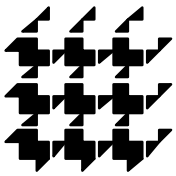
PLEASE POST

DIMACS



Center for Discrete Mathematics & Theoretical Computer Science
A National Science Foundation Science and Technology Center

DIMACS Summer School on Applied Logic and Algorithms

General Announcement

1 Introduction

The 1995–96 DIMACS Special Year topic is **Logic and Algorithms**.

To launch this Special Year, DIMACS will hold a Summer School on Applied Logic and Algorithms during August 1995. The Summer School is intended to expose industry, graduate students, postdocs, and experienced researchers from other fields to the three focus areas of the Special Year — **Finite-Model Theory**, **Proof Complexity**, and **Computer-Aided Verification**. The courses will provide students with a deep understanding of these research areas and will point out connections with applications.

• Finite-Model Theory is the study of the finite mathematical structures that satisfy particular axioms. It already has yielded elegant connections with theoretical computer science, including model-theoretic characterizations of complexity classes. Further, when a class of finite mathematical structures such as graphs is equipped with probability measures, one often can develop powerful meta-theorems called zero-one laws, which give conditions under which probabilities must approach zero or one as the structure size goes to infinity. Much recent interest in this area has concentrated on questions that have no infinite counterparts.

• Proof complexity studies the lengths of proofs and the strength and computational content of inferences in logical formalisms. A recent focus is on lower bounds for the length and syntactic complexity of propositional proofs, with connections to the "$\mathcal{P}$ vs. $\mathcal{NP}$" question, to circuit complexity, and to cryptographic security. Another focus involves bounded fragments of Peano Arithmetic, restricted so that they define exactly the predicates and functions in complexity classes such as the polynomial hierarchy. Exciting recent work has shown that if certain such theories of bounded arithmetic can prove lower bounds in complexity theory, then corresponding cryptographic systems cannot be secure.

• Computer-Aided Verification studies algorithms and structures for formally verifying properties of programs. It draws upon techniques from graph theory, combinatorics, automata theory, complexity theory, Boolean functions and algebras, logic, Ramsey theory and linear

programming. It also is linked to Finite-Model Theory through the study of finite-state programs, which often arise in real-world software and hardware protocols. Since the DIMACS CAV workshop in 1990, worldwide interest in CAV has grown enormously, not only in academia but in companies like Intel, DEC, Sun, IBM, and AT&T. This creates an unusual and rewarding opportunity to see theory put directly into practice.

The Summer School will consist of three successive one-week tutorial courses, one for each topic. Each week, 3–5 experts will lecture for a total of approximately 20 hours, and time and space will be provided for informal discussion as well.

We are seeking funding to support attendance by graduate and postdoctoral students, emphasizing participation by members of underrepresented groups.

The DIMACS Conference Center at Rutgers can accommodate about 80 participants. Subject to this capacity constraint, courses are open to all researchers; there is no registration fee. Although registration at the door is permitted, we ask that you register by June 30, 1995, to give us some idea of attendance. Pre-registrants will be given priority in case of capacity constraints. In addition, information on local housing will be sent to people who pre-register.

DIMACS is a Science and Technology Center funded by the NSF, with institutional participation by Rutgers University, Princeton University, AT&T Bell Laboratories, and Bellcore. Research and education activities at DIMACS focus on such areas as analysis of algorithms, combinatorics, complexity, computational algebra, discrete and computational geometry, discrete optimization and graph theory. A primary activity of the Center is to sponsor *Special Years*, year-long research programs on topics of current interest. The last three Special Year topics have been Computational Biology, Parallel Computing, and Combinatorial Optimization.

The following form may be used either to pre-register for courses or to apply for financial support. The deadline for applying for financial support is May 30, 1995. In either case, please email it (ASCII or PostScript) to `ida@dimacs.rutgers.edu`, or mail it to

Ida Castellano
Rutgers University
DIMACS Center
P.O. Box 1179
Piscataway, NJ 08854–1179

DIMACS Summer School 1995: Registration Form

Name: _____

Affiliation: _____

Current position: _____

Mailing Address: _____

_____

_____

_____

Email address: _____

Daytime telephone number: _____

Fax number: _____

I plan to attend the following tutorial course(s):

Finite-Model Theory	14–18 Aug	☐
Bounded Arithmetic and Proof Complexity	21–25 Aug	☐
Computer-Aided Verification	28 Aug – 1 Sept	☐

. . . . . .

The remaining questions are for people applying for financial support. If you are only pre-registering, not applying for support, **stop here.**

Citizenship: _____

Sex (optional): Female ☐ Male ☐

Major disability (optional): Yes ☐ No ☐

Primary racial or ethnic background (optional):

American Indian or Alaskan Native	☐	Asian	☐
Black, not of Hispanic origin	☐	Hispanic	☐
White, not of Hispanic origin	☐	Pacific Islander	☐

Names, affiliations, and phone numbers or email addresses of two people familiar with your research and education who have agreed to serve as references:

_____

_____

_____

_____

Please ask these people to send letters of recommendation to Ida Castellano at DIMACS (address above). If you're a graduate student or a new Ph.D., please send an unofficial copy of your graduate transcript to the same address.

Please attach a current CV, a bibliography, and a personal statement of at most one page, outlining what you hope to gain from, and contribute to, the Summer School. Discuss related research you've done in the past, your research plans for the near future, and how the Summer School will help you further them.

2 Course in Finite-Model Theory

Dates: August 14-18, 1995
Organizer: James Lynch <jlynch@sun.mcs.clarkson.edu>

The continuous interaction between logic and computer science during the past twenty five years has brought about a change in the focus of traditional mathematical logic. For almost a century mathematical logic centered around the study of first-order logic on the class of all structures (both finite and infinite) or over fixed infinite structures of mathematical significance, such as the integers, the real numbers, or more abstract spaces and algebraic structures. While exploring the connections between logic and computer science, researchers realized the importance of focusing on the logical properties of finite structures. As a result, in recent years first-order logic and various extensions of first-order logic have been investigated in depth with regard to their finite models. This area of research, which became known as finite-model theory, brings together mathematical logic, combinatorics, computational complexity theory, and database theory.

The finite-model theory course will consist of three parallel, but interconnected, series of lectures.

Descriptive Complexity: The main aim of descriptive complexity is to classify algorithmic problems according to the resources needed to express such problems in various logics on finite models. Research in descriptive complexity has amplified the close relationship between logic and computation, as evidenced by natural characterizations of virtually all major complexity classes in terms of logical expressibility on finite models. In turn, this provides an opportunity to analyze complexity classes by studying the expressive power of of logics on finite structures.

Expressive Power of Logics: The lectures on expressive power will present methods for investigating both the strengths and the limitations of logics on finite structures. In particular, the method of combinatorial games has been used extensively to obtain upper and lower bounds for expressibility, and to delineate the relative expressive power of logics. This area interacts directly with database theory, since a relational database is essentially a finite model, and many database query languages correspond to logics on finite structures.

Random Finite Models: The third area, random finite models, explores the connections between asymptotic combinatorics and logics on finite structures. Sets of finite models definable by formulas in various logics are considered, and probability measures are assigned to the sets of finite models of a given size. In many cases, the probability that a formula holds for a random model has a simple asymptotic behavior. Random finite-model theory seeks to determine which classes of random finite models and logics exhibit such behavior, as well as to identify in each case the computational complexity of the asymptotic behavior.

Lecturers

- Neil Immerman (Ph.D. Cornell University) is an Associate Professor of Computer Science at the University of Massachusetts, Amherst. He has also taught at Tufts University and Yale University. His research interests include descriptive complexity, complexity theory, database theory and logic.

- Phokion G. Kolaitis (Ph.D. University of California, Los Angeles) is a Professor of Computer and Information Sciences at the University of California, Santa Cruz. He has held visiting appointments at Stanford University and the IBM Almaden Research Center. His research interests include finite-model theory, computational complexity, and database theory. In recent years, he has served on the program committees of several international conferences and on the editorial boards of two journals. In 1993 he was awarded a John Simon Guggenheim Memorial Foundation Fellowship.

- James F. Lynch (Ph.D. University of Colorado) is an Associate Professor in the Department of Mathematics and Computer Science at Clarkson University. He has held visiting appointments at the University of Florida, the Courant Institute of Mathematical Sciences, New York University, and the University of Western Australia. Most of his work in finite-model theory has involved random finite models. He has also done research in combinatorics, mathematical biology, and complex systems.

3 Course in Bounded Arithmetic and Complexity of Proofs

Dates: August 21-25, 1995
Organizer: Toniann Pitassi <toni@cs.pitt.edu>

The study of the complexity of proofs is fundamental to logic, and has rich connections to complexity theory as well. First, the complexity of propositional proof systems is intimately connected to whether $\mathcal{NP}$ equals co-$\mathcal{NP}$. Secondly, provability of certain statements in feasible arithmetic (theories whose inferences have feasible computational content) gives rise to feasible algorithms. These two areas, propositional proof complexity and feasible arithmetic, are closely related to one another, and also to other fundamental problems in circuit complexity. In this workshop, we shall first cover the basics of propositional proof complexity and feasible (bounded) arithmetic, then introduce more advanced topics toward the end of the week.

Complexity of Propositional Proof Systems: In propositional logic, we are concerned with the basic question "How long must a proof of a propositional tautology be, as a function of the size of the tautology?" The best known upper bound for all known proof systems is exponential in the size of the tautology, and it is conjectured that a similar lower bound holds. The (non-)existence of a superpolynomial lower bound for all proof systems is equivalent to whether or not $\mathcal{NP}$ equals co-$\mathcal{NP}$, a question still unsolved after 25 years of study. An approach toward proving that $\mathcal{NP}$ is not equal to co-$\mathcal{NP}$ is to establish superpolynomial lower bounds for specific proof systems of greater and greater strength. The last ten years have shown substantial progress in proving such lower bounds, often using connections and techniques from complexity theory. We shall discuss Resolution, Frege, Bounded-Depth Frege, Extended Frege, and Cutting Planes systems. We shall also cover recent lower bounds for Resolution, bounded-depth Frege, and Cutting Planes systems.

Bounded arithmetic: The concept of a feasible proof — a proof whose inferences correspond to computationally feasible operations — has attracted much research in the last ten years. It has emerged as a robust and natural concept with several elegant formalizations, including "bounded arithmetic", a hierarchy of bounded subsystems of Peano arithmetic. The subsystem known as S_2^1, introduced by S. Buss in 1986, is widely believed to capture the intuitive notion of "feasibly provable": a fundamental theorem due to Buss states that proofs of certain types of statements in S_2^1 give rise to polynomial-time algorithms with feasible correctness proofs. This theorem was the first of several "witnessing theorems" for bounded arithmetic. We shall thoroughly discuss and prove Buss's witnessing theorem, then state and prove another witnessing theorem for bounded arithmetic, showing that the provable collapse of the bounded arithmetic hierarchy is equivalent to the collapse of the polynomial-time hierarchy.

Connecting Feasible Arithmetic and Propositional Logic: We shall then explain the connections between systems of bounded arithmetic and propositional proofs. In particular, S_2^1 can be viewed as a uniform version of polynomial-sized Extended Frege proofs, and $S_2(f)$ as a uniform version of polynomial-sized bounded-depth Frege proofs. Lastly, we discuss what can and cannot be proven within bounded arithmetic. It turns out that a large portion of mathematics can be carried out within S_2^1, including almost all known explicit circuit lower bounds. This not only shows the surprising power of S_2^1, but has also led to simpler proofs of known lower bounds. We also discuss consequences of the provability/unprovability of certain complexity-theoretic statements in bounded arithmetic.

Advanced Topics: Time permitting, we shall also cover several advanced topics:

1. The work of Razborov and Rudich on Natural Proofs, and Razborov's work on the unprovability of $\mathcal{P}$ versus $\mathcal{NP}$ in S_2^1;
2. A new, algebraic proof system based on Hilbert's Nullstellensatz, and new results in this area; and
3. Interpolation theorems.

Lecturers

- Samuel R. Buss (Ph.D. Princeton University) is Professor of Mathematics and Adjunct Professor of Computer Science at the University of California, San Diego. He has also held visiting positions at the Mathematical Sciences Research Institute and at the University of California, Berkeley. His research interests are in mathematical logic and computer science, specializing in bounded arithmetic, proof complexity, circuit and computational complexity, and pattern matching.

- Paul Beame (Ph.D. Toronto) is an Associate Professor in the Department of Computer Science and Engineering at the University of Washington. He was a postdoctoral fellow at M.I.T. and in 1988 he received an NSF Presidential Young Investigator Award. His research interests include computational complexity, parallel computation, and logic, with a particular emphasis on lower bounds.

- Alasdair Urquhart (Ph.D. Pittsburgh) is Professor of Philosophy and Adjunct Professor of Computer Science at the University of Toronto. His research interests include nonclassical logics, algebraic logic, lattice theory, the history of logic and complexity theory. He is the editor of Volume 4 of the Collected Papers of Bertrand Russell. He has served as a consulting editor of the Journal of Symbolic Logic, and is currently an editor of the Springer Lecture Notes in Logic series.

4 Course in Computer-Aided Verification

Dates: August 28 – September 1, 1995
Organizer: Kenneth McMillan <mcmillan@cadence.com>

The problem of verifying computer programs and computer systems dates back to the invention of the bug. Almost since that time, computer scientists have dreamt of computer-aided techniques that could verify the correctness of these complex systems. Recently, the need for such techniques has become increasingly urgent, as systems with the complexity of the last decade's "mainframes", now are fabricated on a single silicon chip.

This course will cover a variety of methods in computer-aided formal verification, ranging from general and powerful theorem-proving methods, that require significant user guidance, to more specialized, highly automated tools, especially model-checkers.

Overview

The most general approach to verification is to state the correctness condition for a system, as a theorem in a mathematical logic, and to generate a mechanically checked proof of this theorem using a general-purpose theorem-prover. This approach has attained significant successes in verifying microprocessor designs, for example. Theorem-provers generally provide decision procedures or heuristic algorithms to automate some steps of the proof, but require the user to provide at least the broad outline of the proof.

At the other extreme of the spectrum lies the finite-state technique of model-checking. A model-checker works on a finite-state model of the system to be verified, and a logical specification of a desired behavior of the system model. It checks that the model adheres to the specification by effectively searching the entire state space of the model. Model-checking has been used to verify automatically (or find errors in) systems of significant complexity, such as the cache coherence protocols of multiprocessors. Such results are achieved by using specification languages (such as temporal logics) that are expressive enough to state the required properties of concurrent systems, and yet can be checked with feasible computational complexity. In most cases heuristics are required, in order to circumvent the exponential explosion of states that occurs as a function of the number of components, in concurrent systems. These heuristics include using the compressed representations of sets of states afforded by binary decision diagrams, as well as using reduction techniques. The reduction techniques are heuristics which often can reduce the computational complexity of model-checking through a combination of abstraction (eliminating irrelevant detail) and decomposition (breaking large verification problems into small verification problems). Other reduction techniques exploit symmetries, and the partial ordering of events in concurrent systems.

Emphasis of the course

The course will provide a grounding in the basic models, theories and methods of computer-aided formal verification. This material will be introduced using realistic examples, which also will provide exposure to a variety of languages and tools currently used in formal verification.

Emphasis will be placed on comparing different methods, in terms of expressiveness, complexity, degree of automation, and the tradeoff between the expressiveness of a language and the complexity of its decision procedure. Some of the leading edge research problems will be introduced, such as the automatic generation of abstractions.

Course outline

1. Verification using general-purpose theorem-provers
2. Concurrency, temporal logic and automata theory
3. Abstraction and compositional methods
4. Methods based on binary decision diagrams
5. Partial-order based methods
6. Real-time and hybrid systems

Lecturers

- David L. Dill (Ph.D., Carnegie Mellon University, 1987) is an Associate Professor of Computer Science at Stanford University and a member of the Computer Systems Laboratory. Professor Dill's research area is the modeling, analysis, and verification of finite-state concurrent systems, particularly hardware designs.
- Robert Kurshan (Ph.D., Univ. of Washington, 1968) has been a member of the Mathematics Research Center at Bell Labs in Murray Hill, NJ since 1969. His areas of research include: ring theory, analytical number theory, coding theory, digital signal processing, approximation theory and computer-aided verification.
- Kenneth McMillan (Ph.D., Carnegie Mellon, 1992) was previously a member of the technical staff at AT&T Bell Laboratories in Murray Hill, NJ, and is now with Cadence Berkeley Labs in Berkeley, CA. His primary research area is the formal verification of hardware systems.
- J Strother Moore (Ph.D., University of Edinburgh, 1973) spent three years at Xerox Palo Alto Research Center, and five years at the Computer Science Laboratory of SRI

International before joining the faculty of the Department of Computer Sciences at the University of Texas at Austin in 1981. Along with Robert S. Boyer, he created the Boyer-Moore automatic theorem-prover and is a founder of Computational Logic, Inc., where he has worked since 1987, primarily on mechanized theorem-proving and its application to establish properties of hardware and software systems.

- Pierre Wolper (Ph.D., Stanford University, 1982) was a member of of the technical staff of Bell Laboratories in Murray Hill, NJ, from 1982 to 1986, where he pursued research on program specification and verification, temporal and modal logics, and distributed computing. He is now a full professor at the University of Liege. His current research interests include algorithms and data structures for the verification of reactive systems, temporal logics, and temporal databases.