenhagen.

[Sanb] *PostScript*, *DVI*.
David Sands.
Towards operational semantics of contexts in functional languages.
In 6th NWPT, pages 377–384.
vi+483 pp.

Abstract: We consider operational semantics of contexts (terms with holes) in the setting of lazy functional languages, with the aim of providing a balance between operational and compositional reasoning, and a framework for semantics-based program analysis and manipulation.

Comments: DIKU, University of Copenhagen.

[Sas] *PostScript*, *DVI*.
Vladimiro Sassone.
An approach to the category of net computations.
In 6th NWPT, pages 385–399.
To appear in InProceedings of TAPSOFT '95.

Abstract: We introduce the notion of *strong concatenable process* as a refinement of non-sequential (concatenable) processes which can be expressed axiomatically via a *functor* $\mathcal{Q}[\_]$ from the category of Petri nets to an appropriate category of symmetric strict monoidal categories, in the precise sense that, for each net N, the strong concatenable processes of N are isomorphic to the arrows of $\mathcal{Q}[N]$. In addition, we identify a *coreflection* right adjoint to $\mathcal{Q}[\_]$ and characterize its *replete image*, thus yielding an axiomatization of the category of net computations.

Comments: BRICS, Department of Computer Science, University of Aarhus, Ny Munkegade Bld. 540, DK-8000 Aarhus C.

[Tor] *PostScript*, *DVI*.
Olof Torgersson.
Functional logic programming in GCLA.
In 6th NWPT, pages 400–414.
vi+483 pp.

Abstract: We describe a definitional approach to functional logic programming, based on the theory of Partial Inductive Definitions and the Programming Language GCLA. It is shown how functional and logic programming are easily integrated in GCLA using the features of the language, that is combining functions and predicates in programs becomes a matter of programming methodology. We also give a brief description of a way to automatically generate efficient procedural parts to the described definitions.

Comments: Department of Computing Science, Chalmers University of Technology, S-412 96 Göteborg, Sweden. Email: `oloft@cs.chalmers.se`.

[Uus] *PostScript*, *DVI*.
Tarmo Uustalu.
Extensions of structural synthesis of programs.
In 6th NWPT, pages 415–427.
vi+483 pp.
Abstract: Structural synthesis of programs (SSP) is an approach to deductive synthesis of functional programs using types as specifications and based on the Curry-Howard correspondence and on an intensional treatment of the notion of type. The implemented programming environments employing SSP have been based on a fragment of intuitionistic propositional logic (simple type theory) and on a natural-deduction proof system. In the paper, we indicate that the proof search strategy used in these systems is applicable to a variety of natural-deduction proof systems (not necessarily of intuitionistic or propositional logics), and that the object-oriented user front-end specification language of the NUT programming environment can, in fact, be given a useful logical semantics in terms of intuitionistic first-order logic.
Comments: Dept of Teleinformatics, The Royal Inst of Technology, Electrum 204, S-164 40 Kista (Stockholm), Sweden.

[Wei] *PostScript*, *DVI*.
Martin Weichert.
Algebra of broadcasting systems: Value passing, sequential composition, and fork.
In 6th NWPT, pages 428–443.
vi+483 pp.
Abstract: This paper presents ACBS&, Algebra of Broadcasting Systems with Fork, a process calculus with broadcast communication, which combines value passing and sequential composition.

Putting value passing into a calculus with sequential composition turns out to be more complicated than with a prefixing calculus. A semantics for such a calculus can be defined using environments, as in imperative languages.

Parallel processes have parallel environments, which can lead to ambiguity *which* of the environments shall be passed on to the sequentially following process. ACBS& solves this problem by distinguishing a foreground and a background process. The resulting operator can be interpreted as a *fork* operator, which "forks off" a background process.

The calculus is presented both in a "pure" and a value passing version, together with a complete axiomatisation for the pure calculus.
Comments: S-412 96 Göteborg, Sweden.

[vB] *PostScript*, *DVI*.
Franck van Breugel.
From branching to linear metric domains (and back).
In 6th NWPT, pages 444–447.
Abstract.
Abstract: Besides partial orders, also *metric spaces* have turned out to be very useful to give semantics to programming languages (see, e.g., the collection of papers of the Amsterdam Concurrency Group. In the literature, one encounters two main classes of metric domains: *linear domains* and *branching domains*. Linear domains were already studied by topologists in the early twenties. Branching domains have been

introduced by, e.g., De Bakker and Zucker, Golson and Rounds, and the author. The elements of these linear and branching domains are convenient to model—one might even say that they represent—*trace equivalence* classes and *bisimulation equivalence* classes, respectively. The former is a simple observation. The latter has been proved by Van Glabbeek and Rutten.

Linear domains are more abstract than branching domains. Our aim is to show that linear domains can be embedded in branching domains. We focus on the branching domain $\mathcal{B}$ introduced by De Bakker and Zucker in and the linear domain $\mathcal{L}$ the elements of which can be viewed as nonempty and compact sets of sequences.

Comments: McGill University,School of Computer Science, 3480 University Street, Montreal H3A 2A7, Canada.

[AKLN] *PostScript.*
Jørgen H. Andersen, Kåre J. Kristoffersen, Kim G. Larsen, and Jesper Niedermann.
Automatic synthesis of real time systems.
In 6th NWPT, pages 448–464.
vi+483 pp.

Abstract: This paper presents a method for automatically constructing real time systems directly from their specifications. The model–construction problem is considered for implicit specifications of the form:

$$(A_1 \,|\, \ldots \,|\, A_n \,|\, X) \;\mathsf{sat}\; S$$

where S is a real time (logical) specification, $A_1 \ldots A_n$ are given (regular) timed agents and the problem is to decide whether there exists (and if possible exhibit) a real time agent X which when put in parallel with $A_1 \ldots A_n$ will yield a network satisfying S. The method presented proceeds in two steps: first, the implicit specification of X is transformed into an equivalent direct specification of X; second, a model for this direct specification is constructed (if possible) using a direct model construction algorithm. A prototype implementation of our method has been added to the real time verification tool EPSILON.

Comments: BRICS, Department of Math. & Comp. Sc., Aalborg University.

[CM] *PostScript, DVI.*
Catarina Coquand and Lena Magnusson.
Demonstration of ALF.
In 6th NWPT, page 465.
vi+483 pp.

Abstract: We refer to our abstracts for further information about the demonstrated system.

Comments: Chalmers Technical University, Inst. for Computer Science, Göteborg, Sweden.

[SFC+] *PostScript.*
Bernhard Steffen, B. Freitag, A. Claßen, Tiziana Margaria, and U. Zukowski.
Intelligent software synthesis in the DaCapo environment.
In 6th NWPT, pages 466–481.
vi+483 pp.

Abstract: The DaCapo software synthesis environment presented in this

paper supports the *automatic construction of software* from existing components according to a (possibly incomplete) specification. Based on a specification language that uniformly combines taxonomic component specifications, interface conditions, and ordering constraints, our method adds a global view to conventional single component retrieval. Together with its semi-automatic archiving facility for the 'retrievable' inclusion of a satisfactory new software component into the software repository, our systems supports a full software lifecycle, as we illustrate along a user session.

Comments: Fakultät für Mathematik und Informatik, Universität Passau, D-94030 Passau (Germany), `steffen@fmi.uni-passau.de`.

[Sor] *PostScript, DVI.*
Ib Holm Sørensen.
Demonstration of the B-Toolkit.
In 6th NWPT, pages 482–483.
vi+483 pp.

Abstract: The B-Toolkit is a suite of integrated programs which implement the B-Method for Software Development. The B-Method is a collection of formal techniques which give a basis to those activities of Software Development that range from technical software specification, through design and integration, to code generation and into maintenance. The B-Method and the specification language AMN (Abstract Machine Notation) are in many respects similar to other "model oriented" formal methods. They employ a conventional "pseudo" programming style. The B-Tool is a language interpreter for the B Theory Language. This language is a special purpose language for writing interactive and automatic proof assistants and other systems where pattern matching, substitution and re-write mechanisms can be used. The B-Toolkit's component tools are implemented in the B Theory Language and is interpreted by the B-Tool.

Comments: B-Core Limited, Magdalen Centre, Oxford Science Park, Oxford OX4 4GA, United Kingdom. Email: `Ib.Sorensen@comlab.ox.ac.uk`.

[NS-94-6] Uffe H. Engberg, Kim G. Larsen, and Peter D. Mosses, editors.
Proceedings of the 6th Nordic Workshop on Programming Theory (Aarhus, Denmark, 17–19 October, 1994), December 1994.
vi+483 pp.

Abstract: The Nordic Workshop on Programming Theory brings together researchers from the Nordic and Baltic countries, in order to improve mutual contacts and collaboration. The invited speakers, however, generally come from non-Nordic/Baltic countries. The 6th Nordic Workshop attracted 63 participants. The workshop had three invited talks and 41 submitted talks. This proceedings contains full papers or extended abstracts of the talks. For completeness we have included short abstracts for the few remaining talks.