Report on BCTCS 2025

The 41st British Colloquium for Theoretical Computer Science

14–16 April 2025, University of Strathclyde

Alasdair Lambert, Fredrik Nordvall Forsberg, and Sean Watters

The British Colloquium for Theoretical Computer Science (BCTCS) is an annual forum for researchers in theoretical computer science. A central aspect of BCTCS is the training of PhD students, providing an environment for students to gain experience in presenting their work, to broaden their outlook on the subject, and to benefit from contact with established researchers. The scope of the colloquium includes all aspects of theoretical computer science, including automata theory, algorithms, complexity theory, education, semantics, formal methods, concurrency, types, languages and logics.

BCTCS 2025 was hosted by the University of Strathclyde and held from 14th to 16th April 2025. The event featured an interesting and wide-ranging programme. A total of 31 contributed talks — predominantly by PhD students — were presented at the meeting alongside six keynote talks. Wednesday afternoon was focused on theoretical computer science education. We are grateful to The Scottish Informatics and Computer Science Alliance for sponsoring the meeting, and to the Heilbronn Institute for Mathematical Research provided for support in the form of travel bursaries to a number of PhD students and early career researchers.

BCTCS 2026 will be hosted by the University of Birmingham from 30th March to 1st April 2026. Researchers and PhD students wishing to contribute talks concerning any aspect of Theoretical Computer Science are cordially invited to do so. Further details are available from the BCTCS website at www.bctcs.ac.uk.

Invited Talks

Jess Enright (University of Glasgow)

How temporal locality can help us solve problems on temporal graphs

A temporal graph is a graph whose edges are active at only some (specified, known) times. This ability to capture changes over time provides a useful model of real dynamic networks, but renders many problems studied on temporal graphs more computationally complex than their static counterparts.

To contend with this, there has been recent work devising parameters with respect to which temporal problems become tractable. One such parameter is vertex-interval-membership width. Broadly, this gives a bound on the number

of vertices we need to keep track of at any time. We will outline how one uses this sort of parameter algorithmically, define a generalisation of this parameter, and give a meta-algorithm for both parameters. This simplifies proving that a temporal problem is tractable for either parameter.

This is joint work with Laura Larios-Jones, Sam Hand, and Kitty Meeks.

Rob van Glabeek (University of Edinburgh)
Ensuring Liveness Properties of Distributed Systems with Justness

Often fairness assumptions need to be made in order to establish liveness properties of distributed systems, but in many situations they lead to false conclusions. Here I present ongoing work aiming at laying the foundations of a theory of concurrency that is equipped to ensure liveness properties of distributed systems without making fairness assumptions. Instead I advocate assuming justness, a strong progress property that is essentially weaker and more realistic than fairness. When assuming justness, contemporary process algebras and temporal logics fail to make distinctions between systems of which one has a crucial liveness property and the other does not. This necessitates an overhaul of these frameworks.

Nicolai Kraus (University of Nottingham)
Dependent type theory: from propositions and sets to spaces

Type theories (in the sense of Martin-Löf and Coquand) are programming languages with type systems that are expressive enough to formulate mathematical statements. Several proof assistants, such as Agda, Lean, and Coq/Rocq, make use of this version of the Curry-Howard correspondence.

There are various interpretations of type theories, assigning different meanings to types. Traditionally, types were interpreted as propositions or sets, but since the advent of homotopy type theory, they are also seen as spaces. As a consequence, the correspondence between mathematical statements and types is not unique. For example, a group is usually represented as a carrier set with a unit element and an operation satisfying associativity, identity, and inverse laws. However, in homotopy type theory, it can alternatively be defined as a so-called "pointed connected 1-type".

In this talk, I will explain the ideas and insights that eventually led to homotopy type theory, along with an approach to computer formalizations that it enabled.

Conor Mc Bride (University of Strathclyde)
Talking Out Of School

Since 2008, I have been teaching first year undergraduates in semester one: fresh out of school! I have always taught more mathematical topics, which is to say that whatever topics I have been tasked with teaching I have made more mathematical.

Arriving from school, many expect to be taught to the test and few have much left of their sense of adventure. In the interests of prompting discussion, I'll share some tales from the trenches of trouble, tactics, and technology. Pilot light ignition is a hard problem. I have not solved it, but I have had moments. Misery loves company, and perhaps together we can manage more optimism than we can muster in isolation.

Jakub Opršal (University of Birmingham)
Homotopy and complexity of graph colouring

I will talk about an emerging new application of homotopy theory in computational complexity of discrete problems. This approach is build on the question: *How does the topology of the solution space of a computational problem influence its complexity?*

The idea emerged from investigations of the complexity of variants of graph colouring, including approximate graph colouring. A colouring of a graph with k colours is an assignment of colours to vertices under which no edge is monochromatic. Graph 3-colouring is a prototypical example of an NP-complete problem listed by Karp in his seminal paper. There are many variations on this problem whose complexity remains widely open. For example, although it is generally believed that colouring a 3-colourable graphs with a fixed number of colours is NP-hard, only hardness of colouring of such a graph with 5-colours is known (and shown only in 2019).

The talk will focus on several related results about variations of graph colouring that share a common theme of using a topological method based on tools from topological combinatorics and on ideas of Lovász [J. Comb. Theory, Ser. A, 25(3):319-324, 1978]. These ideas formalise the notion of a solution space, and hence provide a rigorous framework for the study of the above fundamental question

Elizabeth Polgreen (University of Edinburgh)
Making the most of large language models for program synthesis (when you want correct answers)

Since the release of ChatGPT, large language models (LLMs) have been dominating the discourse around code generation. In contrast, the best-performing solvers in the world of formal program synthesis are still based around enumerative algorithms. So, are we behind the times? Are LLMs the answer to all our synthesis problems? In this talk, I will present two approaches to making the most of LLMs in formal domains. First, I will present a technique based on combining enumerative program synthesis with guidance from an LLM [1]. Focusing on the Syntax-Guided Synthesis (SyGuS) competition benchmarks, we found that, whilst LLMs can produce syntactically correct SyGuS expressions, they fail to produce

semantically correct solutions for more than half of the benchmarks. We propose a novel enumerative synthesis algorithm, which integrates calls to an LLM into a weighted probabilistic search, allowing 2-way exchange of information between the two components, and increasing the number of benchmarks solved to 80% (and outperforming the state-of-the-art solver). Second, I will discuss our work using LLMs to generate models of systems for verification [2]. Here, we enable LLMs to generate syntactically correct code, even in programming languages that barely feature in the training data, by combining an HCI technique called natural program elicitation and a Max-SMT solver. We apply our approach to generating models of systems in the UCLID5 verification language, improving syntactic correctness from $< 10\%$ to $> 80\%$ on a set of textbook problems. Whilst I don't believe LLMs are the answer to all our synthesis problems, I think our results demonstrate that perhaps techniques used in formal synthesis and programming languages could be the answer to many of the problems facing LLMs...!

[1] Guiding Enumerative Program Synthesis with Large Language Models. Yixuan Li, Julian Parsert, and Elizabeth Polgreen.

[2] Synthetic Programming Elicitation and Repair for Text-to-Code in Very Low-Resource Programming Languages. Federico Mora, Justin Wong, Haley Lepe, Sahil Bhatia, Karim Elmaaroufi, George Varghese, Joseph E. Gonzalez, Elizabeth Polgreen, and Sanjit A. Seshia.

Contributed Talks

Quantale Enriched Semantics for Graph Mathematical Morphology

Ignacio Bellas Acosta (University of Leeds)

L-fuzzy relations, functions from a set $X \times X$ to a lattice L, have been used in mathematical morphology to capture the behaviour of structuring elements of grey-scale and colour-based images. However, L-fuzzy relations fail to capture the graphical behaviour found in graph-based mathematical morphology. In this talk we address this issue, developing a relational framework within quantale enriched category theory. In this framework, bi-modules (also referred to as profunctors in the literature) generalise L-fuzzy relations, addressing the graph nature of the problem. However, this generalisation comes with limitations on the definition of the converse and complement operations. Assuming the underlying quantale is Girard, we show the existence of a local adjunction of operations that play the role of the converse. Furthermore, we show the linear negation induces two pairs of adjoint operations that serve as generalisations of the complement operation. This talk is based on joint work with John G. Stell.

Malin Altenmüller (University of Edinburgh)

A data type of intrinsically plane graphs

Plane graphs are those that can be drawn on the sphere without any edges crossing. They are subject of interest not only in graph theory but also as a combinatorial representation of string diagrams for certain monoidal theories that are sensitive to their topology (for example non-symmetric monoidal categories). I will present an inductive data type of plane graphs which uses a graph's spanning tree as a skeleton and stores the remaining edges alongside it. The order in which we store these additional edges is crucial as it enforces the planarity property of the entire graph. We encode the planarity information as part of a graph's type, thus all graphs of the type are intrinsically plane. Additionally, operations on plane graphs such as substitution preserve the planarity property by definition.

Jungho Ahn (Durham University)
A coarse Erdős-Pósa theorem

An induced packing of cycles in a graph G is a set of vertex-disjoint cycles such that G has no edge between distinct cycles of the set. The classic Erdős-Pósa theorem asserts that for every positive integer k, every graph contains k vertex-disjoint cycles or a set of $O(k \log k)$ vertices which intersects every cycle of G. We generalise this classic Erdős-Pósa theorem to induced packings of cycles. We show that there exists a function $f(k) = O(k \log k)$ such that for every positive integer k, every graph G contains an induced packing of k cycles or a set X of at most $f(k)$ vertices such that the closed neighbourhood of X intersects every cycle in G. Our proofs are constructive and yield a polynomial algorithm.

This is based on a joint work with Pascal Gollin, Tony Huynh, and O-joung Kwon.

Pete Austin (University of Liverpool)
Temporal Explorability Games

Temporal graphs extend ordinary graphs with discrete time that affects the availability of edges. We consider solving games played on temporal graphs where one player aims to explore the graph, i.e., visit all vertices. The complexity depends majorly on two factors: the presence of an adversary and how edge availability is specified. We demonstrate that on static graphs, where edges are always available, solving explorability games is just as hard as solving reachability games. In contrast, on temporal graphs, the complexity of explorability coincides with generalized reachability (NP-complete for one-player and PSPACE- complete for two player games). We further show that if temporal graphs are given symbolically, even one-player reachability and thus explorability and generalized reachability games are PSPACE-hard. For one player, all these are also solvable in PSPACE and for two players, they are in PSPACE, EXP and EXP, respectively.

Jakub Bachurski (University of Cambridge)

Breaking records: structural subtyping as a language design principle

Subtyping is a classical programming language feature, usually associated with nominal subtyping in object-oriented programming. Structural subtyping — where the subtyping relation is derived from the shared structure of types — is an alternative. Structural typing is present in many forms in modern languages, but it is often incomplete without subtyping. Regardless, subtyping often ends up omitted because of its inherent complexity.

I argue that with the introduction of algebraic subtyping — a new framework for ML-style type inference — the time has come to revisit structural subtyping. In my thesis I propose a novel language design, Fabric, and demonstrate the relevance of structural subtyping to modern design problems. I propose a new approach to typing array programs, show how to capture record extension within algebraic subtyping without row polymorphism, and compile Fabric into WebAssembly with a focus on efficient core data structures. I aim to show structural subtyping narrows the expressive gap between static and dynamic languages, it helps us statically type check dynamic programs, and its ideas can improve how we program generally.

Justus Becker (University of Birmingham)

Proof translations for structurally different sequent calculi of intuitionistic modal logic

The proof theory of intuitionistic and modal logics has been extensively studied in recent decades; the advent of labelled and nested systems allowed to come up with very expressive and versatile cut-free complete systems. A good way to compare different calculi for the same logic is by establishing effective translations between derivations of these. Effective translations are functions between derivation trees that are definable via an algorithm, allowing for a more constructive completeness proof and a way to extract new systems out of this translation.

It is known that nested and tree-labelled sequents have the same expressive power, meaning that a calculus employing one can also be expressed in terms of the other formalism. This, for example, enables one to distinguish two calculi written in different formalisms.

The propositional modal logic IK has been developed in the 1980s and 90s as the intuitionistic variant of the modal logic K. Meanwhile, a lot of different proof systems have been developed for this logic.

Here, I will compare the fully labelled calculus labIK, that internalises the birelational semantics for intuitionistic modal logic, with the Maehara-style nested calculus NIKm. The sequents of these calculi are structurally inequivalent, which will lead to a translation that is not preserving the structure of sequents. It requires one, for instance, to add some admissible rules explicitly or to edit derivation trees

before translating them. This is also due to labIK being fully invertible, unlike the simple nested sequent calculus.

This result sheds some light on how structurally different calculi behave, especially comparing a fully invertible calculus and a system with non-invertible rules. We can see, for example, how retrieving lost information in backtracking translates to having all information available in the full calculus. This result can also be applied to many other intuitionistic modal logics.

Géraldine Brieven (University of Liège)
From Visualization to Coding: Practicing Graphical Loop Invariants in CAFÉ 2.0

This talk focuses on a programming approach relying on an informal and graphical version of the Loop Invariant to write the code. This approach is taught in the context of a CS1 course where students are exposed to various C programming language concepts and algorithmic aspects. The key point is to imagine a problem-solving strategy (the Graphical Loop Invariant) prior to coding, which then becomes reasonably straightforward when based on this graphical representation.

This talk presents the rules for building a sound and accurate Graphical Loop Invariant as well as the programming methodology. As such, our programming approach might be seen as a first step towards considering formal methods in programming courses without making any assumption on students mathematical background, as it avoids mathematical notation entirely. We also discuss CAFÉ 2.0, a learning tool we developed to support teaching and practicing the Graphical Loop Invariant concept. Finally, we provide insights into how students adopt this approach and utilize it in CAFÉ 2.0 to solve problems.

Paola Bruscoli (University of Bath)
Linking Proof Theory to Game Design

In the past few years I have started proposing individual projects to UG/PGT students in Computer Science in areas of structural proof theory, substructural logics and at the intersection with game development. Most students have a good acquaintance with game engines for their personal interests, however, they may be less exposed to formal logics than desirable during their studies. By appealing to what they know well, my aim is to provide them with pathway leading to acquire new knowledge on some proof systems, with the idea that there could be some benefit for the (simple) game design. For example: 1) knowing the dynamics of some specific game, can we understand it in terms of proofs and derivations in some logic? 2) is there any benefit for the programmer, in implementing the rules of the game in specific logical languages, as opposed to the (low level) language that game engines provide?

Harry Bryant (Swansea University)
Proof Checking for SMT-solving and its application in the Railway Domain

Railways are safety critical therefore the programs that control them must be safe and secure. Alongside the testing process there are tools that apply formal verification techniques that prove these systems meet the requirements and standards. Many of these railway verification tools utilise both SAT solvers and Satisfiable Modulo Theory (SMT) solvers to determine whether a given Safety Property holds. These solvers are very complex, and there is the danger that they prove the correctness of a verification condition when it does not hold. To increase trust and robustness, the objective of this PhD project is to produce a bespoke independent verified checker for our Industrial Partner for those verification results. This will determine whether the certificates produced by Z3 are correct and therefore implies that the theorem in question is valid — producing a validated log rather than a yes/no answer. Since the standards in the railway industry are high, we want as well to have a verified checker of proofs. Therefore, we want to show that if the checking procedure confirms that the proof is valid, then the theorem in question holds. Our Proof Checker is being developed by using the Proof Assistant Coq to prove the correctness of the procedures. Currently a proof-of-concept SAT checker for Ladder Logic Verification has been developed. The basis of both the SAT and SMT proof scripts in Z3 rely on Resolution and therefore work initially focused on writing a checking procedure for Unit Resolution in Coq. This has been proven in Coq to be correct and a mini checker extracted to OCaml. Unit Resolution was chosen because it is the foundation of the old Z3 proof script format. The new format utilises Reverse Unit Propagation (RUP) instead — built on the principle of Resolution. This allows us to use the proof of Unit Resolution as part of the proof that the RUP rule is correct. The proof-of-concept has been written and successfully tested on Industrial examples, with the proof of correctness ongoing. By using formal methods in the railway domain, it gives us assurances that the interlockings are correct before they go through the testing process – saving both time and resources. However, the engineers want a guarantee that the theorem prover Z3 is correct. Therefore, by developing a checking tool it gives further trust and robustness in the development process. The proof-of-concept will be extended to obtain a fully verified proof checker for Z3 proofs to fully meet the needs of the railway verification tools.

Dhurim Cakiqi (University of Birmingham)
Algorithmic syntactic causal identification

Causal identification in causal Bayes nets (CBNs) is an important tool in causal inference allowing the derivation of interventional distributions from observational distributions where this is possible in principle. However, most existing

formulations of causal identification using techniques such as d-separation and do-calculus are expressed within the mathematical language of classical probability theory on CBNs. However, there are many causal settings where probability theory and hence current causal identification techniques are inapplicable such as relational databases, dataflow programs such as hardware description languages, distributed systems and most modern machine learning algorithms. We show that this restriction can be lifted by replacing the use of classical probability theory with the alternative axiomatic foundation of symmetric monoidal categories. In this alternative axiomatization, we show how an unambiguous and clean distinction can be drawn between the general syntax of causal models and any specific semantic implementation of that causal model. This allows a purely syntactic algorithmic description of general causal identification by a translation of recent formulations of the general ID algorithm through fixing. Our description is given entirely in terms of the non-parametric ADMG structure specifying a causal model and the algebraic signature of the corresponding monoidal category, to which a sequence of manipulations is then applied so as to arrive at a modified monoidal category in which the desired, purely syntactic interventional causal model, is obtained. We use this idea to derive purely syntactic analogues of classical back-door and front-door causal adjustment, and illustrate an application to a more complex causal model.

Alec Critten (Swansea University)
Developing user propagators for graph-based SMT reasoning

Satisfiability modulo theories (SMT) solvers are automated reasoning tools known for their expressivity, efficiency, and robust scalability. They are widely used in industrial contexts for verification of computer programs and as part of quality assurance processes. In my PhD research, I am developing a SMT formalisation of graph theory to be applied to a setting in the UK railway industry. The construction of such a custom theory is possible with the recent introduction of the "user-propagator" interface in the Z3 solver. The application of this work is to support the Nodes-Edges Model of graphs that Siemens Mobility has developed to model diagrams of railway systems (known as "scheme plans"), and enable reasoning about properties over these graphs such as reachability and shortest-path in order to satisfy safety constraints ("design rules"). In this talk I will cover ongoing implementation work to bring this theory to fruition through user-propagators.

Aven Dauz (University of Strathclyde)
Modelling cybersecurity games with compositional game theory

Compositional game theory (CGT) and its corresponding Haskell DSL "open-games-engine", have been used for modelling microeconomic games, auctions, and smart contracts. The primary advantages of this compositional approach are

leveraging modularity and code-reuse to construct larger games. Incidentally, game theory has been used to model complex attack-defense scenarios in cybersecurity, with the simplest case modelling the strategic interaction between a single attacker and defender. Scaling these models to accurately reflect real-world attacks and create effective defense strategies remains an active area of research. In this talk I will present an application of CGT for building attack-defense games from smaller components, with a live-code demonstration showing how to use the DSL and generate equilibrium analytics. I'll construct games in the Bayesian and stochastic setting, and show two complete models: a honeypot allocation game and a DDoS attack-defense game between blockchain mining pools.

Abhishek De (University of Birmingham)
Bounded Henkin Quantifiers and the Exponential Time Hierarchy

In logic, quantifiers typically have an implicit linear order of dependencies. Henkin introduced a more general framework for quantifier prefixes, allowing for partial orderings among quantifiers. These quantifiers, now known as Henkin quantifiers, have been extensively studied in logic and have found significant applications in linguistics, arithmetic, and descriptive complexity. Surprisingly, bounded versions of Henkin quantifiers, which restrict the range of these quantifiers, have not been explored. This paper defines bounded Henkin quantifiers and examines their properties from a complexity-theoretic perspective. In particular, we show that the set of predicates definable by quantifier-free formulas prefixed by a bounded Henkin quantifier is exactly NEXP. The proof goes via machine models defined using bounded Henkin quantifiers. Finally, we show that formulas with Henkin quantifiers define a complexity class contained in Δ_2 in the exponential hierarchy and define a natural complete problem for this class.

Lewis Dyer (University of Glasgow)
The similar connected partition problem

We introduce a new graph partitioning problem where given a graph G, equipped with a function f assigning a number to each vertex, we aim to partition vertices of G into a specified number of parts such that each part is connected, satisfies upper and lower part size bounds, and such that the function values of pairs of vertices within a part differ by at most some specific part width δ. We introduce some early algorithms and hardness results for small part sizes, and show this problem is solvable in XP-time parameterised by treewidth and in FPT-time when parameterised by treewidth and maximum part size. We then briefly outline some possible future variants of the problem and discuss some applications in statistical inference for spatial data.

Aidan Evans (University of Cambridge)

An Algebraic Characterization of NC1

Krebs et al. (2007) gave a characterization of the complexity class TC0 as the class of languages recognized by a certain class of typed monoids. The notion of typed monoid was introduced to extend methods of algebraic automata theory to infinite monoids and hence characterize classes beyond the regular languages. We advance this line of work beyond TC0 by giving a characterization of NC1. This is obtained by first showing that NC1 can be defined as the languages expressible in an extension of first-order logic using only unary quantifiers over regular languages. The expressibility result is a consequence of a general result showing that finite monoid multiplication quantifiers of higher dimension can be replaced with unary quantifiers in the context of interpretations over strings, which also answers a question of Lautemann et al. (2001).

Murdoch Gabbay (Heriot-Watt University)

A shallow arithmetisation of first-order validity

Arithmetisation means translating validity in a model M of some assertion φ, into some property RM of a polynomial $P(\varphi)$, such that (to a high degree of certainty) φ is valid in M if and only if $P(\varphi)$ has property RM. Arithmetisation is useful in cryptography, because many cryptographic protocols work by exploiting properties of finite fields to prove things like "R holds of a polynomial p".

We note a compositional shallow translation from a fragment of first-order logic with equality, into univariate polynomials. This fragment is expressive, and in particular it is easily powerful enough to express computationally significant things like inductive definitions and combinator reduction. Thus, by arithmetising inductive definitions our translation also arithmetises computation, amongst other things.

Techniques already exist to arithmetise computation by coding it in an abstract machine whose computation has been arithmetised by hand; i.e. by a deep embedding in some preconstructed monolithic cryptographic artefact. What distinguishes our translation is its shallowness and compositionality: it translates logical connectives directly to operations on polynomials, without a deep encoding in an abstract machine.

The translation is, as a piece of pure logical manipulation, deceptively simple and straightforward, but we shall see that the outcome is surprisingly powerful and expressive. Future work is to operationalise it to obtain a useful and practically applicable zero-knowledge or succinct proof-scheme, and we conclude by briefly discussing the barriers to doing this.

Murdoch Gabbay (Heriot-Watt University)

A declarative approach to specifying distributed algorithms using three-valued modal logic

Coalition Logic is a three-valued modal fixed-point logic designed for declaratively specifying and reasoning about distributed algorithms, such as the Paxos consensus algorithm.

Coalition Logic adopts a declarative approach, specifying the logic of computation as an axiomatic theory, without prescribing control flow. Notably, message-passing and (algorithmic) time are not explicitly modeled, distinguishing this from approaches like TLA+. This abstraction emphasises the logical essence of distributed algorithms, offering a novel perspective on their specification and reasoning.

I will demonstrate the applicability of this approach to the Paxos consensus algorithm by presenting it as a logical theory and deriving its standard correctness properties.

I see Coalition Logic as a versatile tool for specifying and reasoning about distributed algorithms, offering a new lens through which distributed algorithms can be specified, studied, and checked.

Nathan Flaherty (University of Liverpool)
Fast and Safe Scheduling of Robots

Mobile robots are becoming an increasingly common part of scientific work within laboratory environments and therefore effectively coordinating them to complete necessary tasks around the laboratory whilst preventing collisions is of great practical interest. We model this theoretically by having a graph $G = (V, E)$ along with some tasks and robots, each task is on a vertex and has a certain duration, the time taken to complete the task, we then have robots which are agents that can move between adjacent vertices on the graph. The problem being to find schedules for the robots which complete all the tasks in G without any two robots sharing the same vertex or edge at any given time. In this talk I reintroduce an algorithm for Robot scheduling on a path with extensions for use on other graph classes, and present an Integer Linear programming formulation of this problem for use on all graphs. This leads onto experimental analysis of the algorithm on paths and lattice graphs. This is work done jointly with Duncan Adamson, Igor Potapov and Paul G. Spirakis.

Marek Jezinski (Swansea University)
Creating Synthetic Test Data for Rail Design Tools

The rail industry uses various software tools, e.g., to design scheme plans. However, how can such tools be tested? The manual design of rail artefacts such as scheme plans is laborious, costly, and an error-prone process. Thus, it seems appropriate to generate synthetic artefacts with a view to reducing the cost and workload for producing test data, and possibly with precise knowledge of where there are faults in these artefacts and where there are not.

In this talk, we present a method to generate artificial scheme plans based on Genetic Algorithms [1]. These algorithms simulate how a population evolves over time. Genetic Algorithms operate on data that is represented in the form of 'chromosomes' (standing for the single members of a population). The chromosomes themselves consist of 'genes' (providing the specific characteristics of a single member of the population). Genetic Algorithms start with an initial, randomly generated population. Evolution is then mimicked by iteratively producing new generations. To this end, offspring are generated using two operations: crossover (splitting and recombining selected genes of different members of the current population) and mutation (randomly modifying already recombined genes). Then, the next generation is selected by applying a fitness function to both, the members of the current generation and the newly generated offspring. This guarantees the survival of the fittest. The Genetic Algorithm terminates when the fitness of the current population exceeds a defined threshold [2, 3].

It is an important feature of the presented method that it includes the possibility of controlled fault injection, where we know exactly where there is a design flaw. In the context of scheme plans, a concrete example of a fault would be the following: placing a balise too close to a point's toe, as this contradicts the design rule that requires a minimum distance between a point's toe and a balise. By utilising genes that represent faults, it is possible to have populations which include faulty members. By choosing a fitness function that requires a certain number of flaws for survival, one can guarantee that all members of the final population will exhibit a pre-determined minimal/maximal number of flaws. On the technical level, our method has been realised in the programming language Python. Using Genetic Algorithms, our Python program generates scheme plans by attaching tiles with several passing loop variants, such as straight parallel tracks, points junctions, and diamond junctions. The user can choose several parameters, e.g., the initial population size or the fitness level for termination. From the final generation, one scheme plan is exported into a bespoke data format from Siemens. This allows for the testing of Siemens' tools. Scheme plan generation and scheme plan export is quick, within the order of seconds only.

We will demonstrate a running tool that produces artificial railway scheme plans using Genetic Algorithms. The scheme plans include faults documented within a log file providing the fault locations. This allows us to thoroughly test if the existing design checkers for scheme plans work correctly. Such tools have been developed by various companies and research groups, including Siemens' data checker [4], the ovado tool described in [5, 6] and the junction tool suite developed by Luteberget documented in [7].

Overall, this work provides a preliminary example demonstrating that the generation of synthetic test data using Genetic Algorithms is a viable approach in testing software tools used in railway design. It is future work to evaluate the quality

of the automatically generated test data, e.g., by developing notions of coverage (whether the fault frequency is 'realistic', all 'normal' situations are generated, or if corner cases are considered).

References

[1] A. E. Eiben and J. E. Smith, Introduction to Evolutionary Computing, 2nd ed., Springer, 2015.
[2] M. Z. R. Khan and A. K. Bajpai, "Genetic Algorithm And Its Application In Mechanical Engineering," International Journal of Engineering Research & Technology, vol. 2, no. 5, 2013.
[3] T. Harrison, Exploring the Feasibility of Using Genetic Algorithms for Generating Railway Map Test Data, Dissertation, Swansea, 2024.
[4] M. Banerjee, V. Cai, S. Lakhsmanappa, A. Lawrence, M. Roggenbach, M. Seisenberger and T. Werner, "A Tool-Chain for the Verification of Geographic Scheme Data," RSSRail 2023, LNCS, vol. 14198, pp. 211–224, 2023.
[5] R. Abo and L. Voisin, "Formal implementation of data validation for railway safety-related systems with OVADO," SEFM'13, LNCS, vol. 8368, pp. 221–236, 2014.
[6] T. Lecomte, L. Burdy and M. Leuschel, "Formally Checking Large Data Sets in the Railways," in Workshop on the experience of and advances in developing dependable systems in Event-B, 2012.
[7] B. Luteberget, Automated Reasoning for Planning Railway Infrastructure, PhD Thesis, University of Oslo, Faculty of Mathematics and Natural Sciences, 2019.

Thomas Karam (University of Oxford)

Adapting some basic matrix rank properties to the ranks of tensors

Tensors are higher-dimensional generalisations of matrices, and likewise the main notion of complexity on matrices — their rank — may be extended to tensors. Unlike in the matrix case however, there is no single canonical notion of rank for tensors, and the most suitable notion often depends on the application that one has in mind. The most frequently used notion so far has been the tensor rank (hence its name), but several other notions and their applications have blossomed in recent years, such as the slice rank, partition rank, analytic rank, subrank, and geometric rank. Unlike their counterparts for the rank of matrices, many of the basic properties of the ranks of tensors are still not well understood. After reviewing the definitions of several of these rank notions, I will present a number of results of a type that arises in many cases when one attempts to generalise a basic property of the rank of matrices to these ranks of tensors: the naive extension of the original property fails, but it admits a rectification which is simultaneously not too complicated to state and in a spirit that is very close to that of the original property

from the matrix case.

Thomas Karam (University of Oxford)
Communication of theoretical computer science notions, and mathematics

This teaching talk will focus on how to transmit notions in theoretical computer science in ways that prepare the recipient well for the input of mathematicians and furthermore do not create a tradeoff with the primary goals of theoretical computer science exposition. For concrete illustration we will discuss the case of Shannon entropy, a notion shared by the TCS and mathematics communities, which has an interesting history inbetween them: it was originally devised for communications engineering purposes, but then led to insights illuminating some distant topics in pure mathematics (and some other topics in physics), while at the same time continuing to have the kinds of applications that it had originally been conceived for.

Alasdair Lambert (University of Strathclyde)
Type Directed Programming for Programming Education

Live programming is a widely used technique in programming education. While it can be an effective tool for introductory classes, expert blindness is a common pitfall. It is also easy for students to assume that the problem solving ability is inherent to the lecturer and that they could never manage this independently.

To ameliorate this we propose Type Directed Programming as a teaching tool. This is where the type signature of a function guides its implementation details. Using type declarations as part of the live programming paradigm allows the lecturer to show that there is no magic, merely someone used to reading types.

Laura Larios-Jones (University of Glasgow)
Structural Parameters for Dense Temporal Graphs

A temporal graph consists of a static graph and a function that dictates when its edges are active. This allows us to model networks where the time at which connections occur is important. Unfortunately, many temporal problems are more computationally complex than their static counterparts. We look for temporal parameters such that, when these are small for the input of a temporal problem, we can efficiently solve the problem at hand. Many temporal parameters currently in use rely on some notion of sparsity in either the graph or function. We introduce three parameters that are small for dense and highly structured temporal graphs. These parameters are temporal cliquewidth, temporal modular-width, and temporal neighbourhood diversity. We also give examples of temporal problems which are tractable with respect to these parameters.

Louis Lemonnier (University of Edinburgh)

A recipe for the semantics of reversible programming

In this presentation, we explore the foundational elements required to interpret reversible programming within a categorical framework. We use the symmetric pattern-matching language introduced by Sabry, Valiron, and Vizzotto as a reference point, and we incorporate several improvements. We show that inductive data types and recursion can also be effectively modelled in this setting. However, these results do not straightforwardly extend to the pure quantum case. We provide insights into the challenges encountered and propose potential directions for addressing these limitations, for example with guarded recursion.

Jessica Newman (University of Southampton)
Alternating-Time Temporal Logic with Dependent Strategies

Alternating-Time Temporal Logic (ATL) is a logical system for multiagent strategic reasoning, in which formulae express temporal properties that coalitions of agents can enforce by concurrently fixing a strategy played across an infinite sequence of normal-form games. We extend this with the ability to specify an order in which agents sequentially fix their strategies, where agents that move later are able to select a strategy dependent on those of the previous agents; this gives a richer way of expressing coalitional power, since agents can enforce different outcomes depending on which of their opponents they are able to respond to. We characterise the sets of outcomes that are possible for a coalition to enforce over different move orderings in a normal-form game, and use this to give a sound and complete axiomatisation of this extended ATL with Dependent Strategies (ATLDS) as well as time complexity bounds for its key decision problems.

Olga Petrovska (Swansea University)
The Art of Teaching Theory Across Diverse Backgrounds

As higher education becomes more accessible, with the emergence of pathways like degree apprenticeships, we are seeing classrooms filled with an increasingly diverse student population. While this diversity is welcomed development, it also presents certain challenges, for instance when introducing highly theoretical concepts to learners with varied backgrounds.

In this talk, I will explore strategies to make abstract ideas accessible for all students. Using the example of a module on Algorithms and Automata, I will discuss storytelling as a pedagogical technique that can bring these abstract concepts to life. Additionally, I will present the ways in which students and educators can benefit from generative AI, while also addressing the potential pitfalls of misuse. Finally, I will present some ideas on assessment designs that encourage authentic student work, ensuring that the integration of AI enhances rather than undermines the educational process.

Benjamin Plummer (University of Southampton)
Coalgebraic Traces by Following Strategies in Two-player Games

In this talk, we introduce the process-theoretic notion of trace into two-player controller-versus-environment games. We demonstrate that the trace semantics at a given game state correspond to the set of subsets of plays that the controller can force, capturing the controller's strategic influence. Building on the coalgebraic framework for traces developed by Hasuo et al., we identify a suitable monad to represent the two layers of branching in these games. This monad, constructed using a weak distributive law to combine two powerset monads, effectively computes what the controller can enforce in a single step. We also present a novel categorical description of strategies, and discuss how our approach easily extends to games with probabilistic environments. This work provides fresh insights into the interaction between strategies and traces, and lays a coalgebraic foundation for game-based program synthesis.

Riu Rodríguez Sakamoto (University of Strathclyde)
Decorated Para for linear quadratic regulators

One of the classical examples of optimal control are regulators of linear dynamical systems with quadratic costs. We employ a characterization of the Para construction as certain Spans with left leg projections due to Capucci and Jaz Myers, together with a dualization of the theory of decorated cospans over vector spaces, to define 'decorated Para' as a category of linear quadratic regulators. We then prove that the algebraic Riccati equation forms a lax functor into a category of cost functions and smooth maps between them.

Yury Savateev (University of Hertfordshire)
Visualising the Hyperbolic Plane

Common visualisation techniques for hyperbolic plane usually are limited to standard models like Poincare disk model. This significantly limits the part of the hyperbolic plane that can be displayed at a workable scale, which diminishes usefulness. We propose a new way to visualise the hyperbolic plane that allows more of the plane to be displayed in a useful manner.

Zephyr Verwimp (University of Oxford)
On hypergraph colouring variants and inapproximability

I will discuss (hyper-)graph colouring problems, and how their computational complexity is captured by a certain notion of symmetry. I will then present recent results on inapproximability of conflict-free and linearly-ordered hypergraph colouring.

Tansholpan Zhanabekova (University of Liverpool)

Semantic Flowers for Good-for-Games and Deterministic Automata

We present an innovative approach for capturing the complexity of ω-regular languages using the concept of flowers. This semantic tool combines two syntax-based definitions, namely the Mostowski hierarchy of word languages and syntactic flowers. The former is based on deterministic parity automata with a limited number of priorities, while the latter simplifies deterministic parity automata by reducing the number of priorities used, without altering their structure. Synthesising these two approaches yields a semantic concept of flowers, which offers a more effective way of dealing with the complexity of ω-regular languages. This letter provides a comprehensive definition of semantic flowers and shows that it captures the complexity of ω-regular languages. We also show that this natural concept yields simple proofs of the expressive power of good-for-games automata.