Technical Reports 1991 – 1995

University of Manchester
Department of Computer Science

December 5, 1995

Reports with numbers shown in **bold** below are available by anonymous ftp from `ftp.cs.man.ac.uk` in directory `pub/TR`, or using the WorldWide Web via URL

`http://www.cs.man.ac.uk/csonly/cstechrep/index.html`

UMCS-95-11-1 A Highly Restricted Temporal Logic with a Tractable Decision Procedure
Ian Pratt

This paper demonstrates the tractability of a highly restricted temporal logic, into which English sentences containing a variety of temporal prepositional constructions can be translated. This paper is intended as a preliminary study for a larger project of identifying tractable temporal logics corresponding to temporal constructions in natural language. This larger project also includes current work exploring the automatic translation of English sentences involving temporal prepositions (and related adverbials) into temporal logics.

Technical Report

UMCS-95-10-3 Enhancing the Tractability of Rely/Guarantee Specifications in the Development of Interfering Operations
P. Collette and C. B. Jones

Various forms of assumption/commitment specifications have been used to specify and reason about the interference that comes from concurrent execution; in particular, consistent and complete proof rules relating to shared state operation specifications with rely and guarantee conditions have been published elsewhere. This report investigates methodological issues in the formulation of such specifications, and their way to record design decisions. This work aims at making the use of rely/guarantee conditions more tractable, both at the specification level and in the development towards code.

Technical Report

UMCS-95-10-2 Designing C-elements for Testability
O. A. Petlin and S. B. Furber

C-elements are used widely in asynchronous VLSI circuits. Fabrication faults in some CMOS C-elements can be undetectable by logic testing. Testable designs of static symmetric and asymmetric C-elements are given in this report which provide for the detection of single line stuck-at and stuck-open faults. We show that driving feedback transistors in the proposed testable static C-elements transforms their sequential functions into combinational AND and OR functions or into repeaters of one of their inputs depending on the driving logic value. This simplifies the testing of asynchronous circuits which incorporate a large number of state holding elements. The scan testable C-element described can be used in scan testing of the asynchronous circuit making the states of its memory elements controllable and observable.

Technical Report

UMCS-95-10-1 Intra-Modular Structuring in Model-Oriented Specification: Expressing Non-Interference with Read and Write Frames
Juan C. Bicarregui

Compositionality provides the key to managing complexity in software systems and thus should be sought at all levels in the design process. In the development of code from model-oriented specifications, compositionality can not only be achieved by the decomposition of system specifications into modules but may also arise within the development of individual modules.

This thesis considers the compositional development of operations that share state within modules. A key mechanism in the structuring of such developments lies in the definition of non-interference between sub-operations. The read and write frames in VDM implicit operations definitions can be understood as constraints on the read and write accesses which can be made by valid implementations. Their use as such permits reasoning about non-interference between operations and can thus be a source of compositionality within modules. However, existing semantic models of model-oriented specification and refinement do not provide a basis for such an interpretation of read and write frames.

Investigations are made into alternative denotational semantic models for operation specifications which do support read and write frames and a proof theory for reasoning about non-interference between sub-operations is founded upon them. The proppoсed semantic models provide progressively finer distinctions between implementations over and above the standard model. Thus new possibilities arise for discrimination between specifications that would otherwise be considered equivalent, whilst existing results remain valid.

Technical Report

UMCS-95-8-1 Research Interactions Between University and Industry in Computer Science in the United States and United Kingdom
T. Haigh

The academic discipline of computer science owes its existence to the industrial development of computer technology; the computer industry owes its existence to work undertaken by academic researchers. Close links have always existed between the two, and ideas, equipment and personnel have flowed between them.

The funding of academic research by industry, and the exploitation of research results by universities, are now the subject of more activity and interest than ever before. This report examines the current state of relations between universities and industry in computer science, with particular reference to the characteristics of different kinds of formal and informal relationship, such as research sponsorship, consultancy agreements and technology licensing, and the advantages and disadvantages which each potentially offers to participants. Considerable attention is paid to the special characteristics of computer technology, computer science research and the computer industry which make certain kinds of interaction more productive and widespread than others.

Policies regarding ownership of intellectual property, the establishment of start-up companies and other controversial topics are examined. The role of personal relationships and tacit knowledge is scrutinised, as are the different roles given to formal university bodies in regulating what are essentially relationships between individuals. Where significant, differences between observed activities in the UK and US are analysed.

Primary source material is a series of 27 in depth interviews undertaken with a range of professionals working in university departments and research teams, central university offices and corporate managerial capacities in both the US and the UK. These are supplemented by references to a range of research literature and by a concise account of the historical and political background to current patterns of research support.

Technical Report

UMCS-95-7-2 The Algebraic Theory of Interaction Nets
R. Banach

The theory of interaction nets, invented by Lafont, is re-examined from the algebraic hypergraph rewriting perspective. Supersimple nets are defined and discussed, and some related classes of nets, the polysimple and monosimple classes, are defined and investigated. Their static properties are established, and the invariants that need to be preserved by rewriting are investigated in detail. It is shown that in the general case, context-specific information may be used to ensure that rules actually preserve the characteristics of the rewritten net. Subordinate agents, which like logical constants for falsity may be introduced only in non-void contexts, are presented, and the ramifications of the theory in their presence are investigated, relating it to the simple and semisimple classes of Lafont. Under suitable conditions, describable in purely combinatorial terms, net rewriting systems possess Church-Rosser and Strong Normalisation properties usually associated with rewriting systems derived from logic. This leads to the reformulation of the proof nets of multiplicative linear logic including constants in an essentially logic-free way.

Technical Report

UMCS-95-7-1 The State Evolution Method for Verifying Hardware Systems
Howard Barringer, Graham Gough, Brian Monahan and Alan Williams

We present a novel state evolution method for establishing standard (strong) bisimulation, which gives a tractable verification approach for deterministic machines, possibly with infinite state-spaces, and operates at an abstract level. The problem of establishing equivalence is reduced to one of proving the validity of a set of simpler (first-order) logical verification conditions, generated from the state evolution expressions. The approach maintains a high degree of automation, a feature of state-based methods, whilst offering the potential of containing the usual growth in complexity of verification, one advantage of using theorem-proving techniques.

Technical Report

UMCS-95-4-2 The Evolution of Polygenic Sex Determination with Potential for Environmental Manipulation
M. J. Hatcher and C. Tofts

We present a formal account of a genetic mechanism for Environmental Sex Determination (ESD) using a probabilistic process algebra (WSCCS). We exploit the process algebra to demonstrate that a single chromosome pair based polyfactorial system is intrinsically unstable. The system evolves, with a deterministic relationship between an animal's sex and its polyfactor count, to three possible forms of genetic mechanism :pure ESD; pure genetic sex determination (GSD) (both XY and ZW being possible outcomes); or a mixed solution with a 50/50 division between ESD and GSD. With a probabilistic dependence between gene count and sex we observe the first and last of the above solutions (no pure GSD), but with population gene frequencies that can be far from Hardy-Weinberg equilibrium. A model of the systemic behaviour in the presence of recombination is presented, and the effect of the relative recombination and evolution rates is discussed. Our process algebraic model is extended to account for the polyfactors being distributed over many possible chromosome pairs and demonstrates that only one essentially new stable solution is reachable. We demonstrate that within a multichromosome system there will almost always be a 'leading' chromosome pair which determines sex, other chromosome pairs will have identical sex determining information upon both members of the pair, but not necessarily none.

Technical Report

UMCS-95-4-1 Formal reuse of hardware design
A. Melo

The complexity of hardware components has grown rapidly over the past two decades, with the advances in technologies for manufacturing integrated circuits. Also, the widely spread application of hardware components has required them to be competitive to the market. These factors have led to a requirement for new methods and techniques to enhance controllability, quality and productivity of hardware components. Reusability has been recognized as a basic principle for enhancing productivity and quality of engineering products. Additionally, formal development of hardware has emerged as a method to ensure quality and help handle the complexity of description of hardware components. Based on these premises, this thesis introduces a technique for formal reuse of hardware design.

The main idea of this project is to create a 'context' in which an existing hardware component must be embedded in order to implement a new desired component. A solution for effective reuse of hardware components depends on mechanisms to compare the behaviour of these elements, where their similarities and dissimilarities are recognized. So, ways to relate components from a reuse point of view are introduced and, based on those, the necessary 'context' is formally constructed.

A process algebra is used for the behavioural representation of hardware, and as a foundation for formally reasoning about the reuse of hardware design. Algebraic manipulation of processes allows for managing the representation of complex components. In order to permit reuse of complex components, the 'context' is constructed via a 'decomposition operator' defined for the particular process algebra. To obtain minimal 'contexts', mechanisms for minimization of concurrent processes, in terms of the number of states and communications, are finally presented.

The contribution of this thesis is a basis for algebraic manipulation of reusable components, and a mechanism to create minimal 'contexts' for formal reuse of hardware components.

Ph.D. Thesis

UMCS-95-3-1 Analytic and locally approximate solutions to properties of probabilistic processes
C. Tofts

Recent extensions to process algebra can be used to describe performance or error rate properties of systems. We examine how properties of systems expressed in these algebras can be elicited. Particular attention is given to the ability to describe the behaviour of system components parametrically. We present how analytic formulae for performance properties can be derived from probabilistic process algebraic descriptions. Demonstrating how local approximate solutions can be derived for the properties when their exact solutions would be too computationally expensive to evaluate. As an example we derive the performance of an Alternating Bit protocol with respect to its error and retry rates.

Technical Report

UMCS-95-1-1 A Proof System for a sequential Object-Based Language
C. C. de Figueiredo

This thesis presents a proof system to support the formal verification of the correctness of sequential object-based programs written in a simple programming language named A1. The proof system uses a specification language, named L1, that is based on Hoare-style assertions.

The syntax and formal semantics of the programming and specification languages are given. A formal semantics of correctness formulas (for partial correctness) is then defined. Axiom schemes and proof rules associated with commands of A1, which compose a proof system named P1, are presented. Examples are shown where these axiom schemes and proof rules are used to derive program correctness. The proof system P1 is shown to be sound according to the given semantics of A1 and of the correctness formulas.

Finally, conclusions are drawn and suggestions for further work are made.
This thesis presents a proof system to support the formal verification of the correctness of sequential object-based programs written in a simple programming language named A1. The proof system uses a specification language, named L1, that is based on Hoare-style assertions.

The syntax and formal semantics of the programming and specification languages are given. A formal semantics of correctness formulas (for partial correctness) is then defined. Axiom schemes and proof rules associated with commands of A1, which compose a proof system named P1, are presented. Examples are shown where these axiom schemes and proof rules are used to derive program correctness. The proof system P1 is shown to be sound according to the given semantics of A1 and of the correctness formulas.

Finally, conclusions are drawn and suggestions for further work are made.

Technical Report

UMCS-94-12-3 Variable Delay Timing Analysis of Logic Circuits
B. Bani-Eqbal

We propose a new simulation algorithm for combinational circuits which accurately models delay variations in the gates. The algorithm can also be used for the timing verification of asynchronous circuits, e.g. hazard detection, race/oscillation of sequential circuits, etc. The basic idea is to assign a time variable to each signal transition (edge). The gate delays translate to algebraic relations between the variables. As the arrival order of the edges to a gate can be uncertain, so the simulation result of the gate can diverge. Branching signal paths are introduced to deal with this problem. However the simulation process later often does not depend on earlier branching. The algorithm takes advantage of such cases. We believe that this leads to considerable efficiency improvements.

Technical Report

UMCS-94-12-2 The ELLA Verification Environment: A Tutorial Introduction
Howard Barringer, Graham Gough, Brian Monahan and Alan Williams

The ELLA Verification Environment (EVE) offers both familiar CAD tool support for the ELLA hardware description language, and formal analysis tools, integrated within the one system, accessed by a common window-based graphical user interface. The aim of the system is to enable hardware design engineers to benefit from the use of formal techniques in the design cycle by embedding them into existing technology, and providing interaction with them at the designer level. The formal analysis tools are also designed to operate mainly automatically, significantly reducing the need for formal methods expertise during design and analysis. The EVE is implemented in Common Lisp and CLOS using the LispWorks Environment (Harlequin Ltd. (Cambridge).

This tutorial provides a brief overview of the EVE, and then describes the use of each component tool in detail. Several small design examples are used as illustration, and also act as reader exercises. It also includes background information on suggested tool usage, and a brief introduction to the novel symbolic verification approach used by the analysis tools.

This report was produced by Manchester Informatics Ltd under contract for DRA (Malvern).

Technical Report

UMCS-94-12-1 A Process Algebra Foundation for Reasoning about Core ELLA
Howard Barringer, Graham Gough, Brian Monahan and Alan Williams

A process algebraic foundation is developed, for formal analysis of synchronous hardware designs using the commercially available hardware design language, ELLA. An underlying semantic foundation, based on input/output trace sets, is presented first through the use of state machines. Such a representation enables direct application of standard, fully automated, trace equivalence checking tools. However, to overcome the computational limitations imposed by such analysis methods, the input/output trace semantics is re-presented through a synchronous process algebra, EPA. Primitive processes in EPA denote the behaviour of primitive hardware components, such as delays or multiplexers, with composition operators corresponding to the different ways in which behaviours may be built. Of particular significance is the parallel composition operator which captures the machinery for building networks from other components/networks. Actions in EPA are structured and signify the state of input and output signals. This structure, however, is abstracted by developing an algebra for the actions. In particular, parallel composition on processes neatly lifts to a special (synchronous) product operation on actions.

The EPA representation forms a good basis for semi-automated high-level symbolic manipulation and reasoning tools. Firstly, the original design structure can be maintained, thus easing the problems of user level feedback from tools. Secondly, the application of EPA to ELLA enables a normal form for EPA terms. This provides a route to tractable symbolic verification and simulation, using a state evolution method to establish strong bisimulation properties. The method has been successfully applied to classes of unbounded state space systems.

Technical Report

UMCS-94-11-1 Autonomous acquisition of sensor-motor couplings in robots
U. Nehmzow

Fixed robot controllers are suitable for tasks whose main characteristics are known a priori. However, for tasks such as exploration of unknown territory, fixed controllers tend to be too brittle as they rely on predefined knowledge, supplied by the designer. For such tasks (i.e. tasks in which a-priori knowledge is limited) we propose to use self-organising controllers which allow the robot to acquire the necessary competences autonomously.

The paper describes experiments with mobile robots, in which the robots autonomously determine effective connections between their sensory input and their motor response to it.

In these experiments, artificial neural networks are used to store associations between sensory input and motor responses, the networks are trained in a self-supervised way, making use of so-called instinct-rules which govern the robots' behaviour. The fast learning achieved enables the robots to adapt to changing circumstances such as changes in their environment or their morphology (e.g. sensor failure).

Technical Report

UMCS-94-9-2 Constraint Logic Planning
Y. Zhang and H. Barringer

Deductive planning provides a rigorous approach for planning, such that the capabilities and limitations of the planning systems can be clearly defined. However, most of the deductive planners suffer from the frame problem and they are linear planners. This report presents a novel nonlinear deductive planning framework: constraint logic planning. Procedurally speaking, nonlinear planning is an activity of constraint solving. Constraint logic planning is a combination of the constraint solving for nonlinear planning with a temporal logic paradigm. In comparison with other state based deductive planning methods, constraint logic planning deals with the frame problem efficiently in a way similar to that in classical nonlinear planning, and constraint logic planning is a nonlinear deductive planning approach, it employs the least commitment strategy for both temporal relation decision and object selection. In comparison with classical nonlinear planning systems, constraint logic planning is a deductive planning approach and it achieves completeness, soundness and conflict freeness. Constraint logic planning is believed to be even more efficient than classical nonlinear planning.

Technical Report

UMCS-94-9-1 Action Constraint Maintenance
Y. Zhang and H. Barringer

This report presents a novel approach to detect conflicts in nonlinear plans. Conventional methods detect conflicts in a nonlinear plan by checking that if preconditions of actions in the plan are possibly deleted by some other actions. Our novel method, action constraint maintenance, detects conflicts in a nonlinear plan by checking that if the nonlinear plan satisfies a set of specific action constraints. We prove that, under certain conditions, action constraint maintenance is sound and complete, and we show that action constraint maintenance is much more efficient than conventional methods. The most significant advantage of this method is that, if a nonlinear plan is not consistent with an action constraint, all its possible expansions will not be consistent with the action constraint. Therefore, action constraint maintenance can be employed to detect conflicts in a nonlinear plan during the plan construction, i.e. "on the fly", and backtracking from inconsistencies will not lose possible solutions. By adopting action constraint maintenance "on the fly" in our deductive nonlinear planning framework, we avoid computation for modal truth criteria in nonlinear planning, and we can achieve and maintain the necessary truth for all the goals involved in a planning process. Also, action constraint maintenance itself is a domain independent search heuristic in nonlinear planning.

Technical Report

UMCS-94-8-1 Domain Constraint Maintenance
Y. Zhang, H. Barringer and D. Carlisle

In this report, we propose a set of domain independent and human independent restrictions on actions and we show that the restrictions can prevent actions from being improperly defined. Then, based on the set of restrictions, we propose a new conflict detecting method for nonlinear plans and we show that this new method is sound and complete.

Technical Report

UMCS-94-7-1 A Reified Temporal Logic for Nonlinear Planning
Y. Zhang and H. Barringer

Deductive planning treats planning as a reasoning process within a formal logic. Several formal logics have been applied in various deductive planning systems, but none is completely satisfactory and widely accepted by planning researchers. Almost all the formal logics applied in deductive planning are proposed for more general purpose than planning. Usually, the more general and the more expressive a language is, the less efficient that language is for dealing with problems in a specific domain. In this report, we propose a temporal logic specifically for representing and reasoning about nonlinear planning. Our temporal logic naturally captures the persistence assumption and, when applied in deductive planning, can deal with the frame problem efficiently. Also, in this temporal logic, planning problem can be concisely and clearly specified. Directly upon this temporal logic, we build a deductive nonlinear planning framework, constraint logic planning, which is proved to be sound, complete and is showed to be efficient. In this report, we also briefly describe how the temporal logic provides the basis for constraint logic planning.

Technical Report

UMCS-94-6-1 MathPIP: A Mathematica Interface for PIP
Zbigniew Chamski

The Parametric Integer Programming (PIP) algorithm due to Feautrier is highly suitable for solving linear programming problems arising in program parallelization and synthesis. In this report, we describe the interface allowing to use the arbitrary-precision arithmetic implementation of PIP as an external problem solver in MATHEMATICA, thus providing a convenient means for fast experimentation with algorithms which use PIP as a building block. The first part of the report offers a short overview of PIP, followed by an example-based user's guide to MathPIP. The second part is a reference manual describing in detail each user-visible component of MathPIP.

Technical Report

UMCS-94-5-1 The Uflow Computational Model and Intermediate Format
J. Sargeant, C. Kirkham and S. Anderson

This report motivates and defines a general-purpose, architecture independent, parallel computational model, which captures the intuitions which underlie the design of the United Functions and Objects programming language. The model has two aspects, which turn out to be a traditional dataflow model and an actor-like model, with a very simple interface between the two. Certain aspects of the model, particularly strictness, maximum parallelism, lack of suspension, and the implications of introducing stateful objects, are stressed. The model is embodied in a textual intermediate format, and in a set of UFO data structures. This report also serves as a definition of the intermediate format, and gives a brief overview of the data structures.

Technical Report

UMCS-94-4-3 A Comparative Study of Data-Flow Architectures
David F. Snelling and Gregory K. Egan

Partly as a result of the diversity of the architectures and system software, Data-Flow systems have not been compared directly. In this paper, three widely different Data-Flow systems are compared using a relatively uniform metric which is representative of the actual amount "work" performed by these systems to execute a small collection of common benchmarks. One feature, common to all Data-Flow systems, forms the basis for a metric for "work", the creation of tokens. By counting the creation of tokens in all parts of a Data-Flow system, an accurate measure of the work performed can be obtained. The three systems compared are the Manchester Data-Flow Machine, the Stateless Data-Flow Architecture (also from Manchester), and the CSIRAC II machine. All machines perform approximately the same amount of work when solving a collection of benchmark problems, but two systems (CSIRAC II and SDFA) exhibit successful exploitation of a memory hierarchy.

Technical Report

UMCS-94-4-2 Heuristic reasoning for an automatic commonsense understanding of logic electronic design specifications
Salvador Mir

An automatic heuristic understanding of digital electronic design specifications is discussed in this thesis. The term understanding is used in the sense that knowledge about the functionality and purpose of the cells and signals of a design is abstracted away from the specification. An heuristic analysis which exploits implicit design semantics is carried out. The analysis bypasses the examination of detailed logical and electrical data since it is aimed at the machine simulation of an heuristic way of understanding electronic design specifications exhibited by human experts. By overlooking implicit knowledge, current automatic systems are clearly at a disadvantage with respect to human experts for the analysis, design and management of electronic data. The possibility of getting the machine to heuristically understand a specification is seen in this thesis as one way of improving this situation. The thesis defines and classifies expert knowledge about digital electronic designs, explores ways to generate expert knowledge about a design from the heuristic analysis of its specification and discusses examples of exploiting this knowledge to plan and control automatic tasks. The experimental result of this research is a knowledge-based system aimed at providing an empirical demonstration of the convenience and viability of an automatic heuristic understanding of design specifications. The method of reasoning of the system has without question limitations, but these are also faced by human experts when they attempt this task. The current prototype of the system already indicates that valid knowledge can be generated and it implements methods for avoiding the critical computational complexity of the problem.

Ph.D. Thesis

UMCS-94-4-1 The Global-yet-Personal Information System
J. R. Gurd and C. B. Jones

The edition of this volume set out to compile a set of personal views about the long-term direction of computer science research. In responding to this goal, we have chosen to identify what we perceive as a long-term challenge to the capabilities of computing technology in serving the broader needs of people and society, and to discuss how this 'Grand Challenge' might be met by future research. We also present our personal view of the research methodology. Much of present-day computer technology is concerned with the processing, storage and communication of digital data. The view taken in this contribution is that a far more important use of computers and computing is to manage and manipulate human-related information. Currently, this has been tackled at the level of single organisations (company, institution, government department, etc.) by the use of databases which are often limited to single functions within the organisation. Databases are closed, in the sense that the information itself can be viewed in a limited number of ways, and the ways in which it can evolve are carefully controlled. Interaction between databases containing related data is prohibited, except through the mediation of human experts. This is an unnecessarily restricted concept of information processing,and one which fails to recognise its real social and economic potential. We foresee a huge market for personal information services based on open access to a continually evolving global network of stored information. Although there are significant technical difficulties associated with creating and controlling such networks and services, we predict that the economic inventives will ensure that the necessary development occurs and that this information market will - within a period of decades - dwarf the market in computing machinery and software. With regard to the research methodology, we speculate that computer science is evolving into the more general discipline of informatics and consider the nature of informatics as an emerging science.

Technical Report

UMCS-94-2-2 Accelerated Ray Tracing on the KSR1
M. J. Keates and R. J. Hubbold

This report describes the results of experiments with accelerated ray tracing on a virtual shared-memory parallel computer. The machine used, the Kendall Square Research KSR1, is described in sufficient detail for its novel features to be appreciated. The report describes a new ray tracer written specifically to capitalise on the KSR1's features. Models used in the experiments include well-known test cases from the Haines database, permitting comparison with previously reported results. The results show a speed-up which is 76.9 percent of linear scaling on 256 processors. The report analyses the performance of the software and draws conclusions about the suitability of this type of machine for image synthesis.

Technical Report

UMCS-94-2-1 A Memory Model for Case Retrieval by Activation Passing
Michael Brown

This thesis is concerned with the development of an under-lying model of memory to support selective case retrieval for case-based reasoning. The major requirements are that retrieval should be highly flexible yet efficient. The traditional approach of "indexing" is rejected as being too restrictive while more flexible approaches in analogical reasoning are generally too comutationally expensive. Several important organisational principles are developed in the memory model. A network representation is advocated with a number of required extensions; such as multi-granular representation, context-based segregation and a statistically-based grading of paths. The organisation of memory offers the potential for the serial performance of a number of retrieval tasks that have previously only been addressed by assuming a massively parallel implementation of the memory model supports the feasibility of the technique. A meaningful grading of stored cases is consistently achieved without a reliance on an exhaustive search of memory. Evidence that the memory model will scale to use on large-scale casebases is also presented.

Ph.D. Thesis

UMCS-93-10-2 From types to dataflow: code analysis for an OO language
Andrew J. Barnard

Of all the object-oriented programming languages and systems to appear over the past decade, Smalltalk-80 has remained one of the more popular 'pure' object-oriented languages. However Smalltalk-80 has suffered criticism due to its relatively poor run-time efficiency, together with the lack of compile-time feedback for the programmer. In the past, type systems have been nominated as potential solutions to both these criticisms. The aim of this thesis is to compare and contrast what a type inference system may offer a language like Smalltalk-80, with what is possible from an alternative form of code analysis, namely dataflow analysis. To this end, a type inference system and a dataflow analysis system are both formally defined for a Smalltalk-like language, and their relative properties examined. While both systems are revealed to be capable of detecting potential 'does-not-understand' errors, the type inference system is shown to conflict with the semantics of the original untyped language. In contrast, the dataflow analysis system is shown to have no such problems. In addition, the dataflow analysis system is shown to be capable of extending the possibilities of analysing control structures beyond what is feasible with a type inference system.

Ph.D. Thesis

UMCS-93-10-1 Process-Algebraic Foundations for an Object-Based Design Notation
Cliff B. Jones

Earlier papers give examples of the development of concurrent programs using *POBL* which is a design notation employing concepts from object-oriented programming languages. Use is made of constraints on the object graphs to limit interference and assertions over such graphs to reason about interference. This report merges (and corrects minor inconsistencies between) two papers which document the semantics of *POBL* by providing a mapping to the pi-calculus and indicate how arguments about the design notation might be based on that semantics. It also reflects some recent work not cited in the earlier papers.

Technical Report

UMCS-93-9-2 A Graph-Based Approach to Resolution in Temporal Logic
Clare Dixon

An approach to the mechanisation of temporal logics, based on a form of clausal resolution has been developed. Temporal formulae incorporating both past-time and future-time temporal operators are converted into Separated Normal Form, and then both non-temporal and temporal resolution rules are applied. The temporal resolution rule attempts to match conditions which must be eventually satisfied with sets of formulae which together imply that the condition will never be satisfied. In this dissertation the sets of formulae which are used in the application of the temporal resolution rule are identified using graph theoretic techniques. This is achieved by first extracting formulae or combinations of formulae that could possibly stop a condition being satisfied and then using them to build a graph. Secondly any loops identified during the construction of the graph will mean that the condition can never be satisfied, and are used in the application of the temporal resolution rule thus deriving further formulae. This temporal resolution method has been combined with sub-programs performing translation to normal form and non-temporal resolution to produce an integrated resolution based temporal theorem prover. The main algorithm cycles through these three procedures until false is derived, (the original formula was satisfiable).

Technical Report

UMCS-93-9-1 Trends in Operating Systems Towards Dynamic User-level Policy Provision
K. R. Mayes

It is possible to distinguish between policy and mechanism in operating system design. There is a trend to move policy out of the operating system kernel and into the user-level. This trend is described with respect to example operating system types. A system is proposed which takes this policy/mechanism split to the extreme of having the operating system kernel reduced to a hardware object which provides a low-level but abstract view of the actual hardware. Such a system would be flexible enough to allow investigation of dynamic, user-level, provision of policy.

Technical Report

UMCS-93-8-3 Cut-free Sequent and Tableau Systems for Propositional Diodorean Modal Logics
R. Gore

We present sound, (weakly) complete and cut-free tableau systems for the propositional normal modal logics **S4.3**, **S4.3.1** and **S4.14**. When the modality "box" is given a temporal interpretation, these logics respectively model time as a linear dense sequence of points; as a linear discrete sequence of points; and as a branching tree where each branch is a linear discrete sequence of points.

Although cut-free, the last two systems do not possess the subformula property. But for any given finite set of formulae X the "superformulae" involved are always bounded by a finite set of formulae depending only on X and the logic **L**. Thus each system gives a nondeterministic decision procedure for the logic in question. The completeness proofs yield deterministic decision procedures for each logic because each proof is constructive.

Each tableau system has a cut-free sequent analogue proving that Gentzen's cut-elimination theorem holds for these latter systems. The techniques are due to Hintikka and Rautenberg.

Technical Report

UMCS-93-8-1 Understanding the differences between VDM and Z
I. J. Hayes, C. B. Jones and J. E. Nicholls

This paper attempts to provide an understanding of the interesting differences between two well-known specification languages. The main ideas are presented in the form of a discussion. This was partly prompted by Lakatos' book 'Proof and Refutations' but since this paper is less profound, characters from the childrens' television series 'The Magic Roundabout' are the speakers: Zebedee speaks for Z, Dougal puts the VDM position, and Florence acts as the user. The specifications which are presented have been made similar so as to afford comparison - in neither the VDM nor the Z case would they be considered to be ideal presentations. Some technical details are relegated to footnotes.

Technical Report

UMCS-93-7-2 The Design and Analysis of a Stateless Data-Flow Architecture
David F. Snelling

Data-Flow computing is approximately 20 years old, and progress in that time has been steady but not dramatic. As new technologies have emerged they have been incorporated into new Data-Flow systems, however, Data-Flow has held onto many of its original tenets. One of these tenets, contrary to the Data-Flow ideal, is the use of explicit state for storing data structures, arrays in particular. By imposing a shared memory model this explicit notion of state creates a fundamental barrier preventing traditional Data-Flow systems from being scalable. In this thesis, a dynamic, tagged token Data-Flow architecture, SDFA (Stateless Data-Flow ARchitecture), is presented which diverges from other systems in that it does not support an explicit notion of state. In the absence of explicit state, it is possible to exploit the advantages of Data-Flow, such as uniform load balancing, latency hiding, and good performance on small problems, within a distributed memory architecture. With the shared memory barrier removed, the SDFA system is scalable. The thesis sets out the case for a stateless model, develops an architecture based on this approach, presents a detailed, simulation based analysis of the system,and proposes further research into stateless Data-Flow computing.

Ph.D. Thesis

UMCS-93-7-1 A Data Partioning Algorithm for Distributed Memory Compilation
Michael O'Boyle

This paper proposes a compiler strategy for mapping FORTRAN programs onto distributed memory computers. Once the available parallelism has been identified, it has to be scheduled so as to minimise overhead. The minimisation of different costs will suggest different data and computation partitions. This is further complicated, as the effectiveness of the partition will depend on later compiler optimisations. For this reason, partitioning is at the crux point of compilation and has led several researchers to propose that it be left to the user. it is our contention that it is possible to automatically determine a good data partition and furthermore to make use of the analysis in later optimisation stages. This paper describes an automatic data partition algorithm which is based on four different analysis techniques. By determining the relative merit of each form of analysis, a data partitioning decision is made. By integrating this algorthim into an overall compilation strategy, effective use can be made of any data partitioning decision. To test the strategy, it has been applied to a real non-trivial program, TRED2. The resulting program was ran on 32 cell KSR-1 where this strategy was compared with the native parallelising compiler and a hand coded version of the program. It is shown that the performance of the approach described in this paper is comparable to that of hand-coded techniques.

Technical Report

UMCS-93-6-8 Generalisation for Induction
Sunil Vadera

Proof by induction plays a central role in showing that recursive programs satisfy their specification. Sometimes a key step is to generalise a lemma so thatits inductive proof is easier. This report examines existing heurristics for generalisation for induction. We first examine the applicability of heuristics for generalisation and then develop a new aproach for a class of problems for which the existing methods fail. Unlike the existing methods, we give the conditions under which our method guarantees a valid generalisation.

Technical Report

UMCS-93-6-7 Proof by Analogy in Mural
Sunil Vadera

Conducting proofs by hand, however, can be time consuming to the extent that designers have to judge whether a proof of a particular obligation is worth conducting. Even if hand proofs are worth conducting, how do we know that they are correct? One approach to overcoming this problem is to use an automatic theorem proving system to develop and check our proofs. However, in order to enable present day theorem provers to check proofs, one has to conduct them in much more detail than hand proofs. Carrying out more detailed proofs is of course more time consuming. This report describes the use of proof by analogy in an attempt to reduce the time spent on proofs. We develop and implement a proof follower based on analogy. We present several examples to illustrate the characteristics of our follower. The examples we give include situations in which the follower is unable to complete a proof, as well as when it completes a proof. The incomplete examples illustrate that the follower's failure can provide a hint that enables the user to complete the proof. We conclude our study by comparing our approach with other work.

Technical Report

UMCS-93-6-6 Recursive Models of General Inductive Types
Yuxi Fu

We give an interpretation of Martin-Lof's type theory (with universes) extended with generalized inductive types. The model is an extension of the recursive model given by Beeson. By restricting our attention to PER model, we show that the strictness of positivity condition in the definition of generalized inductive types can be dropped. It therefore gives an interpretation of general inductive types in Martin-Lof's type theory.

Technical Report

UMCS-93-6-5 Understanding Inductive Types in Constructions
Yuxi Fu

In this paper we extend the Calculus of Construction with generalized inductive types. The extension is justified by showing that the usual set theoretical model can be effectivized. It is also pointed out that the model given in a published paper for a collection of inductive types in a different style is wrong.

Technical Report

UMCS-93-6-4 Encodings in Polymorphism, Revisited
Yuxi Fu

We consider encodings in polymorphism with finite product types. These encodings are given in terms of I-algebras. They have the property that all canonical terms (ground terms) are normal terms. We transplant the proof of a well-known result to our setting and show why weak recursion is admissible. We also show how to carry out the dual encodings using the existential quantifier.

Technical Report

UMCS-93-6-3 Categorical Properties of Logical Frameworks
Yuxi Fu

In this paper we give a new presentation of ELF which is well-suited for semantic analysis. We introduce the notions of internal codability, internal definability, internal typed calculi and frame languages. These notions are central to our perspective of logical frameworks. We will argue that a logical framework is a typed calculus which formalizes the relationship between internal typed languages and frame languages. In the second half of the paper, we demonstrate the advantage of our logical framework by showing some categorical properties of it and of encodings in it. By doing so we hope to indicate a sensible model theory of encodings.

Technical Report

UMCS-93-6-2 The Glueing Construction and Lax Limits (with applications to categories of structured posets)
Harold Simmons

Starting life as a way of reconstructing a topological space from a pair of complementary subspaces, the glueing construction has found employment in a wide range of different roles from the construction of free distributive lattices to a supporting part in the 2-categorical analysis of types theories. In this latter role the construction appears to be a fundamental factor in the behaviour of higher order proof theory. What is going on here? Before that can be answered we need at least a less ad hoc description of the contruction. In this paper I set down what is, I believe, the beginnings of a coherent account of the algebraic version of glueing. As well as the abstract theory I give a good selection of different examples to illutrate the diverse nature of the uses of the construction.

Technical Report

UMCS-93-6-1 A Formal Description of the ISPW-6 Software Process Example
Jin Sa and Brian C. Warboys

Recently, software process modelling has attracted an increasing research interest. For the purpose of comparing and contrasting different approaches to software process modelling, a common problem, refereed to as the ISPW-6 software process example,was defined. Many solutions to this example have been provided by different people using different approaches. In this report, a solution to the ISPW-6 software process example by an approach, called BM, is presented. The suitability of BM for software process modelling is assessed against the criteria proposed at the sixth International Software Process Workshop.

Technical Report

UMCS-93-3-2 Semi-analytic Tableaux for Propositional Modal Logics of Nonmonotonicity
Rajeev Gore

The propositional monotonic modal logics **K45**, **K45D**, **S4.2**, **S4R** and **S4F** elegantly capture the semantics of many current *nonmonotonic* formalisms as long as (strong) deducibility of A from a theory G, allows the use of necessitation on the members of G. This is usually forbidden in modal logic where G is required to be empty, resulting in a weaker notion of deducibility.

Recently, Marek, Schwarz and Truszczinski have given algorithms to compute the stable expansions of a *finite* theory G in various such nonmonotonic formalisms. Their algorithms assume the existence of procedures for deciding (strong) deducibility in these monotonic modal logics and consequently such decision procedures are important for automating nonmonotonic deduction.

We first give a sound, (weakly) complete and cut-free, semi-analytic tableau calculus for monotonic **S4R**, thus extending the cut elimination results of Schwarz for monotonic **K45** and **K45D**. We then give sound and complete semi-analytic tableau calculi for monotonic **K45**, **K45D**, **S4.2** and **S4F** by adding an (analytic) cut rule. The proofs of tableau completeness yield a deterministic satisfiability test to determine theoremhood (weak deducibility), because all proofs are constructive. The techniques are due to Hintikka and Rautenberg. We then show that the tableau calculi extend trivially to handle (strong) deducibility for *finite* G.

Using a general theorem due to Rautenberg we also obtain the (weak) interpolation theorem for **K45**, **K45D**, **S4.2** and **S4R**.

Technical Report

UMCS-93-3-1 Implementing State Machines in Smalltalk
Trevor P. Hopkins

Implementing software solutions using finite state machines (FSMs) is a useful technique in many application areas, including compiler implementation and network protocols. This report examines possible implementation techniques for FSMs in the Smalltalk object-oriented system, and identifies techniques which best support the goals of generality, flexibility and software reuse.

Technical Report

UMCS-93-2-3 A Framework for Experimental Analysis of Parallel Computing
J. R. Gurd, M. D. Cooper, G. A. Hedayat, A. Nisbet, M. F. P. O'Boyle, D. F. Snelling and A. P. W. Bohm

A framework is presented for experimental study of the relationships between different approaches to large-scale computational problem-solving on parallel computers. An initial series of experiments is introduced, and preliminary results are presented.

The major areas of concern are introduced by defining five major levels of abstraction, and by describing the application development cycle by means of which useful parallel programs are constructed. At present, attempts to use parallel computers are based on a limited number of hardware-oriented computational paradigms, in which some form of parallel hardware is chosen first, and this determines, by default, the programming and algorithmic tools that can be applied. The framework for experimental analysis of parallel computing is based on a more general, "top-down" view of the process of "mapping" applications onto (parallel) hardware. In this framework, the selection of "moves" in the "mapping space" is informed by an understanding of the costs and benefits associated with each option.

A taxonomy of parallel hardware systems is outlined and a model of parallel hardware that unifies treatment of the various groups in the taxonomy is introduced. Four state-of-the-art parallel hardware systems are described in the terms of this model. A general performance analysis experiment is introduced, in which a number of programs, targeted at various parallel computers, are developed to solve a pair of computational problems. The problems, one an arbitrary test and the other an artificial "control" case, are defined, and specific algorithms for the solution of each are presented. Parallel programs which implement these algorithms are described briefly: These are tailored to the four hardware systems and use appropriate high-level programming languages. Finally, measurement techniques are discussed, and specific hardware-level performance characterstics are introduced, based on the unifying hardware model. Preliminary results for one performance analysis experiment are presented.

Technical Report

UMCS-93-2-2 A Multimedia Information System with Automatic Content Retrieval
Michael Hugh O'Docherty

Traditional databases use simple data types - chiefly numbers and strings - to represent information such as payroll records or scientific data. Queries are commonly expressed in relational calculus or algebra and the results require little more than a textual terminal for presentation. The natural successor to the traditional database is the Multimedia Information System which also uses richer data types - images,text, sound and so on - to describe some application domain. This calls for more advanced technologies such as graphical workstations for data creation and presentation, high-bandwidth networks for fast access to distributed data and large repositories for the storage of objects. These technologies are now widely available. Ideally, queries put to a Multimedia Information System should refer to the content of stored objects - content retrieval - and results should contain complexas well as simple data types. Manual entry of content descriptions is time-consuming, error-prone and subjective; a better approach is to draw on the emerging fields of image interpretation, text interpretation, speech interpretation and knowledge representation to provide automatic content retrieval. This thesis serves as an introduction to Multimedia Information Systems and related fields and describes in detail a pilot system with automatic content retrieval built by the Multimedia Group at Manchester University; the author has been instrumental in the design and implementation of this system, especially those components that cater for the gereration and querying of content descriptions.

M.Sc. Thesis

UMCS-93-2-1 A Process Algebraic Semantics for Core ELLA
H. Barringer, G. D. Gough, B. Q. Monahan and A. Williams

We give a presentation of a trace-based semantics of Core ELLA, a covering subset of the commercial hardware description language ELLA, using a synchronous process algebra called ELLA PROCESS ALGEBRA (EPA). This report is a semantics reference manual for ELLA, giving a detailed definition of the behaviour of ELLA expressions on a construct by construct basis.

EPA is an instantiation of a generic Process Algebra Framework (PAF). It supports parameterised process calls, and has highly structured actions representing pairs of input/output maps labelling its transitions. Value expressions belonging to a rich ELLA action algebra can appear as value-passing arguments to process calls, or in the input/output maps. The EPA process terms reprsenting ELLA are constructed compositionally following the structure of ELLA expressions. Primitive terms, corresponding to atomic operations in ELLA, are assembled using a small set of process operators. In particular the parallel composition operator models synchronous communication between components and is effected by a special product operation on actions. The well-formedness of ELLA expressions is also defined, and acts as a pre-condition for the dynamic semantics.

The EPA terms resulting from the compilation of ELLA expressions can be translated into a special normal form, which is essentially a symbolic state machine representation. This forms the basis for semi-automated high-level symbolic manipulation and reasoning tools.

The work has been conducted as part of a collaborative project sponsored by the DTI and SERC, involving Manchester, DRA (Malvern) and Harlequin Ltd. (Cambridge), with GEC Plessey Semiconductors (Swindon) as evaluators. The aim of the project has been to provide formal verification support tools which are amenable to the industrial hardware designer. Using the formal semantics as a basis, software tools for ELLA compilation, normalisation, (ground-term/symbolic) simulation, (enumerated/symbolic) verification and ELLA-to-ELLA rewriting have been implemented by the project partners, and embedded into an integrated ELLA Design and Verification Environment.

Technical Report

UMCS-93-1-7 The Expressive Power of the English Temporal Preposition System
Ian Pratt and David Bree

In this report, we present an account of the semantics of the English temporal prepositions. The approach taken here stresses the limited range of expressive possibilities afforded by these words, and pays particular attention to the irregular interactions they exhibit with verb tense and aspect. A framework is developed within which the semantics of English temporal expressions can be couched, allowing the temporal prepositions to be grouped according to their 'temporal function'. A key finding of the report is that the eighteen temporal prepositions considered here express, between them, at most six distinct temporal functions.

Technical Report

UMCS-93-1-6 Program and Data Transformations for Efficient Execution on Distributed Memory Architectures
Michael F.P.O'Boyle

This report is concerned with the efficient execution of array computation on Distributed Memory Architectures by applying compiler-directed program and data transformations. By translating a sub-set of a single-assignment language, Sisal, into a linear algebraic framework it is possible to transform a program so as to reduce load imbalance and non-local memory access. A new test is presented which allows the construction of transformations to reduce load imbalance. By a new expression of data alignment, transformations to reduce non-local access are derived. A new pre-fetching procedure, which prevents redundant non-local accesses, is presented and forms the basis of a new data partitioning methodology. By applying these transformations in a straightforward manner to some well known scientific programs, it is shown that this approach is competitive with hand-crafted methods.

Ph.D. Thesis

UMCS-93-1-5 Numerical Linear Algebra and Computer Architecture: An Evolving Interaction
G. A. Hedayat

The aim of this report is to highlight the evolving interaction between computer architecture on the one hand and the design and analysis of algorithms for numerical linear algebra on the other. This report can best be described as a collection of pointers to aspects of the development processes of numerical linear algebra, and to various existing surveys and software. The rich variety of activity in design, analysis, comparison and evaluation of algorithms, new and old, is examined in the context of rapid technological and architectural advances. The characteristics of algorithms for linear algebra are discussed without going into mathematical details: There are references to important surveys that provide such details. Reflections on the practices of numerical linear algebra will, it is hoped, point to possibilities for and limitations of automating the "mapping" of algorithms onto high performance computers.

Technical Report

UMCS-93-1-4 United Functions and Objects: an Overview
John Sargeant

United Functions and Objects (UFO) is a general-purpose, implicitly parallel language designed to allow a wide range of appolications to be efficiently implemented on a wide range of parallel machines while minimising the conceptual difficulties for the programmer. To achieve this, it draws on the experience gained in the functional and object-oriented "worlds" and attempts to bring these worlds together in a harmonious fashion. This report concentrates on examples which illustrate various aspects of UFO, including the various encapsulation and abstraction mechanisms it provides, and the various forms of parallelism which can be exploited.

Technical Report

UMCS-93-1-2 How to eliminate Predicate Variables
H.Simmons

I describe an algorithm which when given a sentence G involving universally quantified predicate variables, first reduces this sentence to normal form G^* and then, if G^* satisfies certain extra restrictions, eliminates the predicate variables of G^* to produce a predicate-free equivalent of G. This process may involve the introduction of skolem functions but in many cases does not (and so produces a more elementary version of G). The algorithm eliminates the predicate variables by witnessing them with certain definable predicates. The choice of these predicates is guided by the shape of the normal form G^*. The algorithm was initially designed to deal with modal correspondence problems. In this respect it is an extension of Sahlqvist's algorithm which produces first order equivalents of a wide class of modal formulas. The algorithm is illustrated by several examples from modal logic.

Technical Report

UMCS-93-1-1 A Density Property of the Primitive Recursive Degrees
Harold Simmons

The standard Ackermann function which is produced by diagonalization to lie outside the class of primitive recursive function. By slowing down the diagonalization I produce a whole continuum of functions which, in complexity, lie above the primitive recursive functions and below the Ackermann function. There is a dense set of these functions which have a rudimentary graph.

Technical Report

UMCS-92-12-4 A Guide to Reading VDM Specifications
Bob Fields

The intention of this document is to give an overview of the syntax of the Specification language of the Vienna Development Method (VDM) as an aid to those who are to some extend familiar with formal notations and have to read or review documents containing VDM. This document is not a tutorial introduction to or motivation for formal methods in general or VDM in particular; nor is it a formal description of what VDM is or means. Both these types of description can be found in the references.

Technical Report

UMCS-92-12-3 idC: A subset of standard C for initial teaching
Ian Cottam

A subset of the programming language ISO Standard C is defined informally. It is claimed that the resulting language (called idC) is suitable for introductory programming classes for university-level students. The claim comes with a caveat that the language requires a certain level of software support not normally mandated (e.g. by the C standard). After justifying each restriction, a sketch of the first prototype implementation is presented. idC may well be a suitable "workhorse" language to introduce concurrently with, or shortly after, a functional lanhuage such as Haskell or SML.

Technical Report

UMCS-92-12-2 Formal methods - selected historical references
C. B. Jones and A. M. McCauley

This report contains citations to papers which are of historical interest in the area of formal approaches to software development.

Technical Report

UMCS-92-12-1 An object-based design method for concurrent programs
C. B. Jones

The property of a (formal) development method which gives the development process the potential for productivity is compositionality. Interference is what makes it difficult to find compositional development methods for concurrent programs. This paper is intended to contribute to tractable development methods for concurrent programs. In particular it explores ways in which object-based language concepts can be used to provide a compositional development method for concurrent programs. This text summarizes results from three draft papers. It firstly shows how object-based concepts can be used to provide a designer with control over interference and proposes a transformational style of development (for systems with limited interference) in which concurrency is introduced only in the final stages of design. The essential idea here is to show that certain object graphs limit interference. Secondly, the paper shows how a suitable logic can be used to reason about those systems where interference plays an essential role. Here again, concepts are used in the design notation which are taken from object-oriented languages since they offer control of granularity and ways of pinpointing interference. Thirdly, the paper outlines the semantics of the design notation mapping its constructs to Milner's pi-calculus.

Technical Report

UMCS-92-11-2 Towards a Formal Framework for Deductive Synthesis of Logic Programs
Kung-Kiu Lau and Mario Ornaghi

In this report, we set up a formal framework for studying deductive synthesis of logic programs. We describe a model for logic program systhesis, where synthesis is a process that extracts a logic program from a first-order specification. A first-order specification is formalised in the framework of a first-order theory, and the atomic relations computed by the synthesised program are new symbols intoduced by definition axioms. Synthesis methods of this kind have been proposed, for example by Hogger and Lau and Prestwich. We also present uniformity results for SLD and SLDNF as another important element of our framework. Finally, using our framework, we give some incompleteness results for deductive synthesis in Peano Arithmetic.

Technical Report

UMCS-92-6-2 Declarative Integration of Object-Oriented Programming and Knowledge Representation
Steven K. Wallis

Knowledge representation is concerned with the representation of real-world knowledge in a computer. Object-oriented programming provides facilities for encapsulation and code reuse. It is argued that their integration on a declarative basis yields significant benefits, including more modular knowledge bases and more reliable programs. This thesis describes a search for declarative object-oriented knowledge. The search begins with RUN, an object-oriented approach to semantic networks. It continues with FOOD, which integrates declarative and object-oriented techniques. The search ends with ROOK, which combines the advantages of RUN and FOOD. ROOK represents object-oriented code and knowledge with the same formalism. An item of declarative object-oriented knowledge is both an item of knowledge and a piece of object-oriented code, and is completely declarative. This integration facilitates the execution of knowledge and reasoning about programs. ROOK supports four knowledge representation paradigms (logic, rules, semantic networks and frames), plus other programming paradigms such as functional programming and constraints.

Ph.D. Thesis

UMCS-92-4-6 A semantics and verification framework for ELLA
H. Barringer, G. Gough, T. Longshaw, B. Monahan, M. Peim and A. Williams

We present an automata-based compositional semantics for an abstraction of the hardware design and description language ELLA. The semantics forms the basis for the automata-theoretic verification of ELLA descriptions, or programs. We further outline the principles of the verification strategy and use a small example, a twisted-ring counter, to demonstrate both the semantics and the verification techniques.

Technical Report

UMCS-92-4-4 The Search for Tractable Ways of Reasoning about Programs
C. B. Jones

This paper traces the most important steps in the history of research on reasoning about programs. The main focus is on ways of verifying that sequential imperative programs satisfy their specifications. Over time it has become clear that post facto verification is only practical for small programs; verification methods which support the development of larger programs exploit a notation of compositionality. Coping with concurrent algorithms is much more challenging – this and other extensions are considered briefly. The main thesis is that the idea of reasoning about programs has been around for a long time; the research has been to find tractable methods.

Technical Report

UMCS-92-4-3 UFO – United Functions and Objects: Draft Language Description
John Sargeant

UFO is a general-purpose, implicitly parallel language designed to allow a wide range of applications to be efficiently implemented on a wide range of parallel machines while minimising the conceptual difficulties for the programmer. To achieve this, it draws on the experience gained in the functional and object-oriented "worlds" and attempts to bring these worlds together in a harmonious fashion.

Technical Report

UMCS-92-4-1 Exclusive normal form of Boolean circuits
B. Bani-Eqbal

Recently, Binary Decision Diagrams (BDDs) have been proposed as an efficient representation of Boolean functions. Although most practical functions have a compact BDD, some useful ones do not. A prime example of these is the integer multiplier function, which was proved to have at least exponential complexity, and to be much worse in practice. We propose a modification of BDDs using the exclusive OR form, and show that it leads to a more compact representation for the multipliers.

Technical Report

UMCS-92-3-2 Knowledge-Based Applications = Knowledge Base + Mappings + Applications
Paul Kwai-Chung Pun

The proliferation of research work carried out in knowledge-based systems shedslight upon issues of knowledge acquisition, manipulation and representation. However, the issue of how this knowledge, once captured and stored in a knowledge base, can be incorporated into and reused in different application programs has not received enough attention. The main difficulty of this issue is the inherent differences between modelling concepts manifested by the objects described in the knowledge base and those manifested by the data structures manipulatedwithin the applications programs. The aim of this thesis is to investigate this issue in a more formal way and to try to propose a systematic approach in which mappings between these two different modelling manifestations can be deduced. In order to study the problem systematically, a formal framework for modelling relationships between objects inside a knowledge base and objects in application data structures is presented. An entity functionalization process is then defined. This provides an abstraction in which different modelling manifestations can be expressed both in terms of sets of entities and sets of entity functions. Two types of mappings, the entity mappings and the entity function mappings, are then deduced based on tree reduction and algebraic isomorphism. Given these mappings, objects and inter-object relationships in a knowlege base can beassociated with the appropriate data structures in application programs.

Ph.D. Thesis

UMCS-92-2-2 Towards a Strictness Analyser for Haskell: Putting Theory into Practice
Julian R. Seward

This thesis examines the design and implementation of a strictness analyser for a small functional programming language. I look at implementation issues neglected by theoreticians and show how a practical analyser can be implemented and incorporated within the larger structure of a compiler.

Previously published implementations have handled non-flat domains and polymorphism badly. This thesis presents some new insights which largely overcome both problems, and which lead to a surprisingly fast implementation. Three key elements of this implementation are the Frontiers representation, Abstraction and Concretisation maps, and appropriate representation of abstract domains. This thesis discusses their roles. The main deficiency of the current analyser is a lack of higher-order analysis. To some extent, previous work shows how to handle this. In time, such analysis may be incorporated.

In the last three or four years, there have been several claims to the development of fast analysers. Upon examination, these often rely either on using insufficiently detailed abstract domains, or work with some ad-hoc algorithm built on dubious mathematical foundations. This analyser, by contrast is based solidly on the theoretical work presented in Geoffrey Burn's Ph.D. thesis "Abstract Interpretation and the Parallel Evaluation of Functional Languages".

M.Sc. Thesis

UMCS-92-2-1 Transformation and Synthesis in METATEM, Part 1: Propositional METATEM
Michael Fisher and Philippe Nöel

In this report, the syntactic manipulation of temporal logic programs is considered. Transformation rules are provided for a temporal programming language that forms part of the METATEM framework for executable temporal logics. Soundness of the various transformations is shown and several applications are given, such as the production of a normal form for programs, which is the basis of both compilation and resolution techniques. The use of transformation techniques in METATEM program synthesis is also considered. Here, a high-level specification, represented as a non-deterministic METATEM program is translated into an 'implementation', which is again represented as a METATEM program.

Technical Report

UMCS-92-1-1 Load Balancing of Parallel Affine Loops by Unimodular Transformations
M. O'Boyle and G. A. Hedayat

This paper is concerned with the automatic mapping of array computation to processors efficiently. One of the major overheads associated with the mapping of computation is load imbalance. An optimising compiler should find a mapping such that this overhead is minimised.

This paper describes formally the mapping of loop iterations to processors so as to minimise load imbalance. The class of perfectly load balanced affine loops is defined whereby using unimodular transformations, it is shown that a large class of loops are equivalent. An algorithm is detailed which can determine both whether a loop structure may be load balanced, and the necessary transformations to do so. This algorithm has a worst case complexity of O(m3) where m is the dimensionality of the iteration space. The analysis is extended to the case when many loops are to be partitioned where it is shown that a transformation may be constructed which simultaneously balances all appropriate loops. Finally it is shown that if a loop is perfectly load balanced, then there exists a transformation such that it may be placed outermost so as to aid partitioning.

Technical Report

UMCS-91-12-5 An Introduction to Categorial Grammars
Mary M. Wood

Categorial grammars come of a long tradition of linguistic description rooted in philosophy of language, logic, and algebra. Like dependency grammars, they look for the constructive patterns of semantic linkage that hold a sentence together, rather than for the analytic patterns by which it can be segmented. They are distinctive for their direct representation of semantics in syntax - which endears them to philosophers of language, and to some linguists - and for their arithmetical transparency - which endears them to logicians, and especially to formal and computational linguists. It is no coincidence that they have finally come to prominence with the dramatic rise of computational linguistics during the last ten years or less.

This introduction outlines first the most important principles and practice of the categorial enterprise. Chapter 2 offers a brief historical introduction. Chapter 3 describes one specific "benchmark" categorial grammar, the "Lambek calculus", and Chapter 4 discusses a range of different proposed extensions to that grammar.

Technical Report

UMCS-91-12-4 A Set Theoretic Semantics for First Order Temporal Logic: definition and Application using Isabelle
P. A. J. Noel

The paper contains a brief description of an exercise in using the generic theorem prover Isabelle to perform the following :

- define the semantics of a temporal logic within Zermelo-Fraenkel (ZF) set theory;
- define a number of inference rules concerning the temporal logic;
- define a tactic which uses some of the derived inference rules in backward proofs to generate automatically a model for propositional temporal formulae.

One advantage of using a theorem prover to define the semantics of a logic within another logic is that (sound) proof systems for the defined logic may be derived from its semantics. Another advantage of this approach is that it allows the derivation of prototype 'programs' (in the form of inference rules under the control of an automatic tactic) correct with respect to the semantics. An example of such a derivation is discussed in this paper.

Technical Report

UMCS-91-12-3 German Temporal Prepositions from an English Perspective
Martin Durrell and David S. Bree

The following is an attempt to analyse the procedures for selecting temporal prepositions (or conjunctions) in German according to the framework developed in Bree et al. (1990). Systematic contrast is made with the English as analysed there (and, where interesting or relevant with Dutch). The German time words follow similar selection trees for those in English and Dutch, but there are some quite interesting differences (and not a few problems), and some amendments had to be made to the analysis in Bree et al. (1990). Despite these differences, there would seem to be no reason in principle why the analytical framework in terms of decision trees adopted by Bree et al. (1990) should not be as revealing for German as for other languages.

Technical Report

UMCS-91-12-2 A Discrete Systematic Model for racing in an SR Latch
W. P. R. Mitchell and A. S. Hilditch

This paper develops a model of propagational delay for digital logic which assumes all voltages to be boolean valued. This allows for calculations on racing which can not normally be done with only boolean valued models. To illustrate this point the paper goes through a particular instance of how a latch performs meta-stably when its set and reset inputs are made 1 simultaneously, and then simultaneously made 0. The extra ingredient which generates the propagational delay is the incorporation of a value for the MOSFET SiO2/Si interface charge. That charge is then modelled as varying in a linear way over time with changes in the gate value.

Technical Report

UMCS-91-12-1 Learning a Grammar
W. Daniel Solomon

A prototype system ('L') for the induction of a lexical grammar from a sample of natural language has been produced. L takes as its input a selection of children's reading books, and induces, or learns, a Categorial Grammar, which takes the form of a lexicon, together with further corpora, to augment its grammar. L also demonstrates the validity of the grammar produced by generating further sentences from that grammar.

Technical Report

UMCS-91-11-2 Modularity in Model-Oriented Formal Specifications and its Interaction with Formal Reasoning
John S Fitzgerald

This thesis concerns the provision of facilities for the modular structuring of model-oriented formal specifications of computing systems.

The basic concepts of model-oriented specification and the need for modular structuring of specifications are reviewed. It is argued that a structuring mechanism should permit formal reasoning to proceed separately for each unit of structure. It is also argued that the provision of structuring facilities should be governed by practical experience.

The body of the thesis applies these principles to the development of a collection of structuring facilities and their application to a model- oriented specification language. A naive initial case study helps determine a basic repertoire of structuring facilities. Structured specifications will consist of definitions of modules. Each module contains an unstructured specification encapsulated by an interface which defines dependencies on, and services supplied to, other modules. Reuse of modules within and between structured specifications is supported. Abstract syntax, context conditions and denotational semantics for this system are proposed. The semantics is presented in terms of theories associated with modules and does not make reference to any particular language for the module contents. A case study using the structuring facilities developed thus far leads to a consideration of their use with a particular model-oriented language. The generation of a theory from a module's contents; the management of the name space due to the other modules in the structure and the use of state-based specifications within modules are considered. A final case study applies the techniques developed in earlier chapters. A concluding chapter assesses the extent to which the definition of structuring facilities described in the thesis has achieved its stated aims. Subjects for further research are suggested.

Ph.D. Thesis

UMCS-91-11-1 System software on the Flagship parallel graph reduction machine
K. R. Mayes, J. A. Keane and S. Holdsworth

The Flagship system was a graph reduction machine having a distributed physical architecture. Although Flagship was firmly in the declarative world, explicit state was supported in the graph reduction model to express the behaviour of the operating system. This state not only had to be isolated from the declarative aspects of the Flagship machine, but also had to be supported with respect to distribution. The mechanism provided for maintaining the consistency of state are discussed with respect to atomic actions at levels in the Flagship system. This approach is used in order to demonstrate how the software environment was supported by the basic execution mechanism of the machine. The nature, construction and creation of system software is discussed with particular reference to optimising access to distributed resources.

Technical Report

UMCS-91-7-2 Specifying Concurrent Object-based Systems using Combined Specification Notations
J. Sa and B. C. Warboys

Although many specification notations for sequential systems have been widely used in practice, most specification notations for concurrent systems, such as temporal logics, are still far too complicated to be used in practice. This report describes a specification method for concurrent object-based systems by utilising simpler notations, such as VDM, and restricted temporal logic notations. In this specification framework, specifications of systems are defined in a temporal logic. However the specifier of a system does not need to use the complicated temporal logic to specify the behaviour of the system, instead, different simpler specification notations are used to specify the different aspects into a formula defined in the temporal logic.

Technical Report

UMCS-91-7-1 An Attempt to Reason about Shared-State Concurrency in the Style of VDM
Ketil Stölen

The paper presents an attempt to develop a totally correct shared-state parallel program in the style of VDM. Programs are specified by tuples of five assertions (P,R,W,G,E). The pre-condition P, the rely-condition R and the wait-condition W describe assumptions about the environment, while the guar-condtion G and the eff-condition E characterise commitments to the implementation. The pre-, rely- and guar-conditions are closely related to the similarly named conditions in Jones' rely/guarantee method, while the eff- condition corresponds to what Jones calls the post-condition. The wait-condition is supposed to characterise the set of states in which it is safe for the implementation to be blocked; in other words, the set of states in which the implementation, when it becomes blocked, eventually will be released by the environment. The implementation is not allowed to be blocked during the execution of an atomic statement. Auxiliary variables are introduced to increase the expressiveness. They are used both as a specification tool; to characterise a program that has not yet been implemented, and as a verification tool; to show that a given algorithm satisfies a specific property. However, although it is possible to define history-variables in this approach, the auxiliary variables may be of any type, and it is up to the user to define the auxiliary structure he prefers. Moreover, the auxiliary structure is only a part of the logic. This means that auxiliary variables do not have to be implemented as if they were ordinary programming variables.

Technical Report

UMCS-91-6-2 Proposal for an Information Model for EDIF
Rachel Lau

This report describes a partial information model of EDIF 2 0 0. First of all, an outline of the purpose of an information model for EDIF is presented. Then, a general overview of the model is given, followed by a more detailed description of the modelling work. The complete dictionaries of terms are given in the following chapters and an index of terms is included at the end of the report. The model itself which consists of a total of 17 schemas can be found in appendices A-Q. The four appendices R-U identify respectively : the point of definition of an EXPRESS object within the schemas; whether or not an EDIF construct has been modelled; the available attributes for each entity; and the relationships in which an object may occur.

Ph.D. Thesis

UMCS-91-6-1 A Method for the Development of Totally Correct Shared-State Parallel Programs
Ketil Stölen

A syntax-directed formal system for the development of totally correct programs with respect to an (unfair) shared-state parallel programming language is proposed. The programming language is basically a while- language extended with parallel- and await-constructs. The system is called LSP (Logic of Specified Programs) and can be seen of as an extension of Jones' rely/guarantee method. His method is strengthened in two respects :

- Specifications are extended with a wait-condition to allow for the development of programs whose correctness depends upon synchronisation.
- Auxiliary variables are introduced to increase expressiveness. They are either used as a specification tool to eliminate undesirable implementations or as a verification tool to prove that a certain program satisfies a particular specification.

Technical Report

UMCS-91-5-1 Interference Resumed
C. B. Jones

Compositional development methods for concurrent (interfering) programs are harder to find than for sequential (isolated) programs. Rely and guarantee conditions extended operation decomposition methods for sequential programs to cover concurrent shared-variable systems: interference was specified in order to achieve compositionality. Ketil St¨{o}len's thesis addresses shortcomings of the rely/guarantee idea: synchronization is specified by a wait-condition and expressiveness is increased by judicious use of auxiliary variables. This paper describes a new step in the quest for compositional development methods for concurrent programs. A collection of temporal operators aimed at natural reasoning about programs is indicated; these operators apply to predicates of two states and a style of specification is proposed which clearly separates transitions of a component from those of its environment. Furthermore a case is made for 'resumptions' as a semantic model for the logic. Since this paper reports on work-in-progress, a list of known problems is given.

Technical Report

UMCS-91-4-3 The HDG-Machine: A Highly Distributed Graph-Reducer for a Transputer Network
Hugh Kingdon, David R. Lester and Geoffrey L. Burn

Distributed implementations of programming languages with implicit parallelism hold out the prospect that the programs are immediately scalable.

This paper presents some of the results of our part of Esprit 415, in which we considered the implementation of lazy functional programming languages on distributed architectures. A compiler and abstract machine were designed to achieve this goal. The abstract parallel machine was formally specified, using Miranda (trade mark of Research Software). Each instruction of the abstract machine was then implemented as a macro in the Transputer Assembler. Although macro expansion of the code results in non-optimal code generation, use of the Miranda specification makes it possible to validate the compiler before the Transputer code is generated. The hardware currently available consists of five T800-25's, each board having 16M bytes of memory. Benchmark timings using this hardware are given. In spite of the straight forward code-generation, the resulting system compares favourably with more sophisticated sequential implementations, such as that of LML.

Technical Report

UMCS-91-4-2 A Performance Assessment of the GSM Radio Link Protocol (RLP)
Brian W. Marsden

his paper presents the results of a simulation evaluation of the Radio Link Protocol (RLP), which is an adaptation of HDLC for use on radio links in cellular networks. The RLP protocol is first described, in particular the ways in which it differs from standard HDLC, in the context of the special characteristics of radio links. The simulation model is then discussed, with particular emphasis on the choice of operational parameters and error recovery policies, which are crucial issues given a radio link which is inherently unreliable and contributes a propagation delay considerably in excess of the time taken to transmit a frame. The results of the study are then presented, with recommendations of suitable operational parameter values and some suggested changes to the RLP specification. The central aim of this work is to determine the combination of operational parameters which will maximise throughput and minimise transit time for a useful range of link error rates.

Technical Report

UMCS-91-4-1 An Algorithm for Planning 'Sensible' Routes
Ian Pratt

The problem of finding an optimal route between two points in a space strewn with obstacles has been extensively studied in robotics. The standard approach to such problems is first to reduce the original geometrical specification to a finite graph of possible routes, and then to use graph-searching techniques together with the metrical information available in the original geometrical specification to select the best of these routes. The algorithm presented here, by contrast, employs a 'first-past-the-post' approach to route planning. Specifically, we show that the algorithm is guaranteed to produce a route satisfying a certain criterion of 'sensibleness', because routes failing this criterion are always slower to get constructed. The approach taken here differs in three respects from most current work on route planning:

- it finds sensible, but not necessarily optimal, routes;
- it is uninfluenced by perspective (though it is limited by occlusion problems);
- it produces its output in a qualitative form, which is nevertheless well-suited to guide the appropriate route- following behaviour.

Technical Report

UMCS-91-3-3 Branching Time and Partial Order in Temporal Logics
Wojciech Penczek

The aim of this report is to present existing formal languages of propositional temporal logic with frames based on branching time structures or, more general, partial orders. Branching time and partial order logics do not only differ in their underlying frames, but also in the way the logics are linked to the behaviour of concurrent systems.

Technical Report

UMCS-91-3-2 Distributed Algorithms for Detecting Distributed Deadlock
Steve Hilditch and Tom Thomson

We present two simple algorithms for the detection of deadlock on a distributed database system. They are designed to be scalable and minimise network message traffic. An object-oriented approach is used and semi-formal proofs of the algorithms' correctness are given.

Technical Report

UMCS-91-3-1 A Distributed Termination Detection Scheme
Xinfeng Ye and John A. Keane

A fully distributed scheme for detecting the termination of distributed computations is proposed. The scheme does not require a pre-defined detector, and takes into account problems such as network delay and the non-order-preserving arrival of messages. The scheme can be applied to any kind of connection topology. The correctness of the scheme is presented in terms of showing that the global stable condition holds when the scheme declares the termination of the computation. The upper bound of the number of the messages which are used to detect termination is also discussed.

Technical Report

UMCS-91-2-2 Operational Semantics for Hardware Design Languages
Howard Barringer, Graham Gough and Brian Monahan

An operational semantics for two hardware design languages, abstracted from Abel and ELLA, are given using Structural Operational Semantics. This can be used to calculate a state machine representation suitable for verification purposes. We show how, despite differences in semantic structure, the languages discussed possess various common features, such as the use of iterative fixpoints to calculate stable states under value propagation. Although hardware designers expect their designs to represent finite state machines, explicit semantics is needed to calculate these state machines in general. This enables construction of a reference simulator against which output of other, more efficient simulators can be compared.

Technical Report

UMCS-91-2-1 Distributed Simulation Using "Relaxed Timing"
Alvaro Garcia Neto

Research in computer architecture relies increasingly in simulation as opposed to prototyping. However, as the size and complexity of the simulated systems increases, "conventional" sequential simulation fails to deliver the necessary performance. We identify the requirement to model time accurately as a main cause for poor simulation performance. We propose a new simulation philosophy, Relaxed Timing, in which the emphasis is on speed of execution rather than correct modelling of time. In this philosophy, modules are entirely event-driven, and consume data as soon as they become available – no synchronisation is done to ensure timing accuracy. Because performance and accurate timemodelling are conflicting requirements, relaxation of temporal precision yields higher performance. This approach clearly leads to preemption errors. We show that, for "self-scheduling" and "well balanced" systems, such as dataflow, these errors are acceptable. We show that a relaxed timing dataflow simulator performs better than a sequential simulator for a large number of algorithms and significant range of input data. Having shown the adequacy of the model, we investigate possible improvements, and find that the insertion of time-priority queues minimizes preemption. By blocking these queues at strategic times, a spectrum of timing precision is possible. We implement and discuss this enhancement. The development language for the simulator is occam. We assess the suitability of occam as a specification language for the generic simulation of computer architectures, and find it adequate. We also discuss the need for tools of higher levels of abstraction to ease the task of simulating distributed parallel systems, and place this need in the context of current research in programming environments, software engineering and program visualization. We describe the available tools in the hierarchy of abstraction levels, and argue that occam is a good candidate to fill an existing gap in this hierarchy.

Ph.D. Thesis

UMCS-91-1-1 Development of Parallel Programs on Shared Data-Structures
Ketil Stölen

A syntax-directed formal system for the development of totally correct programs with respect to an unfair shared-state parallel while- language is proposed. The system can be understood as a compositional reformulation of the Owicki/Gries method for verification of parallel programs. Auxiliary variables may be of any sort, and it is up to the user to define the auxiliary structure he prefers. Moreover, the auxiliary structure is only a part of the logic. This means that auxiliary variables do not have to be implemented as if they were ordinary programming variables. The system is proved sound and relatively complete with respect to an operational semantics and employed to develop three nontrivial algorithms: the Dining-Philosophers, the Bubble-Lattice-Sort and the Set-Partition algorithms. Finally, a related method for the development of (possibly nonterminating) programs with respect to four properties is described. This approach is then used to develop Dekker's algorithm.

Ph.D. Thesis