# Automated reasoning: The impossible made indispensable

What caused Bob Constable, a dyed-in-the-wool theoretician, to get into the business of automated reasoning? "The need to formally verify programs," said Constable. "By 1974, David Gries and others in programming methodology were looking at program correctness as a way to begin understanding the programming process, and I thought that the computer could be a great help in this."

This was a radical departure from the accepted wisdom of the late 1960s. Automatic theorem proving had lots of skeptics. How could computers prove theorems? The problem of searching for a proof in a formal logic would be combinatorially infeasible. Moreover, formal theories such as *Principia Mathematica* were totally unreadable and unusable, and so would be proofs by computers —if they ever became possible.

Automated reasoning has become indispensable. Yet, according to Gödel, it is impossible! How can that be?

To make matters worse, Gödel's incompleteness theorem, one of the most widely discussed theorems of the 20th century, said essentially that no consistent formal theory of mathematics worth talking about is capable of even enumerating all true theorems, let alone deciding if statements are true.

In 1974, Constable started with two students to build a "program verifier", PLCV, which would check formal proofs in a logic of programs. Mike O'Donnell was studying programming languages, and Scott Johnson was a systems student. Together, they focused on interactive theorem proving in which the computer helps fill in tedious details and checks all steps, with the human providing the creative steps. To make the logic clean, the compiler had to support restrictions on PL/1 programs, so they collaborated with Dick Conway's PL/C compiler group. Ever since, the Cornell work in automated reasoning has involved theory, languages, and systems.

Thirty PhD students have worked in this area under Constable. Two of them, Doug Howe and Chet Murthy, used Cornell provers to solve open problems —for Howe the Girard Paradox (akin to Russell's paradox in set theory), and for Murthy an automatic constructivization of Higman's Lemma (certain constructions on well quasi-ordered sets preserve well quasi-orderedness). Cornell provers have verified important programs and systems used in industry and in science. They have helped a worldwide community create automated reasoning tools that are essential to Intel, AMD, Microsoft, and other companies.

Robert L. Constable, Dean of the Faculty of Computing and Information Science

Today, automated reasoning is alive and well. Formal proofs of over 50,000 theorems are accessible, many on the Web —that's about 500 books worth of formal mathematics. Among them are famous results, such as the fundamental theorems of arithmetic and algebra, and even Gödel's theorem. Georges Gonthier recently used the Coq prover to produce the definitive proof of the Four Color Theorem.

Automated reasoning has become indispensable. Yet, according to Gödel, it is impossible! How can that be?

Just as the intuitionist mathematician L. E. J. Brouwer predicted in 1907, mathematicians and computer scientists are more interested in proof than in truth. True theorems that can never be proved will remain outside the body of "certain knowledge" and will command less attention as the body of proven truths becomes more fully integrated into the intellectual fabric. Indeed, says Constable, "I think people will be more fascinated by computer-proved theorems whose proofs are too complex for humans to grasp without considerable additional work."

Computer scientists appreciate the constructive proofs espoused by Brouwer because such proofs implicitly include algorithms and data structures with proven properties. From them, systems like Coq and Nuprl can automatically synthesize programs known to meet specifications. When Constable first demonstrated this technique in 1984, it seemed like magic. But now in Europe, it is routinely taught and used. As Murthy's work showed, constructive logics can also make mysterious proofs far more comprehensible.

Remarkably, proof terms comprise a high-level programming language, and, as PhD student Jason Hickey showed, the language of proof terms is object-oriented. In this setting, interactive theorem proving becomes programming in a knowledge-intensive programming environment. "It is doubly thrilling," says Constable, "to find a constructive proof and then watch it execute." This connection between programs and proofs, which lay hidden for hundreds of years, is now being used to create industrial code of the highest reliability in applications requiring security, such as isolating domains of personal information on Smart Cards.

The atmosphere of collaboration in CS at Cornell has also led to joint work with Mark Bickford, Ken Birman, Christoph Kreitz, and Robbert van Renesse in integrating distributed computation into the Nuprl programming logic. Distributed systems are derived (in Java) from proofs of theorems about "event structures". As a side effect, Nuprl 5 is itself a distributed theorem prover —perhaps the only one of its kind. It allows people to collaborate remotely in proving theorems or, as demonstrated in one memorable seminar by student Lori Lorigo, to compete remotely to improve a proof.

Thirty years of work at Cornell on theory and experimentation have helped establish that computers can automate many intellectual processes. With access to substantial mathematical knowledge in digital form and by relying on heuristic knowledge captured from the best users, computers have become indispensable partners in knowledge formation. This is one of the profound contributions of computer science to intellectual history. The next 30 years will see a proliferation of specialized automated assistants regarded as indispensable partners in scientific problem solving —with CS at Cornell continuing to make significant contributions.

## The scholarly publishing revolution

The traditional vehicle for dissemination of scholarly results has been the paper journal, but we are in the midst of a revolution with far-reaching consequences. And Cornell is at the vanguard of this revolution.

There are several strands in this Cornell story. The first involves 15 years of research. In the 1990s, CS researchers Dean Krafft, Jim Davis, and Carl Lagoze were involved in a DARPA-funded consortium, working on the dissemination of computer science technical reports over the Internet. Cornell's work led to the Networked Computer Science Technical Reference Library (NCSTRL) and the underlying Dienst architecture for federating distributed document repositories and services.

Dienst provided the foundation for the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH), developed jointly by Cornell, Los Alamos National Laboratory, and international collaborators. OAI-PMH is now the global standard for exchanging structured data and metadata in the library, museum, and publishing communities.

Further evolution of Dienst led to the Fedora project, a collaboration between Cornell and the University of Virginia. Fedora integrates text, data, and services to reflect the nature of modern scholarly communication. Cornell's open source implementation of Fedora is used worldwide as the basis for scholarly publishing, commercial library systems, and content management.

In related work, Cornell is creating the core production systems and technical infrastructure for the National Science Digital Library (NSDL) project, which consists of over 193 NSF grants. Led by CS professor Bill Arms, Krafft, and Lagoze, the Cornell team is integrating the OAI-PMH and Fedora work with state-of-the-art Web crawling, classification, preservation, content management, and distributed authentication and authorization technologies to deploy a world-class digital resource.

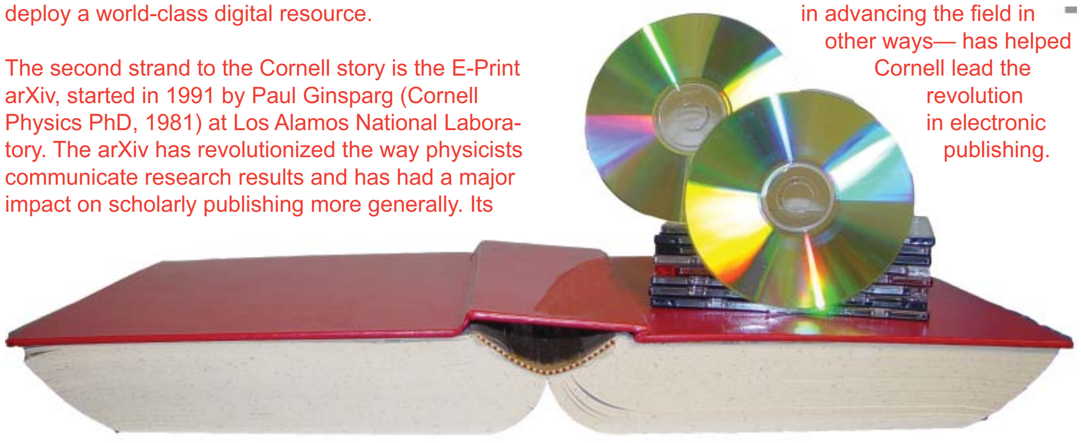
The second strand to the Cornell story is the E-Print arXiv, started in 1991 by Paul Ginsparg (Cornell Physics PhD, 1981) at Los Alamos National Laboratory. The arXiv has revolutionized the way physicists communicate research results and has had a major impact on scholarly publishing more generally. Its innovations have been emulated by other online literature and database systems, both academic and commercial. Unlike articles submitted to paper journals, articles submitted to arXiv.org are immediately available online, at no cost to the user. Ginsparg received a MacArthur Genius Award for this work.

In 2001, Ginsparg received a joint appointment in Physics and CIS, bringing with him arXiv and fellow researcher and arXiv developer Simeon Warner. The Cornell Library took over the day-to-day running of the arXiv, but Ginsparg and other colleagues in CIS continue to expand its functionality and use it as a testbed for research on large databases and user behavior in large archives. The arXiv now contains over 325,000 papers in physics, math, and computer science and is growing by about 50,000 submissions per year. It remains in the vanguard of ongoing transformations in scholarly communications infrastructure, serving as the prototype for many recently developed open-access systems.

The paper format has been eclipsed in importance by the electronic format in the majority of scientific and technical fields, and the trend is expected to become even more pronounced in the future.

The third strand of this story is the shortest. In the late 1990s, CS Professor Joe Halpern convinced the ACM to set up a preprint repository for computer science and led the effort. He and his committee decided that if the arXiv architecture were more open, it would be the ideal solution. Figuring out how to change the arXiv architecture was easy; Halpern just collaborated with Lagoze and Ginsparg —players in the other two strands. The result was CoRR, the Computing Research Repository, which is now the CS part of the arXiv.

The synergy between researchers in digital libraries and innovative faculty —who are interested not only in their research but in advancing the field in other ways— has helped Cornell lead the revolution in electronic publishing.



Omar Khan receives the CRA Outstanding Male Undergraduate Award.

Undergrad Eugene Lee takes first place in a national Intel Student Research Contest. Lee's project, supervised by Kavita Bala, tackled the challenge of producing high-quality, interactive rendering of sophisticated graphics, such as those used in movies or computer games.

Fred Schneider co-chairs the Microsoft Trustworthy Computing Academic Advisory Board.

Eva Tardos becomes editor-in-chief of the SIAM *Journal on Computing*.

Bill Arms becomes series editor of the MIT Press Series on *Digital Libraries and Electronic Publishing*.

Joe Halpern publishes *Reasoning About Uncertainty* (MIT Press).

Kavita Bala publishes *Advanced Global Illumination* (AK Peters) with Philippe Bekaert, and Phil Dutre.

Juris Hartmanis becomes Sr. Assoc. Dean of CIS.

2004

CS offers an undergrad degree in Information Science, Systems, and Technology in Engineering, joint with Operations Research & Industrial Engineering.

The PhD program in Information Science is approved.

The new lab $(CL)^3$, designed by David Schwartz, is inaugurated.

Dick Conway is honored by Management Science for his early, seminal research in computer simulation. The citation describes Conway's findings as "visionary" and says that they "established the research agenda for the simulation field for decades".

Researcher Carl Lagoze receives the LITA Frederick G. Kilgour Award. Lagoze's research, the citation says, "has lead to significant achievements in the areas of distributed digital collections, the harvesting of metadata, and establishment of open standards."

Lillian Lee shares the Best Paper Award at the Human Language Technology Conference, with Regina Barzilay. Their incorporation of context models in information ordering and extractive summarization yields substantial improvements.

Carla Gomes and Bart Selman receive the Distinguished Paper Award at the Conference on Principles and Practice of Constraint Programming.