Report on BCTCS 2013

The 29th British Colloquium for Theoretical Computer Science

24–27 March 2013, University of Bath

Guy McCusker

The British Colloquium for Theoretical Computer Science (BCTCS) is an annual forum in which researchers in Theoretical Computer Science can meet, present research findings, and discuss developments in the field. It also provides an environment for PhD students to gain experience in presenting their work in a wider context, and to benefit from contact with established researchers.

BCTCS 2013 was hosted by the University of Manchester, and held from 24th – 27th March 2013. The event attracted over 49 participants, and featured an interesting and wide-ranging programme of four invited talks and 28 contributed talks, in large part from PhD students, covering the full gamut of topics in theoretical computer science; abstracts of the talks are provided below.

The conference began with an invited talk by Samson Abramsky, University of Oxford, entitled “From Quantum Mechanics to Logic, Databases, Constraints, and Complexity”. Other invited talks were given by Angela Wallenburg, Altran UK (“Proof and test: will they blend?”) and Assia Mahboubi, INRIA–École Polytechnique (“Computer-checked Mathematics”), As in previous years, the London Mathematical Society sponsored a keynote talk in Discrete Mathematics: Susanne Albers, Humboldt-Universität zu Berlin, gave an excellent lecture on “Energy Efficient Algorithms”. The financial support of the London Mathematical Society (LMS) in support of this lecture is gratefully acknowledged. We also acknowledge the financial support of the Heilbronn Institute for Mathematical Research which made available 24 student bursaries to cover full costs of attendance.

Invited Talks at BCTCS 2013

Samson Abramsky, University of Oxford

From Quantum Mechanics to Logic, Databases, Constraints, and Complexity

Quantum Mechanics presents a disturbingly different picture of physical reality to the classical world-view. These non-classical features also offer new resources and possibilities for information processing. At the heart of quantum non-classicality are the phenomena of non-locality, contextuality and entanglement. We shall describe recent work in which tools from Computer Science are used to shed new light on these phenomena. This has led to a number of developments, including a novel approach to classifying multipartite entangled states, and a unifying principle for Bell inequalities based on logical consistency conditions. At the same time, there are also striking and unexpected connections with a number of topics in classical computer science, including relational databases, constraint satisfaction, and complexity theory. The lecture will present an introduction to contextual semantics, in a self-contained, tutorial fashion.

Angela Wallenburg, Altran UK

Proof and Test: Will They Blend?

Extensive and expensive testing is the primary method used to gain confidence in safety-critical software today. There are some notable exceptions where formal software verification has been successfully used and scaled to large industrial projects. SPARK is a programming language, a set of verification tools, and a design approach for such critical systems. A number of military and commercial high integrity projects, ranging from 10 000 to 5 million lines of code, have been developed in SPARK. Examples include Rolls Royce Trent (engine control), EuroFighter Typhoon (military aircraft), and NATS iFACTS (air traffic control). We have identified two reasons why formal program verification is still a hard sell: 1) the difficulty of reaching non-expert users, and 2) the lack of a convincing cost-benefit argument. In this talk I will describe our approach to solve those two problems in the design of the new SPARK 2014 language and its associated verifying compiler, developed jointly by Altran UK and AdaCore. I will give an overview of some lessons learned from the programming language and verification research community, from the development of industrial standards such as DO-178C, and from our experiences in the industrial use of SPARK. In particular I will describe our unique integration of testing and proving. We argue that subprogram level formal verification using SPARK 2014 can be cheaper than testing in DO-178C terms, and that our integrated approach allows a mix of test and proof so that the most cost-effective method can be used for each part of a program.

Susanne Albers, Humboldt-Universität zu Berlin, the LMS-sponsored keynote

speaker in Discrete Mathematics.
Energy-Efficient Algorithms

We study algorithmic techniques for energy savings in computer systems. We consider power-down mechanisms that transition an idle system into low power stand-by or sleep states. Moreover, we address dynamic speed scaling, a relatively recent approach to save energy in modern, variable-speed microprocessors. In the first part of the talk we survey important results in the area of energy-efficient algorithms. In the second part we investigate a setting where a variable-speed processor is equipped with an additional sleep state. This model integrates speed scaling and power-down mechanisms. We consider classical deadline-based scheduling and settle the complexity of the offline problem. As the main contribution we present an algorithmic framework that allows us to develop a number of significantly improved constant-factor approximation algorithms.

Assia Mahboubi, INRIA–École Polytechnique
Computer-checked Mathematics

For the last decades, computers have been playing an increasing role in the everyday activity of many researchers in mathematics: for typesetting articles, for testing conjectures, and sometimes even for validating parts of proofs by large computations. However most mathematicians are hardly familiar with "proof assistants", which are also pieces of software for "doing mathematics with a computer". These systems allow their users to trust with the highest degree of certainty the validity of the proofs they have carefully described to the machine. So far proof assistants have been successfully employed to verify the correctness of hardware and software components with respect to given specifications, scrutinizing proofs that are too long and pedestrian to be checked by hand. In September 2012, a proof of the Odd Order Theorem (Feit-Thompson, 1963), which is a milestone for the the classification of finite simple groups, was machine-checked by the Coq proof assistant. In this case, the computer has verified a proof which does not rely on heavy computations but on a sophisticated combination of mathematical theories resulting in one of the longest published proof of its time. In this talk we will give an overview of the panel of research areas and methodologies that should be combined in order to ensure the success of such a formalization. Black (or white) board will eventually never be surpassed to convey and give rise to the intuitions of the mind who discovers new mathematics, but having proofs checked by a machine rather than by a human reviewer may open some new perspectives we will discuss.

Contributed Talks at BCTCS 2013

Chris Bak, University of York
Rooted Graph Programs

We present an approach for programming with graph transformation rules in which graph programs can be as efficient as programs in imperative languages. The basic idea is to equip rules and host graphs with distinguished nodes, so-called roots. At the start of the search process for the match of a graph transformation rule, roots in rules are matched with roots in host graphs. This facilitates a local search of the host graph in the neighbourhood of its root nodes, enabling rules to be matched and applied in constant time, provided that host graphs have a bounded node degree (which in practice is often the case). Hence, for example, programs with a linear bound on the number of rule applications run in truly linear time. We demonstrate the feasibility of this approach with a case study in graph colouring using the graph programming language GP.

Mohamed Arikiez, University of Liverpool
Combinatorial Optimization Techniques in Domestic Renewable Power Management

Our work is in the emerging area of Computational Sustainability. We contend that the area has a great potential for fostering cutting-edge research in Computer Science and related disciplines. In particular, the main aim of the research presented here was to design an intelligent interactive control system that efficiently manages the household energy needs taking into account presence of renewable power (hybrid Solar/Wind) and the resident's preferences in order to reduce consumed power from the utility grid and increase the immediate renewable power (RP) utilization (the ratio of total consumed RP to total Generated RP) without decreasing the comfort level. Despite the fact that installing a domestic renewable power generation system can reduce power bills, the utilization of this power still needs improvement because sometimes the surplus of RP could hit 70% (depending on output power of generation system and consumed power in the building) but no intelligent mechanism exists to try and exploit such resource before it gets dumped to a storage system or the national grid. We describe a novel Knapsack formulation that can be used to solve the resulting allocation problem and analyse its performances both in a real-life and simulated environment. Our results suggest that the approach could allow the immediate use of as much as 90% of the generated power surplus.

Giles Reger, University of Manchester
A pattern-based technique for inferring first-order temporal specifications

Formal program specifications are useful for a number of different applications – the most obvious being formal program verification. But they can also aid program understanding, test generation, bug location, software development and other new applications that are the subject of active research. However, formal specifications are difficult and costly to write, and as a consequence, precise specifications are often missing, incomplete or informal. This has led to a growing interest in the area of specification inference (also known as model inference, specification mining, automata learning). These techniques extract temporal properties or state-based invariants from code or, more often, dynamic program traces. These techniques are defined by the coverage they can achieve, the expressiveness of the specification language they target and their ability to scale with program size. In this talk I introduce a technique for inferring temporal specifications that deal with data. In order to handle data effectively I make use of a highly expressive specification language (Quantified Event Automata) developed within the context of runtime verification to infer specifications using a technique where specification patterns are mined from program traces and then combined together. By targeting an expressive specification language this technique is able to discover useful specifications whilst maintaining scalability by adopting algorithms for efficient runtime monitoring.

Andrew Lawrence, Swansea University
Program Extraction in Action: A Verified Clause Learning SAT Solver

Modern SAT solvers typically include optimizations such as clause learning which are rarely treated with formal methods in practice. In this talk we show how to obtain such an optimized SAT solver together with a formal correctness proof by the method of program extraction from proofs: we have formalized a constructive proof of completeness for a modified DPLL proof system combined with unit resolution and extract a conflict driven clause learning SAT algorithm. This algorithm is capable of learning information during the search for a proof as well as performing non-chronological backtracking. This is a new case study in the area of program extraction and opens up many possibilities for future work. It also demonstrates how efficiency considerations can be taken into account at the proof level. The formalization and extraction has been carried out in the interactive proof assistant Minlog.

Gregory Woods, Swansea University
A Case Study On Imperative Program Extraction

The process of program extraction has long been associated with functional programs with little research in the direction of imperative program extraction. While many useful tools exist to extract functional programs (Agda, Isabelle, Coq and NuPRL) the simple fact is that most programs that are written are more towards the imperative paradigm. In this talk we explore a case study which demonstrates that imperative program extraction is possible. The problem we choose to solve using this method is the classic of sorting a list of numbers. Many algorithms exist to solve this problem and we will focus on one of the most famous, Quicksort. We present a successful attempt at extracting a program, that yields imperative behaviour, from a constructive proof. The software used for this is the interactive theorem prover Minlog.

Matthew Gwynne, Swansea University
Towards a theory of good SAT representations

We aim at providing a foundation of a theory of "good" SAT representations (CNF clause-sets) F of boolean functions f. The hierarchy UC_k of unit-refutation complete clause-sets of level k was introduced by the authors, based on notions of hardness and generalised unit-clause propagation (UCP). We argue UC_k provides the most basic target classes for representation. That is, for a good representation, F in UC_k is to be achieved for k as small as feasible.

The first level of the hierarchy, UC_1, is the same as the class UC of unit-refutation complete clause-sets, introduced in 1994. The aim of UC was to offer a class of clause-sets which was good for knowledge compilation and representation. More formally, UC is the class of clause-sets where unit-clause propagation (UCP), a simple linear-time inference algorithm, is sufficient decide questions of clausal entailment. In 1995 the class $SLUR$ (Single Lookahead Unit Resolution) was introduced as an umbrella class for efficient satisfiability (SAT) solving. The motivation was to offer an algorithm for efficiently deciding satisfiability for existing poly-time SAT classes, including renamable Horn, extended Horn, hidden extended Horn, simple extended Horn, and CC-balanced clause-sets. In previous work we generalise SLUR to a hierarchy $SLUR_k$, again using generalised UCP, and show that these two hierarchies are in fact equal ($SLUR_k = UC_k$). This brings together the two notions of representation and efficient SAT solving, and allows one to think of "finding a good representation" as a form of "SAT knowledge compilation". As a first application of this dual perspective, we show that, for (fixed) $k \geq 1$, deciding whether a clause-set is in UC_k is coNP-complete.

UC_k is directly related to the space complexity of tree resolution. However, in general, it is known that modern SAT solvers can (in some sense) simulate stronger proof systems such as full-resolution. Using the notion of resolution width, we introduce the hierarchy WC_k of clause-sets with width-hardness k; for all k the class UC_k is a subset of WC_k. We introduce lower bound methods for WC_k and use these to prove separation results between UC_{k+1} and UC_k, as well as between WC_{k+1}) and WC_k. More formally, we show that for every $k \geq 1$ there are sequences of boolean functions with polynomial size equivalent clause-set representations in UC_{k+1} which have no equivalent polynomial-size representations in WC_k. The boolean functions for these separations are "doped" minimally unsatisfiable clause-sets of deficiency 1; we generalise their construction and show a correspondence to a strengthened notion of irredundant sub-clause-sets. Turning from lower bounds to upper bounds, we believe that many common CNF representations fit into the UC_k scheme, and we give some basic tools to construct representations in UC_1 with new variables, based on the Tseitin translation.

Augustine Kwanashie, University of Glasgow
The Hospitals/Residents Problem with Free Pairs

The classical Hospitals/Residents problem models the assignment of junior doctors to hospitals based on their preferences over one another. In an instance of this problem, a stable matching M is sought which ensures that no blocking pair can exist in which a resident r and hospital h can improve relative to M by becoming assigned to each other. Such a situation is undesirable as it could naturally lead to r and h forming a private arrangement outside of the matching. This however assumes that a blocking pair that exists in theory would invariably lead to a matching being undermined in practice. However such a situation need not arise if the lack of social ties between agents prevents an awareness of certain blocking pairs in practice. Relaxing the stability definition to take such a scenario into account can yield larger stable matchings.

In this talk, we define the Hospitals/Residents problem with Free pairs (HRF) in which a subset of acceptable resident-hospital pairs are defined as free. This means that they can belong to a matching M but they can never block M. Free pairs correspond to resident and hospitals that do not know one another. Relative to a relaxed stability definition for HRF, called local stability, we show that locally stable matchings can have different sizes and the problem of finding a maximum locally stable matching is NP-hard, though approximable within 3/2. Furthermore we give polynomial time algorithms for three special cases of the problem.

Alexander Baumgartner, RISC, Johannes Kepler University of Linz
A Variant of Higher-Order Anti-Unification

The anti-unification problem of two terms t_1 and t_2 is concerned with finding their generalization, a term t such that both t_1 and t_2 are instances of t under some substitutions. Interesting generalizations are the least general ones. The purpose of anti-unification algorithms is to compute such least general generalizations. For higher-order terms, in general, there is no unique least general higher- order generalization. Therefore, special classes have been considered for which the uniqueness is guaranteed. One of such classes is formed by higher-order patterns. These are lambda-terms where the arguments of free variables are distinct bound variables. A rule-based anti-unification algorithm in simply-typed lambda-calculus which computes a least general higher-order pattern generalization will be presented. The algorithm computes it in cubic time within linear space and it has been implemented.

Iain McBride, University of Glasgow
The Hospitals / Residents Problem with Couples

Large scale allocation processes can be modelled as matching problems involving sets of participants who may express preferences over members of other sets. Centralised matching schemes, which use algorithms to solve the underlying matching problems, are often employed in such allocation processes.

The National Resident Matching Program (NRMP) was established in 1952, in response to problems with the previous competitive system, to match graduating medical residents to hospitals in the US, matching 25,526 students in 2012. A similar process is used in Scotland to match medical graduates to Foundation Programme places via the Scottish Foundation Allocation Scheme (SFAS). These schemes may be modelled by a classical combinatorial problem, the Hospitals / Residents Problem (HR).

Centralised matching schemes such as these have had to evolve to accommodate couples who may wish to be allocated to (geographically) compatible hospitals. This extension, which can be modelled by the Hospitals / Residents Problem with Couples (HRC), has been in operation in the NRMP for a number of years and has also been applied more recently in the SFAS context.

The classical Gale-Shapley algorithm solves the Hospitals / Residents problem by finding a so called stable matching. We prove that, even under some very severe restrictions, the problem of deciding whether a stable matching exists, given an instance of HRC, is NP-complete. These complexity results drive the search for alternative methods of dealing with such problems.

We describe an Integer Programming model of the Hospitals / Residents Problem with Couples which produces exact, optimal solutions in larger instances where previously only heuristics, which are not guaranteed to terminate, have been applied. We prove the validity of the model and demonstrate the empirical performance of an implementation over a number of randomly generated datasets in addition to anonymised real data from the SFAS context.

Nosheen Gul, University of Leicester
A Process Calculus for Ubiquitous Computing

In the ubiquitous computing setting computing devices are distributed and could be mobile, and interactions among devices are concurrent and often depend on the location of the devices. Process calculi are formal models of concurrent systems and mobile agents. In particular, Calculus of Communicating Systems (CCS, for short) of Milner is a well suited formalism for agents executing concurrently, and Mobile Ambients (MA) by Cardelli and Gordon is a formalism for agents' mobility. We propose a process calculus for specifying mobility, communication, and concurrency in the ubiquitous computing setting. The calculus is inspired by CCS and Mobile Ambients. We use the idea of ports as in CCS, that allow agents to communicate on, and ambient capabilities as in Mobile Ambients, allowing the agents to move around. We give an LTS-based operational semantics for our calculus, which is inspired by Merro and Hennessy operational semantics. Then we provide some examples to show the usefulness of our calculus.

Andrew Fish, University of Brighton
Ordered Gauss Paragraphs

The talk will discuss recent work on the EPSRC funded Automatic Diagram Generation project which aims to build a unified framework for the automatic generation of mixed-type diagrams arising as the integration of Euler diagrams, knot diagrams, and graphs. There has been limited prior consideration of mixed-type diagram generation, and the intent is bring theoretical benefits by developing methods which make use of any commonality of abstraction, together with practical oriented benefits in terms of providing the groundwork for generic tool support for such diagrams that may be used in areas such as diagrammatic logics, or ontology and network visualisations. The talk will focus on Euler diagrams, which are collections of closed curves used to visualise set systems, discussing a new encoding for Euler diagrams, using Ordered Gauss Paragraphs, making use of an existing code together with methods for solving the planarity problem for knots in order to solve the corresponding planarity problem for Euler diagrams. We indicate how the code encapsulates the topology of the diagram, demonstrate the generality of the approach, and provide a link between knots and Euler diagrams via a construction which yields a family of Brunnian links which project to Edwards' construction of Venn diagrams, observing that the code rewriting methods developed are more widely applicable

Kevin McDonald, University of Aberdeen
A Substructural Logic of Layered Graphs

Complex systems, be they natural or synthetic, are ubiquitous. In particular, complex networks of devices and services underpin most of society's operations. By their very nature, such systems are difficult to conceptualize and reason about effectively. The concept of layering is widespread in complex systems, but has not been considered conceptually. Noting that graphs are a key formalism in the description of complex systems, I will establish a notion of a layered graph and provide a logical characterization of this notion of layering using a non-associative, non-commutative substructural, separating logic.

Layering need not be defined in one direction only: it may be that two graphs are layered over each other. In modelling terms, this would mean that whilst it remains useful to separate the two layers, resources can flow both up and down. To this end, I establish a notion of 'bi-layering' that is consistent with the basic notion of layering and also the intuitive notion of a stack.

I will define a class of algebraic models that includes layered graphs for which soundness and completeness results can be obtained. This gives a mathematically substantial semantics to this very weak logic.

The notion of layering that I develop has many natural applications in complex systems modelling. One particularly appealing area of application lies in security, such as instances of security circumvention or a flaw in the security policy of an organisation based on lax protocols. There are many others in a variety of network settings, the IP Stack, for example. I will present some simple examples before discussing how my work could be applied to more complex security issues such as investigating how I may begin to compose security models such as Bell-LaPadula and Biba in the layered environment.

Abiar S. Al-Homaimeedi, King's College London
Achieve pi-calculus Style Mobility in CSP

In process calculi, passing channel names is considered as transferring of communication capabilities from one process to another, usually called mobility. Introducing mobility into CSP as in the pi-calculus is not straightforward for the following reasons: (i) the parallel composition in CSP is parametrised with an interface set which governs the synchronisation between participants. Events in this set should be simultaneously per- formed by all participants whereas events outside this set (even if they are shared) are not. Although CSP parallel composition improves com- munication exibility, it lets processes alphabets play a significant role in the communication. Therefore, the silent growth of alphabets as in pi-calculus is not enough. Processes alphabets should be grown explicitly because of its relation with the parallel operator. (ii) restricting communication to names as the pi-calculus, is insuficient in the CSP. The restriction will compromise the CSP typed multi-way communications To overcome this problem several solutions have been proposed. However, each of these models have some drawbacks, therefore, in this talk, we propose a new mobility model to accommodate mobility into CSP. Our mobility model generalises the notations and relaxes the restrictions which are made by one of the previously proposed models. Additionally, we introduce a novel dynamic algorithm to update the synchronisation set of the generalised parallel operator.

Shang Chen, Loughborough University
Computability of Hybrid Systems

In this presentation, we will introduce several models of hybrid systems and discuss the computability of reachability and convergence properties of them. Hybrid systems are a model incorporating both discrete and continuous dynamics in the same formalism, which can be used to describe a large number of real-world applications. They are often used in places where we have some form of discrete device acting in a continuous environment. Firstly, we will introduce several mathematical models studied in this area such as piecewise constant derivative systems (PCD), piecewise affine maps (PAM), timed automata (TA) and rectangular automata (RA). We will then explain the problems we are interested in for these models, which include reachability, control and stability problems. We will then survey some known results from the literature from the point of view of decidability. Finally we will discuss future research directions, some applications of hybrid systems and some new areas which seem worthy of study.

Casper Bach Poulsen, Swansea University
Partial Derivation in Modular Structural Operational Semantics

Abstract: The scientific study of programming languages requires a formal specification of their semantics. However, the incentives of applying formal specification frameworks during programming language design are often outweighed by more pragmatic concerns, such as developing and maintaining an executable interpreter for the language under design.

One way of bridging the gap between formal specification and pragmatic programming language design is by making formal specifications pragmatic for the language developer. Modular structural operational semantics (MSOS), a modular variant of structural operational semantics (SOS), is a formalism that supports incremental and scalable language design, e.g., by taking a component-based approach to semantic specification.

Interpreting the transitive closure of the transition function for a set of MSOS rules gives a prototype interpreter, where evaluation corresponds to proof derivation using the underlying MSOS rules. However, a naive implementation of such an interpreter has a worst-case interpretive overhead where each proof step requires a number of inferences that is linear in the depth of the input term. Furthermore, while small-step semantics have several declarative advantages, term reduction using small-step rules requires more inference steps than when using their big-step counterparts. For the programming language designer who is concerned with efficiency, the considerable interpretive overhead incurred by a naive interpretation may be unacceptable in practice.

Here, we explore how to reduce interpretive overhead of small-step MSOS rules through partial evaluation techniques which, in our modular structural proof system setting, we will call partial derivation. Combining ideas from partial evaluation in logic programming, bisimulation theory, and refocusing in reduction semantics we show how to derive rules whose proofs require fewer inferences (and hence, whose evaluation requires less computation). Applying partial derivation to a semantics is a fully mechanisable transformation that gives a provably semantically equivalent set of rules. Furthermore, the techniques are broadly applicable, being constrained by only a very mild set of conditions for correctness. The transformations result in rules with a big-step flavour, hinting at the inter-derivability of small-step and big-step style semantics.

As a proof of concept, we have prototyped semantic rules in Prolog, where we can observe a significant reduction in the running time of interpreters based on partially derived semantics in comparison with their naive counterparts. We conclude that partial derivation is a viable technique for reducing interpretive overhead in modular structural proof systems and practical interpreters derived from these, and that partial derivation is a viable tool for prototypical and pragmatic language design.

Timothy Revell, University of Strathclyde
Relational Semantics of Type Systems

Category Theory, in particular cartesian closed categories, provide a powerful semantics for the simply typed lambda calculus (STLC). Logical relations are another model of the STLC using the category of relations. In this talk, we shall describe the relationship between these two models using a fibrational framework. We show how two important results, the Fundamental Theorem of Logical Relations (otherwise known as the Parametricity Theorem) and the Identity Extension Lemma have natural and simple formulations within this fibrational framework. We will conclude by discussing fibred category theory and how it can describe concisely the ideas of this talk. In particular, parametricity simply means that we shift from working in a categorical universe of categories, functors and natural transformations, to working in a fibrational universe of fibrations, fibred functors and fibred natural transformations.

David Wilson, University of Bath
Advances in Cylindrical Algebraic Decomposition

Cylindrical Algebraic Decomposition (CAD) was initially introduced to tackle the classic problem of quantifier elimination over real closed algebraic fields, however it has since seen many applications in its own right. Given a set of polynomials, multiple algorithms exist to produce a CAD such that over each cell the polynomials have constant sign. Inherently doubly exponential in the number of variables present, much work has been done to make CAD a practical tool through preconditioning, more efficient construction and truncated algorithms.

I will give a brief history of CAD before covering work conducted by the University of Bath real geometry research group. Recently, we have shifted emphasis to try and produce a CAD for a given *problem* rather than the set of polynomials involved. A major step forward is research on Truth Table Invariant CADs (TTI-CADs) for which a set of given clauses have invariant truth value over each cell. This research has also led to further investigation of how problems are formulated for input into various related algorithms.

Alongside new research, key applications will be discussed. In particular, recent work on the use of CAD to verify identities involving multi-valued functions over the complex numbers will be described. This work will be included in the forthcoming release of the computer algebra system MAPLE 17.

This work was conducted with James Davenport, Russell Bradford and Matthew England at the University of Bath. The work on TTICADs was also conducted jointly with Scott McCallum of Macquarie University.

Joseph Davidson, Heriot-Watt University
Elegance requires eloquence

oChaitin's exploration of his notion of program elegance using the Lisp language does not explicitly take into account the balance between a notation's expressive power and the richness of its semantics. To investigate further this link, we have developed a flavour of the Random Access Stored Program (RASP) machine to compare with the traditional Turing machine model.

By implementing interpreters and compilers from RASP to TM and vice versa, in both RASP and TM, we believe that we can gain a more precise view into the expressive power of these languages. Bootstrapping the compilers on one another will allow examination of the models from a common representation. We can also investigate the full abstract chain of the model, from the most abstract - the operational semantics - to the most concrete - the implementation of programs which actually run on realisations of these models.

This talk presents where we have come from, where we want to end up and what we hope to find along the way.

Paolo Torrini, Swansea University
Parametric polymorphism, value restriction and resource logic

Hindley-Milner polymorphism is a form of parametric polymorphism that is widely used in functional languages, for efficiency reasons. It is also known as *let* polymorphism, as it allows for generalisation of type variables that do not occur free in the environment of *let* expressions. The soundness of this form of generalisation relies on the logic of propositional quantification as enshrined in system F, although it can be syntactically defined on top of a distinction between types and type schemes, making it possible to dispense with explicit use of quantifiers.

In languages with references, there are well-known problems that arise when the term fed to the *let* expression is not a value. If the evaluation of this term requires allocation of new references, and its type depends on type variables that occur in the types of such references, one may end up with typeable expressions that still lead to runtime errors. This problem is usually dealt with by means of some form of the so-called *value restriction*.

In the classic approach, which dates back to the early '90 and is essentially due to Mads Tofte, value restriction is handled by distinguishing variables that may occur in the type of references (imperative), from those that cannot (applicative). The analysis in Tofte's paper shows that the justification of value restriction boils down, again, to the fact that variables can be generalised only when they do not occur free in the environment — though this time in an extended sense, that should take the store into account.

Tofte's analysis may then suggest, that by relying on a more expressive logic, allowing for premises to represent resources needed for evaluation, a more declarative formulation of the restriction could be given, simply based on the free variable criterion. In fact, given a typeable program, when the types of the references that need to be allocated for its evaluation are included in the premises of its typing judgement, the restriction on generalisation turns out be logically enforced without further ado.

In this talk, we first present the standard approach to value restriction in terms of imperative and applicative variables. We then outline an alternative approach based on intuitionistic linear logic, allowing for more expressive typing judgements which include store types. At a basic level, the new typing may be intuitively understood as obtained by reversing the operational semantic evaluation big step $\rho \vdash \langle t, \sigma \rangle \longrightarrow \langle v, \sigma' \rangle$ where the value v has type τ, and the new store σ' is obtained by extending σ with the newly allocated references $a_1 \mapsto v_1, \ldots, a_k \mapsto v_k$ of types $\tau_1, ..., \tau_k$, into a judgement of form $\Gamma; \Delta \vdash t \Rightarrow \tau$ where Γ types the environment ρ, and crucially, $\Delta = \{l_1 : \tau_1, \ldots, l_k : \tau_k\}$ types the difference between the old store and the new one, in terms of the locations $l_1, \ldots, l_k$ needed to allocate the new references.

Thomas Gorry, University of Liverpool
Faster Communication-less Agent Location Discovery on the Ring

This talk will be about our on going study of a randomised distributed communication-less coordination mechanism for n uniform anonymous agents located on a circle with unit circumference. We assume the agents are located at arbitrary but distinct positions, unknown to other agents. The agents perform actions in synchronised rounds. At the start of each round an agent chooses the direction of its movement (clockwise or anticlockwise), and moves at unit speed during this round. Agents are not allowed to overpass, i.e., when an agent collides with another it instantly starts moving with the same speed in the opposite direction. Agents cannot leave marks on the ring, have zero vision and cannot exchange messages. However, on the conclusion of each round each agent has access to (some, not necessarily all) information regarding its trajectory during this round. This information can be processed and stored by the agent for further analysis. The location discovery task to be performed by each agent is to determine the initial position of every other agent and eventually to stop at its initial position, or proceed to another task, in a fully synchronised manner. Our primary motivation is to study distributed systems where agents collect the minimum amount of information that is necessary to accomplish this location discovery task. Previously we have shown that by using a fully distributed randomised technique this location discovery problem can be solved in $O(n \log^2 n)$ rounds. However, we can now show improvements to this with an algorithm that solves the location discovery problem w.h.p in $O(n + \log^2 n)$ rounds.

Diana Cionca, University of Surrey
Path Dependency Analysis in Complex Systems

Nowadays, business environments are changing very fast from centralised and closed to distributed and open. Typically, they involve a large number of entities or agents who interact in a dynamic, uncertain and unpredictable fashion. There is growing interest in the development of analytical tools for understanding the behaviour of such complex systems both from an individual's point of view and from the global interaction perspective. Agent-based scenario analysis has been proposed for the analysis of complex systems. The agent's behaviour is considered the key factor that influences the overall system's evolution. An agent can reason to achieve certain goals, can act autonomously, has a knowledge-base about its environment and can interact with other agents. The objective here is to predict and model the evolution of a complex system through a set of rules which describe the behaviour and interactions of participating agents. We look into web services applications for open and distributed systems like the Web, and find that similar issues arise, especially with regard to orchestration (individual viewpoint) and choreography (global viewpoint) of participating services. We argue that the way these interactions are modelled, in particular with respect to handling concurrency, is important when it comes to specification and verification (conformance and realisability) of a choreography specification. In addition, we are keen to investigate the use of business rules in arriving at a choreography specification in a declarative fashion. We take a case study from the ERIE project on global food supply chains as a complex system and build a model which can be used to reason about the system's behaviour, in terms of inter-dependencies and different possible outcomes.

Ben Horsfall, University of Sussex
Using a separation logic for verification of reflective programs

Reflective programming allows one to construct programs that can manipulate or examine their behaviour or structure at runtime. One of the benefits is the ability to create more generic code that is able to adapt to being incorporated in different larger programs without modification to suit each concrete situation. Due to the runtime nature of reflection, static verification is limited and has largely been ignored. This talk gives an overview of research into a method for specification and verification of a reflective library by utilising a separation logic that has been extended with support for stored procedures. The approach stores the metadata on the heap such that a reflective library can be implemented and verified in terms of primitive commands, rather than developing new proof rules for the reflective operations. The specified library may then be used to verify programs that use reflection. The support for stored procedures in the logic is important for the chosen technique, where the metadata representation of method and constructor objects are realised as stored procedures. The supported reflective operations characterise a subset of Java's reflection library, and the approach is supported by a tool providing semi-automated verification.

Ferdinand Vesely, Swansea University
Compiler back-end for a component-based semantic specification framework
Traditional approaches to formal programming language specification are generally criticised for being difficult to use. This difficulty impedes their wider adoption. The main points of criticism are usually the notation, which requires too much effort to penetrate, or lacking tool support. Action semantics is one example of a framework that was designed to address the issue of comprehensibility in particular. It provides a closed collection of semantic entities. Concrete programming language constructs are defined by translations into action notation. The notation itself has some shortcomings, such as a somewhat unusual syntax using action combinators. A new framework is currently being developed that will provide an open ended collection of named fundamental constructs, or funcons. Each funcon has formally defined dynamic and static semantics and is stored in a repository. Real programming language constructs are defined in terms of funcons and thus programs can be translated into funcon terms. Case studies on a subset of OCaml and Caml Light have already been carried out and there is tool support for translating program terms into funcon terms as well as an interpreter for these translations. Modular SOS is used do give definitions of individual funcons. This variant of SOS was designed to address modularity issues of standard SOS and allows independent definition of language constructs. This is made possible by using transition labels for auxiliary entities and automatically propagating all unmentioned entities between the premises and conclusion of a rule.

As good tool support for prototyping of the language being designed is deemed critical for the success of a specification framework, multiple tools have been developed for action notation. Iversen developed a compiler for action notation which translates actions into Standard ML code. This code can then be compiled by an optimizing ML compiler. The compiler chain has been tested on Standard ML programs with satisfying results in execution speeds. The action compiler itself did not perform any optimisations. We build on Iversen's work on the action compiler and aim to develop an optimising compiler back-end for translating funcons into executable code. In the first phase, translations of funcons into Caml Light programs will be designed and implemented. A specification for this language is already available and we can use the existing tool support as a front-end to translate Caml Light programs into funcon terms. Once we have a working prototype of the back-end, we should be able to do a round-trip by translating from a Caml Light program into a funcon term and then back into Caml Light in a similar manner to Iversen's action compiler. This will allow us to evaluate our approach by comparing performance of code generated through funcons to code generated directly by the Caml Light compiler. In this talk we will discuss preliminaries and observe the differences between action notation and funcons. We will give an overview of Iversen's action compiler and suggest an approach to compiling funcons into Caml Light.

Andrew Collins, University of Liverpool
Visualisation and Analysis of Graphs

In this talk I will discuss work completed by the authors in the area of graph visualiation and graph analysis. Specifically I will show our current work in force directed algorithms for graph layout and the methods that we use to identify a significant vertex within a graph. Further I will make a reference to the future directions that we hope to take the work we have completed. While I will be showing mostly applied concepts, nearly all aspects of the work are backed by deeply theoretical work. Throughout this talk we will look at the visualisation and analysis of the retirement of Pope Benedict XVI and (possibly) the election of his successor.

Patrick Totzke, University of Edinburgh
Checking Equivalences and Preorders of One-Counter Processes

I will outline recent results on the Verification of Pushdown Systems, specifically on checking Bisimulation, Simulation and Trace inclusion of various restrictions of One-Counter Processes.

Of particular interest is a model called One-Counter Nets, that can be seen both as restriction of PDA and Petri nets and inherits a structural monotonicity from the latter. I will provide some intuition on how to expoit this property to provide decision procedures.

If time permits I will discuss the interplay of monotonicity and infinitely branching.

Robert Powell, Durham University
Skew Bisubmodularity and Valued CSPs

An instance of the finite Valued Constraint Satisfaction Problem (VCSP) is given by a finite set of variables, a finite domain of values, and a set of finite valued functions, where each function depends on a subset of the variables. The goal is to find an assignment of values to the variables that minimises the total sum of the functions. We study (assuming that PTIME ≠ NP) how the complexity of this very general problem depends on the functions allowed in the instances. The case when the variables can take only two values was classified by Cohen et al., with submodular functions giving rise to the only tractable case. Any non-submodular function can be used to express, in a certain specific sense, the NP-hard Max Cut problem. We investigate the case when the variables can take three values. We identify a new infinite family of conditions, that includes bisubmodularity as a special case, which can collectively be called skew bisubmodularity. By a recent result of Thapper and Zivny, this condition implies that the corresponding VCSP can be solved by linear programming. We prove that submodularity with respect to a total order and skew bisubmodularity give rise to the only tractable cases, and, in all other cases, Max Cut can be expressed. We also show that our characterisation of tractable cases is tight, that is, none of the conditions can be omitted. Thus, our results provide a new dichotomy theorem in constraint satisfaction research, and lead to a whole series of intriguing open problems in submodularity research.

Jules Hedges, Queen Mary University of London
Selection functions and games

Selection functions are a family of higher-type functionals related to continuations, introduced by Martin Escardo and Paulo Oliva to extract computational content from proofs in classical analysis. An unexpected connection with game theory arose: many apparently unrelated proofs in constructive mathematics can be seen as computing subgame-perfect equilibria of a suitable kind of generalised sequential game. I show that a certain amount of classical game theory carries over to this more general setting: generalised sequential games can be turned into simultaneous games based on von Neumann's 'strategic-form' construction, and Nash's theorem for the existence of mixed-strategy equilibria of finite games still holds.