

Welcome from the CS Chair	3
Symposium program.....	4
Meet the speakers.....	5
The Cornell environment	
The Cornell CS ambience.....	6
Faculty of Computing and Information Science.....	8
Information Science Program	9
College of Engineering	10
College of Arts & Sciences.....	11
Selected articles	
Mission-critical distributed systems	12
Language-based security	14
A grand challenge in computer networking.....	16
Data mining, today and tomorrow	18
Grid computing and Web services	20
The science of networks	22
Search engines that learn from experience	24
Sentiment analysis	26
Beauty is skin deep	28
Structural fingerprints of molecular evolution.....	30
The impact of computing on medicine	32
Reasoning about knowledge.....	34
Automated reasoning: The impossible made indispensable.....	36
Computational complexity.....	38
Sidelights	
The TRUST consortium.....	13
Programming methodology and program correctness	15
Computing in the arts.....	17
The marriage of structured and unstructured data	19
The Cornell Theory Center	21
Algorithms at Cornell	23
The father of information retrieval	25
CURIE comes to CS.....	27
Program of Computer Graphics.....	29
Bridging the Rift.....	31
The Game Design Initiative.....	33
Forty years of numerical analysis and scientific computing.....	35
The scholarly publishing revolution	37
Computational complexity and the ice cube.....	39
About the department	
Computer Science chairs	40
Computer Science faculty.....	40
Degrees granted	41
Selected faculty awards	42
Grants to young faculty	43
Books by the faculty	44
Selected invited addresses	46
The 40th anniversary committee.....	48



Welcome from the CS Chair

Ten years ago,
we realized that just as
computer science
had revolutionized
engineering in the
20th century,
it would revolutionize all
academic fields in the
21st century.

This brochure presents the Cornell Department of Computer Science and the symposium that celebrates its 40th anniversary. The information about the department is given in four ways:

- Description of the environment in which the Department of Computer Science operates, which has contributed in no small way to our success.
- Fourteen selected articles. Not all faculty are represented, and only a portion of our research is described. Accompanying these articles are sidelights that summarize other aspects of the department.
- Information about the department: a list of faculty, books by the faculty, and so on.
- A timeline of events, including when each tenure-track faculty member joined the department.



There are many good departments of computer science but only a few great ones. What distinguishes the great from the merely good? I believe the difference comes down to three things.

The first hallmark of a great department is excellence in research and teaching. No department can be a world leader in all areas of computer science, but a great department is a world leader in many of them.

The second hallmark of a great department is that it takes the lead in defining how the field evolves and has the courage to invest heavily in that future.

Finally, a great department must have character—it must possess some noteworthy characteristic, some idiosyncrasy, some texture that makes it unique among its peers.

Looking through a draft of this brochure, I was reminded of what it is that makes the CS department at Cornell one of the great CS departments in the world.

For many years, we have had world-class research programs in areas such as algorithms, complexity theory, distributed systems, languages, and numerical analysis. These areas continue to flourish at Cornell, as you will see in these pages.

Ten years ago, we realized that just as computer science had revolutionized engineering in the 20th century, it would revolutionize all academic fields in the 21st century. So, we worked with Cornell to create what is effectively a college of Computing and Information Sciences. In a very short time, this new structure has led to a university-wide explosion in interdisciplinary research organized around CS themes. Moreover, the timing of our decision to grow a first-class group in AI and machine learning couldn't have been better, because this area is crucial to such interdisciplinary work.

Finally, the unique texture of our department stems, I believe, from our collegiality. One of our enduring traditions is that the faculty gathers at noon every day to eat lunch while discussing the latest research results or barbecuing some unfortunate faculty candidate. A remarkable number of the collaborations between theoreticians and practitioners that you will find described in the pages of this brochure were sparked by lunchtime discussions at the Statler Club or a restaurant in Collegetown.

It is my privilege to chair this great department, and it is my pleasure to welcome you to the symposium to celebrate the 40th anniversary of its founding.

Charlie Van Loan

Symposium program

40th Anniversary Symposium

Department of
Computer Science,
Cornell University

1 October 2005

*Celebrating 40 years
of leadership in
research and
education*

- 9⁰⁰ *Welcome* Robert L. Constable
Dean, CIS
Introduced by Charles Van Loan
Chair, Computer Science
- 9¹⁵ *When complexity was king
and life thereafter* Al Borodin
Professor, Computer Science
University of Toronto
- 9⁴⁵ *From a bear to a lion* Zvi Galil
Dean, Engineering & Applied Science
Columbia University
- 10¹⁵ *Break*
- 10³⁰ *A cryptographer's perspective on privacy-preserving
data mining and statistical disclosure control* Cynthia Dwork
Senior Researcher
Microsoft Research
- 11⁰⁰ *Certifying algorithms* Kurt Mehlhorn
Director
Max Planck Institut für Informatik, Saarbruecken
Vice President, Max Planck Society
- 11³⁰ *Model checking: My 30-year quest to
make verification practical* Ed Clarke
FORE Systems Professor, Computer Science
Carnegie Mellon University
- 12⁰⁰ *Lunch*
- 1³⁰ *Beyond mice and menus* Barbara J. Grosz
Dean of Science, Radcliffe Inst. for Advanced Study
Harvard University
- 2⁰⁰ *Competition, cruelty, and compassion at Cornell
and the future of computer science* Bobby Schnabel
Vice Provost
Colorado University
- 2³⁰ *Gerry Salton's information retrieval
reaches the masses* Amit Singhal
Distinguished Engineer
Google
- 3⁰⁰ *Break*
- 3³⁰ *Security in distributed systems: Where we are, how we
got there, and how Cornell is trying to save us* Mike Reiter
Professor, Computer Science
Carnegie Mellon University
- 4⁰⁰ *Databases aren't dull
and other life lessons after Cornell* Jennifer Widom
Professor, Computer Science
Stanford University
- 4³⁰ *Evaluation + Design +
Implementation (Repeat) = Systems* Randy Katz
UMC Distinguished Professor, Computer Science
University of California Berkeley
- 7⁰⁰ *Banquet* David Gries
Associate Dean of Engineering
Introduced by Kent Fuchs
Dean of Engineering

Symposium speakers



Cynthia Dwork, PhD Cornell, 1981. Advisor: John Hopcroft
Cynthia's PhD thesis was on bounds on fundamental problems in parallel and distributed computation. She is a senior researcher at

Microsoft Research, Silicon Valley Campus, and a consulting professor at Stanford. Her principal areas of research are cryptography, distributed computing, and data privacy. She has made significant contributions in complexity theory, Web search, voting theory, interconnection networks, and algorithm design and analysis.



Randy Katz, PhD Berkeley, 1980; Cornell BA, 1976.
Randy joined EECS Berkeley in 1983, serving as chair from 1996–1999. While on leave in 1993–1994, he established whitehouse.gov and connected the White House to the Internet. He was one of the developers of RAID.

His current research interests are reliable, adaptive distributed systems supported by new services deployed on “network appliances”. He has numerous awards and is a member of the National Academy of Engineering.



Robert Schnabel, PhD Cornell, 1977. Advisor: John Dennis
Bobby's thesis was on quasi-Newton methods for unconstrained optimization. Bobby then headed west to Colorado, where he discovered perpetual sunshine and his wife. The combination have conspired to keep him

in Boulder ever since. His career has evolved from research in numerical optimization and parallel languages and systems to heading the Alliance for Technology, Learning, and Society, serving as campus CIO, and co-founding the National Center for Women and Information Technology.



Allan Borodin, PhD Cornell, 1969. Advisor: Juris Hartmanis
Allan's PhD thesis was on the existence of complexity gaps, so there must have been some simplicity in it, too. Allan planned to spend a year or so at his first academic position, CS at Toronto, but he forgot to move and has

been there for 36 years. In recent years, his interests have gravitated toward the design and analysis of algorithms (greedily utilizing the title of well-known textbooks).



Zvi Galil, PhD Cornell, 1973. Advisor: John Hopcroft
Zvi's PhD thesis concerned the complexity of resolution procedures for theorem proving. After a postdoc at IBM Watson, Zvi joined Tel-Aviv University. In 1982, he joined Columbia. He has been the Dean of Engineering &

Applied Science since 1995. His main interests are in algorithms, and he also contributes to complexity and cryptography. He is a member of the National Academy of Engineering.



Kurt Mehlhorn, PhD Cornell, 1974. Advisor: Bob Constable
Kurt's thesis was in abstract complexity theory. He then returned to Germany and switched his interest to algorithms. In 1975, he became full professor of computer science at the University des Saarlandes and in 1989 founding director

of the Max Planck Institute of Computer Science. His research interests are in algorithms, algorithms engineering, and software libraries. He is a co-founder of Algorithmic Solutions GmbH.



Amitabh Singhal, PhD Cornell, 1997. Advisors: the late Gerry Salton and Claire Cardie
Amit's thesis revisited Term Weighting. After Cornell, he worked at AT&T labs as a senior member of the technical staff and then joined

Google, where he is now a Distinguished Engineer. Amit's research interests include information retrieval and its application to Web search, Web graph analysis, and user interfaces for search.



Edmund Clarke, PhD Cornell, 1976. Advisor: Bob Constable
In his thesis, Ed proved that certain control structures did not have good Hoare-style proof systems. After his PhD, Ed spent time at Duke and Harvard and ended up at CMU. In 1981, Ed and his student Allen

Emerson first proposed the use of model checking as a technique for finite state verification, and that set Ed off on 25 years of pioneering research. Ed is a member of the National Academy of Engineering.



Barbara Grosz, PhD Berkeley, 1977; Cornell BA, 1969.
Barbara's thesis, on focus of attention in dialogue processing, established the field of computational modeling

of discourse. After 13 years out west, she realized she would never turn into a Californian and took a professorship at Harvard. She works on the design of collaborative multi-agent and human-computer interface systems. Her research and service to AI have been recognized by various honors, and she is widely respected for her contributions to the advancement of women in science.



Michael Reiter, PhD Cornell, 1993. Advisor: Ken Birman
Mike's PhD thesis was on security architectures for fault-tolerant systems. He joined Bell Labs in 1993, moved to AT&T Labs—Research during the breakup of AT&T, returned to Bell Labs in 1998, and joined CMU in 2001. He is

Technical Director of CMU's CyLAB, a university-wide center focused on developing new technologies for trustworthy computing. His research interests include computer security and distributed computing.



Jennifer Widom, PhD Cornell, 1987. Advisor: David Gries
David made Jennifer play her trumpet at her Admission-to-Candidacy Exam (her minor was music). Her thesis was on trace-based networks proof systems. After a few

years at IBM Almaden, she joined CS at Stanford in 1993. A leader in the database community, she was a Guggenheim Fellow and is a member of the National Academy of Engineering; but she is best known for scoring the first goal of the first game of the first Cornell CS women's intramural ice hockey team.

The Cornell CS ambiance

Computer Science at Cornell opened its doors in 1965, with just an MS-PhD program. The field's first task was to produce faculty to populate the future CS departments.

Long before computer science became a mature discipline whose technology is changing the face of the world, the Cornell CS Department believed that computer science was a deeply coherent intellectual discipline. We saw its traditional scientific character, the interplay between theory and experiment, and we imagined traits never seen before, such as the ability to design in ways nearly unconstrained by the physical world.

Juris Hartmanis, the founding chair and a Turing Award winner, who helped shape the theoretical character of the discipline, gave us a distinguished presence in computing theory. John Hopcroft joined in 1967 and soon became a leading theoretician, eventually winning the Turing Award for his fundamental contributions to the field of algorithms.

Founding member Gerry Salton, the father of information retrieval, helped establish the experimental side of computer science. Gerry's experiments with his SMART system gave rise to the vector space model and other technical concepts on which today's search engines are built and which helped create the new cyberspace frontier. More recently, Jon Kleinberg's fundamental work on ranking Web documents illustrated a similarly powerful synergy between theory and practice. His work, which is based on an elegant hubs-and-authorities model using fixed points in high-dimensional vector spaces, has greatly influenced how search engines rank pages.

The renowned Program in Computer Graphics (PCG) was founded in 1973 by Don Greenberg. Its theoretical work on light reflection models and surface modeling was directed toward synthesizing realistic images. The PCG succeeded, with several *Scientific American* covers, five SIGGRAPH achievement awards, and five alumni with Hollywood's technical Oscars, including CS professor Steve Marschner.

We came to stress the interplay between theory and practice in system design. For instance, many concepts and components in Ken Birman's Isis system, which is still in use in the NY Stock Exchange and the French air traffic control system, were inspired by theoretical ideas and algorithms developed by Fred Schneider, Birman, and others at Cornell. More recently, a long-term collaboration between Birman and Bob Constable led to the automatic verification of the equivalence between optimized and unoptimized protocol stacks. This work built on insights from Susan Owicki and David Gries's work on concurrent program verification.

Our AI group —now our largest group, with nine faculty— is known for its rigorous approach to central problems in AI. Joe Halpern (with Yoram Moses) won the 1997 Gödel Prize for using deep ideas from logic to solve problems in distributed systems. Bart Selman's research, which has led to new methods for solving large-scale reasoning and logistics problems, combines concepts from computation with statistical physics techniques for the study of phase transition phenomena, pioneered by Cornell's Ken Wilson, the 1982 Nobel Laureate in Physics and founder of the Cornell Theory Center.

This booklet illustrates in numerous ways both the traditional science model and its unique expression in computer science.

Our well-known culture of collegiality, nurtured from the start, has made our collective vision for computer science more than the sum of the individual faculty efforts.

Our collegial atmosphere has fostered a continual dialogue among the faculty on teaching, research, the nature of computer science, and the future of the field. In such a fast-moving and diverse field, such discussions have been essential. When faculty from decidedly different research areas talk together over an extended period of time, perspectives change; research broadens; respect increases; at times, joint research is done where it was not previously contemplated; and new subfields emerge at these boundaries.

Our culture of collegiality goes back to 1965, with the tradition of the CS faculty lunching together. Discussions over lunch and coffee covered everything from research, teaching, and student admissions to the culture of our new field. Of course, sports, cars, boats, literature, and politics were not ignored. Everyone, from instructor to full professor, voiced their opinions. This forum gave people a chance to meet a colloquium speaker, grill a recruit, or discuss the previous day's speaker.

The best theory is inspired by
practice. The best practice is
inspired by theory.

Donald E. Knuth



collegial: Characterized by
or having authority or
responsibility shared equally by
each of a group of colleagues.
Characterized by camaraderie
among colleagues.

Frequent faculty discussions, besides leading to mutual respect, have inspired a great deal of interdisciplinary work.

A collaborative, collegial atmosphere led our faculty to interact with faculty from other departments and to embrace interdisciplinary work. We helped create Cornell's Cognitive Studies Program and the Theory Center. We hired computational biologist Ron Elber, who works with Steve Tanksley in Plant Biology and David Shalloway in Biochemistry, among others, on protein structure problems. We helped create the new graduate field of Computational Biology. Keshav Pingali works with Tony Ingraffea in Civil &

Environmental Engineering to bring grid-computing ideas to bear on large multi-physics computational science simulations in the aerospace domain. This kind of work is at the core of the new subfield of Computational Science & Engineering, which Cornell is creating.

Another example of this kind of work is the new \$2 million NSF grant for research on petabyte storage devices for database-driven science, led by Alan Demers. It brings together seven CS faculty in databases, graphics, and Web analysis, along with Astronomy's Jim Cordes, who is collaborating on the design and implementation of a data management system for the Cornell Arecibo Telescope in Puerto Rico.

Because of the pervasive use of computing and the need for computer scientists to engage in joint research, we pressed Cornell to create the unique, college-level Faculty of Computing and Information Sciences

(CIS), whose sole purpose is to promote computing throughout Cornell. We are now part of CIS, although we are still affiliated with our two traditional homes: the colleges of Engineering and Arts & Sciences. Yes, CS at Cornell was born with interdisciplinary expectations, met them, and now has still broader interdisciplinary expectations.

Our department has always been a leader in education, both on campus and nationally.

We produced texts that influenced the development of the field. For example, John Hopcroft co-authored groundbreaking texts in algorithms and in formal languages and automata theory, David Gries wrote the first text on compiler construction, and Gerry Salton's texts led the field of information retrieval.

In the 1980s, Gries's text brought ideas about formal programming methodology into the undergraduate curriculum. Tom Coleman, Nick Trefethen, and Charlie Van Loan (co-author, Gene Golub of Stanford) wrote influential texts in scientific computing. And the influence continued into the 1990s and 2000s, with texts by Bill Arms, Dexter Kozen, Johannes Gehrke, Jon Kleinberg, and Eva Tardos, to mention a few. On page 44 is the list of books written by our faculty.

A teacher affects eternity; a teacher can never tell where their influence stops.

Henry B. Adams

Teaching has always been central to our mission. Senior and junior faculty continue to create and teach undergrad courses at all levels, and undergrad research is an important component of our curriculum. Research ideas move quickly into the curriculum, students benefit from faculty who are at the forefront of their areas, and our own research derives benefit from the interaction.

Small wonder that Cornell has given exclusive teaching and advising awards to seven of our current faculty members and that CS professor Dan Huttenlocher was recognized as the New York State Professor of the Year (over all disciplines).

Computer science has seen momentous changes since it started in the mid 1960s. One true invariant over these 40 years is the Cornell CS Department's leadership in research and education. We offer this symposium and publication as a celebration of that long and influential success. And we celebrate what distinguishes the department: a passionate dedication to computer science as a coherent discipline with deep synergy between theory and practice, a collegial atmosphere of mutual respect and support, a collaborative, interdisciplinary environment, and attention to education at all levels.

BCS

Dick Conway and Bill Maxwell of Industrial Eng. develop CORC on the Burroughs B-220 and Control Data 1604 to provide a simpler language than Fortran or Algol. CORC can be described on a single page. CORC is taught beginning in Fall 1962.

CS starts with faculty Dick Conway, Pat Fischer, Juris Hartmanis (Chair), Chris Pottle, Gerry Salton, Sid Saltzman, Bob Walker.

1965

The Computer Science Department is formed. (Conway spent his later years in the Cornell School of Management and is retired, and Salton passed away in 1995.) Housed in both Engineering and Arts & Sciences, CS starts with an MS/PhD program.

Gerry Salton brings his SMART system, started in 1961 at Harvard. SMART is his main tool for 35 years of experimental research in information retrieval.

Juris Hartmanis publishes the paper that starts the field of computational complexity, with Dick Stearns: *On the computational complexity of algorithms*, *Trans. Amer. Math. Soc.* 117 (1965), 285-306. Later, they receive the ACM Turing Award for this work.

CS produces its first PhD, Joel Sturman, a transfer from Electrical Engineering.

Ken Brown, Peter Wegner join.

1966

Juris Hartmanis and Dick Stearns publish the first of many influential texts by CS: *Algebraic Structure: Theory of Sequential Machines* (Prentice Hall).

Roland Sweet, John Hopcroft join.

1967

Dick Conway, Bill Maxwell, and Louis Miller, publish the classic text *Theory of Scheduling* (Addison-Wesley).

Howard Morgan, Alan Shaw, Robert Wagner, Bob Constable join.

1968

Gerry Salton publishes the classic IR text *Automatic Information Organization and Retrieval* (McGraw-Hill).

Faculty of Computing and Information Science

The creation of CIS in 2000 is just one of several innovative moves by Cornell in computing. In 1965, it placed the newly established CS department in both Engineering and Arts & Sciences so that it could blossom in many directions. The Program of Computer Graphics, created in 1973, has educated many of the leading graphics researchers in the world. The Cornell Theory Center, created in 1984, brought Windows-based high-performance computing to computational science and engineering. (The Center's cluster complex now has over 2000 processors.) Now, all three units are part of CIS.

The concepts, modes of thought, and technology of computing and information science have fundamentally extended our means of creating knowledge and are thus relevant in every academic discipline. They are also transforming the arts, because they provide new means of expression and virtual experience. But how should universities respond?

Cornell hit upon a unique strategy: create a college-level Faculty of Computing and Information Science (CIS), with computer science at its core. This college-level structure can create new programs, organize and recruit faculty, and sponsor research. But CIS has no students per se; instead, it offers undergraduate degrees in the colleges of Engineering, Arts & Sciences, and Agriculture & Life Sciences.

The mission of CIS is to integrate computing and information science—its ideas, technology, and modes of thought—into every academic field. This means working with seven colleges and four professional schools, which, combined, provide an unprecedented breadth of study.

CIS brings its faculty together with faculty throughout the university—from Anthropology, Astronomy, Aerospace Engineering, Biology, Electrical and Computer Engineering, History, Mathematics, Operations Research, Philosophy, Psychology, Sociology, and the social sciences and the humanities. Already, 25 academic departments cross-list courses with CIS.

Cornell grants PhDs in Information Science and in Computational Biology. Undergraduates in Cornell's three largest colleges can major in Information Science, and students in each of Cornell's colleges can pursue a minor in Information Science. Students can minor in Computational Biology in several colleges. Various programs in the Digital Arts are being developed, as well.

Interdisciplinary research is affecting academic disciplines and stimulating a new level of student interest in computing. Just as the needs of computational science and engineering have led to dramatic advances in high-performance computing, other areas such as biology, law, and the social sciences are calling for new software methods, tools, and products that serve a broad spectrum of commercial and individual users. Cornell and CIS have driven such innovation and will continue to do so in the future.

Ultimately, CIS will reach every Cornell undergraduate in more than 50 departments, as CIS and its subfields, like human/com-

puter interaction, attract a more diverse group of students. This broad reach is critical, as the information technology sector seeks to expand its impact and attract more young people to careers in industry.

At the core of CIS is the CS Department, and, as we partner more deeply with the social sciences and humanities, we are discovering new problems and new paradigms for using computers and information resources to automate intellectual processes and to create knowledge in ways that require these new “tools for thought”, leading to more research problems for computer science.

CIS houses the following academic units and institutes:

Department of Computer Science

Department of Statistical Science A home for faculty in statistics throughout Cornell who are working at the interface of math, computing, and data analysis.

Cornell Theory Center See sidelight on page 21.

Program of Computer Graphics See sidelight on page 29.

Information Science Program Brings together people who share an interest in combining computer science with the social sciences. Fifteen CS faculty are involved, along with a dozen others.

Computational Biology Program Five CS faculty are involved in this program, along with 20 others.

Computational Science and Engineering Program Brings together faculty from several dozen departments in offering interdisciplinary work. Four CS faculty are involved.

Information Assurance Institute Directed by CS professor Fred Schneider, the institute supports activities aimed at developing a science and technology base to enhance information assurance and networked information systems' trustworthiness—system and network security, reliability, and assurance.

Intelligent Information Systems Institute Directed by CS professor Carla Gomes, its main mission is to perform and stimulate research in computing-intensive and data-intensive methods for intelligent decision-making systems. Thirteen CS faculty members are involved.

National Science Digital Library Part of the NSF's long-term program to enhance all aspects of education in science, mathematics, and engineering. Cornell is a major contributor to the program, with six separate NSF grants. At Cornell, this program is directed by CS professor Bill Arms and CS researchers Dean Krafft and Carl Lagoze.

Information Science Program

Faculty of IS come from across campus:

- Earth and Atmospheric Sciences
- Communication
- Computer Science
- Design and Environmental Analysis
- Economics
- Applied Economics and Management
- Electrical and Computer Engineering
- Human Ecology
- Labor Economics
- Linguistics
- Physics
- Psychology
- Science and Technology Studies
- Sociology
- Operations Research and Industrial Engineering
- Law School
- Johnson School of Management
- School of Hotel Management
- Cornell Library

Information science is concerned with the design and use of information systems in a social context—with the creation, representation, organization, application, and analysis of information in digital form. IS examines the social, cultural, economic, historical, legal, and political contexts in which information systems are employed, to inform the design of such systems and to understand their impact on individuals, social groups, and institutions. IS is where computer science meets the social sciences.

The interdisciplinary Program of Information Science (IS) is a unit within CIS. The Charles and Barbara Weiss Director of IS is CS professor Claire Cardie, and half the CS faculty are members of IS. IS also has faculty from 15 other departments, spread over six college-level units and the Cornell Library.

The National Science Digital Library studies the problems of large-scale electronic publishing, Web information systems, scholarly communication, and the long-term preservation of digital information. For example, physics professor Paul Ginsparg founded the arXiv (see p. 37), and the Law School's Legal Information Institute is the leading public source for law. The CS part of this research investigates architecture, protocols, and services that facilitate the creation, management, accessibility, and longevity of distributed information. A Cornell team, headed by CS professor Bill

Arms and CS researchers Dean Krafft and Carl Lagoze, is building the central computing system for the NSF's National Science Digital Library program.

The HCI (Human-Computer Interaction) group, directed by professor Geri Gay, investigates social, psychological, and design issues involving computers at school, work, and home. This group worked with CS professor Thorsten Joachims on machine learning for Web search, reported on p. 24. The group is working on the evaluation of the NSF-sponsored Kinematic Models for Design Digital Library (K-MODDL), an open access, multimedia resource for learning and teaching about kinematics (the geometry of pure motion) and the history and theory of machines.

The core of K-MODDL is the wonderful Reuleaux collection of mechanisms, maintained by Frank Moon of Mechanical & Aerospace Engineering. Involved in K-MODDL is CIS professor Hod Lipson, who was the first to demonstrate physically working machines synthesized by self-organizing processes.

IS offers a PhD and three undergrad degrees: IS in the College of Arts & Sciences, IS in Agriculture & Life Sciences, and Information Science, Systems, and Technology in Engineering. Cornell University's motto, direct from Ezra Cornell, is, "I would found an institution where any person can find instruction in any study." CIS and its IS Program are taking this to an extreme, making it easy for students in all seven colleges to learn about computing and for faculty throughout the university to take part in IS.

Few universities offer such organized flexibility in interdisciplinary work related to computing. And CS itself benefits tremendously from the presence of IS.

107 Dart's Chamber Wheel Mechanism. Application: pump, steam engine, water motor. In the Reuleaux collection.

John Dennis, David Gries join.

1969

Gerry Salton becomes Editor-in-Chief of the *Journal of the ACM*—the first of many influential editorial positions held by members of CS.

John Hopcroft and Jeff Ullman publish their classic text *Formal Languages and Their Relation to Automata* (Addison-Wesley).

Ellis Horowitz, Jorge More, John H. Williams join.

1970

Dick Conway's group develops PL/C, a subset of PL/1 designed for instructional purposes. The PL/C compiler is distributed to 100 institutions and instantly becomes the standard instructional PL/1 compiler.

Jim Bunch joins. Gerry Salton becomes Chair.

1971

David Gries publishes the first text on compiler construction: *Compiler Construction for Digital Computers* (John Wiley & Sons).

Gerry Salton publishes *The SMART Retrieval System Experiments in Automatic Document Processing* (Prentice Hall).

Faculty members Jim Bunch and Jorge More win Householder Prizes for their PhD theses in numerical analysis.

Charles Moore, Tim Teitelbaum join.

1972

Bob Tarjan, Alan Demers join. CS grows to 15 faculty.

1973

Dick Conway and David Gries publish the first programming text to deal with issues of correctness, like loop invariants: *An Introduction to Programming, a Structured Approach using PL/1 and PL/C* (Winthrop).

Juris Hartmanis becomes the founding editor of Springer-Verlag's LNCS series (*Lecture Notes in Computer Science*) and David Gries becomes the founding Editor of Springer-Verlag's *Text and Monograph Series* (TMCS). Hartmanis and Gries maintain these positions for over 30 years.

John Hopcroft becomes Managing Editor of the *SIAM Journal on Computing*.



A department does not live in a vacuum; its environment helps shape its approach to education and research. CS is in the Faculty of Computing and Information Sciences, but it is also an integral part of the College of Engineering and offers two undergrad degrees there: Computer Science and Information Science, Systems, & Technology. CS is heavily involved in educational initiatives with Engineering and has research and education ties to faculty in Biomedical Engineering, Civil & Environmental Engineering, Mechanical & Aerospace Engineering (MAE), Computer & Electrical Engineering (ECE), and others.

The Engineering College is second to none in the range and quality of its experiential learning opportunities, with over 15 project teams, undergraduate research, and ESW courses. Many teams are interdisciplinary in nature and actively recruit CS majors. Below, we highlight some of these opportunities.

Robocup

Robocup designs and constructs autonomous robot soccer teams and competes in national and international competitions under the direction of Prof. Raffaello D'Andrea (MAE). The team, which first competed in 1999, has won the championship four times: 2003 in Italy, 2002 in Japan, 2000 in Australia, and 1999 in Sweden. The team takes pride in using a systems engineering approach. Contributions from students in CS, ECE, and MAE are equally important to the team's success.

Underwater Autonomous Vehicle

Under the direction of Prof. Kevin Kornegay (ECE), this team designs, builds, and tests autonomous underwater vehicles. The team placed first or second in each of the past three years in an underwater vehicle competition, held in the Space and Naval Warfare Systems Center in San Diego. AI is an important element of this project.

DARPA Grand Challenge

Cornell's entry was one of 40 from a field of 195 to advance to the semifinals in this DARPA-sponsored competition to have an autonomous vehicle race through the desert in Oct. 2005. Vision and AI are crucial to this project. Profs. Ephraim Garcia and

Hod Lipson of MAE and Dan Huttenlocher and Bart Selman of CS advise this team.

Formula SAE

Each year, a group of undergrad and grad students design, build, test, and race a car in the Formula SAE competition under the direction of Prof. Al George of MAE. In May 2005, about 140 schools competed in Detroit; Cornell won—for the ninth time in the past 18 years!

Engineers for a Sustainable World

ESW, a nonprofit organization founded in 2001 at Cornell under the inspiration of Regina Clewlow (CS '01), has expanded to chapters at 21 universities. Through domestic and international development work, education, and outreach, ESW "mobilizes engineers to address the challenges of global poverty and sustainability". An ESW course, offered by Civil & Environmental Engineering, teaches students design through projects that serve society in some way. Examples are vegetable oil as an alternative to diesel fuel, storm-water management in the Virgin Islands, and construction of a bridge here in Ithaca. CS professor Graeme Bailey is vice president of the board of directors.



CS student Dan Stowell did his MEng project in 2004-05 in connection with the ESW course. His team reduced the time and effort needed to survey, plan, and implement the design of viaducts, canals, and holding tanks for bringing water to villages. Their testbed was in Honduras. Using software that he and others wrote to use GPS in recording data, his team eliminated the need for traditional surveying equipment, reduced surveying time by 90%, and reduced construction costs considerably. Stowell learned valuable lessons in engineering design. Some trips to Honduras were an added bonus!

College of Arts and Science



When Cornell created the CS Department, it had the foresight to place CS in both Engineering and Arts & Sciences. CS has elements of both engineering and science, and to place it in one college could have hampered development. The later placement of CS in the Faculty of Computing and Information Science (CIS) provides more flexibility in furthering interactions with Cornell's seven colleges.

A&S students can major in two computing-related degrees —CS and Information Science— and can minor in CS, Information Science, and Computing in the Arts.

Below, we summarize some of the excellent research ties that CS has with Arts & Sciences faculty.

Astronomy

CS faculty Alan Demers, Johannes Gehrke, and Jai Shanmugasundaram, along with Jim Cordes of Astronomy, have an NSF grant for data management for large-scale astronomical surveys using the Cornell Arecibo radiotelescope, where astronomers are amassing a petabyte of data.

Economics

CS faculty member Joe Halpern collaborates on research in decision theory with economics professors David Easley and Larry Blume. They co-teach the course Decision-Making in Complex Environments.

Science & Technology Studies (S&TS)

Computer scientist and cultural theorist Phoebe Sengers has a joint appointment with S&TS and CIS. Several CS undergrads work in her Culturally Embedded Computing Group, which analyzes, designs, builds, and evaluates computing devices in cultural context. To illustrate the flexibility we have at Cornell, consider the case of Lucy Dunne. As a student in Textiles & Apparel Design in the College of Human Ecology, she did her Masters with Sengers and others on the design of wearable technology; she addressed the human-device interface through functional apparel design. She is now doing her PhD on this topic at the University College Dublin, Ireland.

Cognitive studies program

In the early 1990s, CS professor Bob Constable was a founding member and one of the leaders of Cognitive Studies. CS professor Joe Halpern has been a co-director, and Claire Cardie, Dan Huttenlocher, Lillian Lee, Bart Selman, and Ramin Zabih are connected with the program. The program has almost 80 members representing 18 departments and schools, making Cornell an exciting environment for cognitive studies. Strengths in cognitive psychology, CS, theoretical and experimental linguistics, philosophy of mind and language, and logic lead to extensive interactions in teaching and research.

The logo of the Institute of Social Sciences. The institute brings together about a dozen faculty, mostly from Cornell, to work on a common theme, which is currently computational social science.

Computational Biology

A year ago, CS faculty Ron Elber, Uri Keich, Jon Kleinberg, and David Shmoys were involved, along with 30-odd other faculty, in the creation of a PhD program in computational biology. Elber is on the executive committee of the new program, and CIS professor David Shalloway is the director. Shalloway and Kleinberg also headed the team to develop undergrad programs in computational biology.

Institute for Social Sciences

In April 2005, Cornell's Social Science Advisory Council selected *Computational Social Science: Social and Information Networks* as the second Theme Project of Cornell's Institute for the Social Sciences. The project is led by Michael Macy (Sociology), David Easley (Economics), Geri Gay (Communication), Dan Huttenlocher (CS), and Jon Kleinberg (CS). The project "aims to advance the social sciences at Cornell by tapping the expertise, tools, and skills of network analysts across the university, from computer scientists archiving the Web to social psychologists studying adolescent behavior".

Natural Language Processing Group

CS faculty Claire Cardie and Lillian Lee have teamed with faculty from Psychology, Linguistics, and Philosophy to form the Natural Language Processing Group.



Shih-Ping Han joins.

1974

Al Aho, John Hopcroft, and Jeff Ullman publish their classic text *The Design and Analysis of Computer Algorithms* (Addison-Wesley).

John Dennis and Jorge More publish their landmark paper *Quasi-Newton Methods, Motivation and Theory*. The nonlinear equation solving business has not been the same since they showed just how far you could go with approximate Jacobians.

Jim Donahue,
Charlie Van Loan join.

1975

Gerry Salton's new book receives the Best Information Science Book of 1975 from ASIS: *Dynamic Information Library Processing* (Prentice Hall).

Bob Constable starts the development of PL/CV. Developed over nine years, PL/CV eventually resulted in Nuprl, a system for mathematical reasoning, which is in heavy use today. Thirty PhD students learned how to do research using PL/CV and Nuprl.

Corky Cartwright joins.

1976

Dick Conway becomes series editor for Winthrop Publishers.

Dick Conway and David Gries publish several variations of their intro to programming text.

John Dennis and Charlie Van Loan procure HP-67 programmable calculators. For the first time within the confines of the department, it was possible to execute a stored program.

Juris Hartmanis becomes Chair for the second time.

1977

CS acquires its first computer, a PDP 11/60.

David Gries and student Susan Owicki receive the ACM Programming Languages and Systems Award for their paper *An axiomatic proof technique for parallel programs*, *Acta Informatica* 6 (1976), 319-340. Based on Susan's PhD thesis, this paper introduces interference freedom as the basis for proving parallel programs correct.

Frank Luk, Fred Schneider join.

1978

CS introduces two undergrad degrees: BA in Arts & Sciences and BS in Engineering. CS started with just an MS/PhD program in order to produce PhDs to populate future CS departments.

Mission-critical distributed systems



Cornell researchers in distributed systems have earned a reputation for solving difficult problems in reliable distributed applications and systems. Abstractions—and their implementations—are our product; deployment in real systems is how their impact is measured. By that measure, our impact has been substantial. For example, many of the building blocks used in today’s distributed systems can trace their roots back to research in our department, including the fail-stop processor abstraction, fault-tolerant broadcast, state machine replication, virtual synchrony, and failure detectors.

Cornell-developed systems that employ these abstractions are widespread. One example is CS faculty Ken Birman’s Isis Toolkit, which runs the New York and Swiss Stock Exchanges, the French Air Traffic Control System, and the US Navy’s AEGIS warship. Also, the CORBA fault-tolerance standard is based on Isis-style process-group replication.

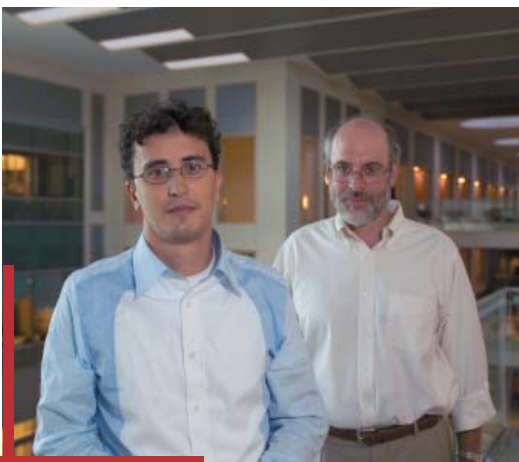
The later Horus and Ensemble systems of Birman and CS researcher Robbert van Renesse have found their way into several well-known systems. IBM’s Websphere product employs a version of Ensemble to implement high availability, and Microsoft’s forthcoming Windows Cluster technology draws on ideas and an architecture first demonstrated in Horus. CS professor Fred Schneider and van Renesse are working with an intranet search engine company to incorporate their “chain replication” algorithm into products running on many large and well-known Web sites.

CS professor Gün Sierer’s work examines the construction of large-scale, resilient infrastructure services based on the newly emerging “peer-to-peer” paradigm, in which clients act as both resource consumers and providers, greatly improving the performance and security properties of the system. Sierer’s work on the CoDoNS system has shown how to build a peer-to-peer name service for the Internet and has been deployed to serve the Internet domain name space for all of China.

Cornell’s strength in this area can be traced to theoretical and practical computer scientists working side by side. Of those, Birman, Schneider, van Renesse, and Sierer remain at Cornell. Dale Skeen left to found a series of companies that specialized in reliable communication infrastructures. Ozalp Babaoglu, Keith Marzullo, Sam Toueg, and Thorsten von Eicken, who built their careers here, are credited with a wide range of seminal contributions to the theory and practice. And, after transforming himself into the world’s foremost expert on scalability while at Cornell, Werner Vogels is now CTO at Amazon.com.

Distributed computing will play an increasingly critical role in the global cyber-infrastructure. The need for trustworthy systems (we use the term in the broadest sense) has received tremendous press and government attention. Massive data centers are appearing everywhere and surging in size. Programmers who used to build software for a single machine are now being asked to port their code to run on distributed platforms and to think about self-management, self-diagnosis, self-repair, and stability under stress.

We study precisely these problems at Cornell. For example, by fusing classical protocol architectures with ideas from peer-to-peer computing (such as “gossip” styles of data replication, which mimic the spread of an epidemic through a dense population), we have obtained high reliability protocols that scale seemingly without limit and offer a wide range of self-managing properties.



Ken Birman’s (right) system Isis Toolkit is used in the NY and Swiss Stock Exchanges. Gün Sierer’s system CoDoNS is being deployed to serve the Internet domain name space for all of China.

“My hope,” says Birman, “is that by integrating these new technologies into Web services, we can offer developers turnkey high-integrity computing. A Web services developer, faced with the need to improve reliability or scalability, would push a button, fill out a property sheet, and see a clustered, scalable solution emerge.”

After 25 years of developing mission-critical distributed systems, we expect to have pulled all the

pieces together into a package that large numbers of developers can master and deploy. “I’ve worked in the field since its inception,” says Birman, “and I’ve never felt closer to finally having the whole story and seeing the technology take off.”

This story illustrates one of Cornell’s enduring strengths: the ability to marshal a broad, sustained response, literally over decades, harnessing skills in both theory and practice.



The TRUST consortium Team for Research in Ubiquitous Secure Technology

Who trusts computers and the Internet? Every day, it seems, a credit-card database is stolen, a new worm or virus is unleashed, military computers are breached, a university computing system is broken into, a phishing scam is exposed—the list goes on and on. Even the breakdown of the Ohio power grid a few years ago was computer-related.

A new \$19 million NSF multi-institutional Center has been created to research this problem of trust and security in cyberspace. TRUST researchers want to develop new technologies that will radically transform the ability of organizations to design, build, and operate trustworthy information systems that control critical infrastructure. Besides figuring out how to protect networks from attacks, they also want to develop ways for systems to “degrade gracefully” when attacked, so that the systems keep running properly. Reliability is also an important issue, because the networks we increasingly rely on are vulnerable not only to intrusion but also to breakdown. The electric power grid is a prime example and will be a test bed for the research.

The proposed research goes far beyond technical considerations; TRUST relies on collaboration with experts in economics, public policy, social science, law, and human-computer interface, for technology developed without attention to these areas risks irrelevance. Cyberspace trust is truly a global, all-encompassing problem. TRUST also has an education and outreach component, geared to K-12 schools, undergraduate students, and institutions serving under-represented populations, which will lay the groundwork for educating new scientists and engineers to develop the next generation of trustworthy systems.

CS at Cornell is poised to take a leading role in TRUST. The Chief Scientist is CS professor Fred Schneider. Other members include CS faculty Ken Birman and Gün Sirer and ECE faculty Stephen Wicker, Rajit Manohar, and Lang Tong.

It’s not surprising that Cornell is playing a critical role. Schneider chaired the National Academy study that produced *Trust in Cyberspace* and is Director of Cornell’s Information Assurance Institute. Moreover, Cornell’s research in mission-critical distributed systems (see the facing page) and language-based security (p. 14) will play a key role in the TRUST effort.

Academic partners

CMU
Cornell
Mills College
San Jose State
Smith College
Stanford
U.C. Berkeley
Vanderbilt

Industrial affiliates

Bellsouth
Cisco Systems
ESCHER
Hewlett Packard
IBM
Intel
Microsoft
Oak Ridge National Lab
Qualcomm
Sun Microsystems
Symantec

Bob Constable and student Mike O’Donnell publish *A Programming Logic* (Winthrop).

Daniel Leivant joins.

1979

Cornell adopts the Cornell Program Synthesizer for instruction in programming. Tim Teitelbaum and student Tom Reps developed this precursor to today’s integrated development environments (IDEs) for teaching a subset of PL/1 on Terak microcomputers, replacing the batch-processing punch-card system then in use. In 1980–1981, the Cornell Program Synthesizer is distributed to 80 institutions.

Gerry Salton becomes Chair of ACM SIGIR.

Bengt Aspvall, John Gilbert, Sam Toueg join.

1980

CS obtains a \$2.6 million, 5-year CER (Coordinated Experimental Research) grant, a major step in increasing its presence in experimental computing.

Ozalp Babaoglu, Paul Pritchard, Dale Skeen, Tom Coleman join.

1981

David Gries publishes *The Science of Programming* (Springer-Verlag), which brings ideas on the formal development of programs to the undergrad level.

Kevin Karplus, Ken Birman join. David Gries becomes Chair. CS grows to 20 faculty.

1982

The 1982 NRC Assessment of Research-Doctorate CS programs places Cornell fifth out of 58 departments.

Bob Constable, with students Johnson and Eichenlaub, publishes a book on their verifier: *Introduction to the PL/CV Programming Logic* (Springer-Verlag).

Gerry Salton receives the first SIGIR Award for outstanding contributions to information retrieval.

Dina Bitton, Greg Johnson, Abha Moitra join.

1983

Tom Coleman publishes *Large Sparse Numerical Optimization* (Springer-Verlag LNCS 165).

CS begins to move into interdisciplinary work, helping to start a new graduate field of “manufacturing systems engineering”.

Language-based security

Cornell has become a leader in a research area that has come to be called language-based security.

Computer security seems like an oxymoron these days, with the Internet providing an easy way to attack computers anywhere in the world. Moreover, virtually all software is designed to be extensible, so you are only a mouse click away from downloading the latest software upgrade or virus—and sometimes it is hard to tell the difference.

The computer security problem has changed dramatically in 40 years. The building-block security properties (confidentiality, integrity, and availability) remain a fundamental part of any solution. The assurance issue also remains crucial—not only must a system be secure, there must be some basis to believe it to be so. But the solution space has decidedly changed due to revolutions in two fields: cryptography and programming languages. Leveraging these developments and further advancing them is the subject of intense activity at Cornell, which has become a leader in a research area that has come to be called “language-based security”.

Forty years ago, the design of programming languages was informed largely by aesthetics and need. A new language design was explored by writing programs for a standard set of problems and writing a compiler so people could use the language. Today, research is focused on program analysis and synthesis agendas, which are applicable to programs in machine-language as well as high-level languages.

- Program analysis methods provide mechanical means to determine whether a program’s execution will satisfy certain properties. The properties might be relative to annotations the programmer provides, as in type checking.
- Program synthesis methods provide mechanical means to ensure that execution will satisfy certain properties, by rewriting a program to capture additional state and add additional checks.

Policy enforcement is an obvious target of opportunity. (An example of a policy one might want to enforce is to limit the number of open windows in a GUI, thereby preventing denial-of-service attacks that succeed by opening countless windows.) Prior to executing a machine language program, analysis and synthesis methods can be used to ensure that the program will not violate its policy. CS profes-

sor Fred Schneider’s work, with student Ulfar Erlingsson, on “in-lined reference monitors” pioneered the idea of using program synthesis to add runtime checks that block execution if a program is about to violate a security policy. Program analysis is then used to delete unnecessary checks prior to execution.

As another example, CS professor Andrew Myers work on enforcing confidentiality and integrity policies employs a mix of synthesis and analysis. Prototyped as an extension of Java, Myers has created an infrastructure that lets programmers use types for defining what hosts in the system are trusted to manipulate the different kinds of data. His system automatically partitions distributed programs into pieces that can safely be executed on each host and generates protocols to coordinate host communication in a way that is consistent with the specified confidentiality and integrity policy.

More surprising than its use in policy enforcement is a role that program analysis and synthesis techniques can play in redefining what constitutes the trusted computing base (TCB) for a system. The smaller the TCB the better, since assurance ultimately comes from people understanding program code, and larger programs are harder to understand. The use of analysis and synthesis techniques in implementing security seemingly adds to the TCB size, but this size increase can be reversed as follows.

An implementation of analysis or synthesis can be instrumented so that it outputs as a formal proof the justification for what that implementation did on a given input program. The formal proof can be bound cryptographically to the input program, resulting in “proof carrying code”. The analyzer or synthesizer in a TCB can then be replaced with a proof checker. Proof checking can be implemented by a small, easy-to-understand piece of code. Thus, the replacement reduces the size of the TCB by replacing a relatively large component by a small one. Questions of efficiency—the size of the proof and the cost of checking—and expressiveness remain active areas of investigation; CS professor Dexter Kozen has been exploring these.

The assurance question is being addressed head-on by CS professor Tim Teitelbaum, whose company GrammaTech (founded with former PhD student Thomas Reps) has developed and successfully marketed a collection of tools to help programmers find vulnerabilities and track the flow of information in C, Ada, and C++ programs. These tools are based on static analysis and other program analysis techniques.

New security defenses lead to new kinds of attacks. Developing specific defenses is important, but keeping ahead of attackers can be unsatisfying because the job is never done. More satisfying is the discovery of general principles about defense mechanisms, and recently this has been aided by applying insights from programming languages. Schneider, for example, has used results from concurrent programming semantics to characterize the class of security policies that can be enforced by in-lined reference monitors. This result not only answered the obvious

Cornell has become a leader in a research area that has come to be called language-based security.

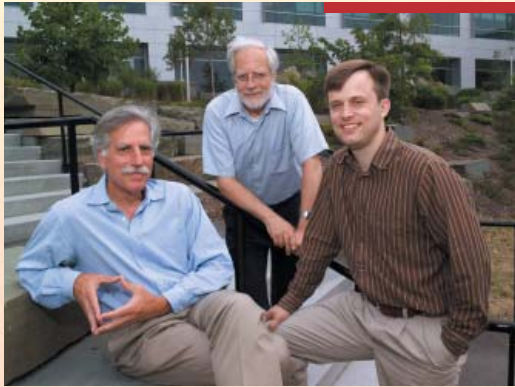
CS professor Fred Schneider’s work, with student Ulfar Erlingsson, on “in-lined reference monitors” pioneered the idea of using program synthesis to add runtime checks that block execution if a program is about to violate a security policy.

Photo: Jon Reis



question about the new security mechanism but defined a research agenda: characterizing what policies can be enforced by various mechanisms. Other mechanisms have since succumbed: static checkers, program rewriters, and so on.

For those who know CS at Cornell, the style of security work reported here will not be surprising. Our systems work is often tied to principles and often addresses problems that transcend technology or specific engineering issues. “Think first, build second” is a succinct characterization of our primary *modus operandi*, and it continues to serve, as exemplified by the impact the security group is having.



Tim Teitelbaum (left), David Gries (center), and Andrew Myers are part of the Languages and Compilers Group in the department.

Programming methodology and program correctness

The 1968 Nato Software Engineering Conference was a wake-up call for the programming world. For the first time, academicians and industrialists spoke honestly and openly about the software crisis —caused by missed deadlines, massive budget overruns, and software riddled with errors. Everyone admitted they did not know how to program and develop software. The conference inspired research in a number of areas, among them, the correctness of programs.

Tony Hoare’s 1969 paper on an axiomatic basis for computer programming provided a foundation for work on correctness by giving a new way to define a programming language —in terms of how to prove a program correct (with respect to a specification) instead of how to execute it.

Cornell got into this field, in terms of education and research, early. The 1973 text *Introduction to Programming*, by CS faculty Dick Conway and David Gries, was the first programming text to take correctness issues seriously, and discussions at Cornell inspired CS professor Bob Constable to begin working on automated proof checking (see p. 36).

Cornell played a significant role in developing approaches to concurrent-program correctness. In 1975, Susan Owicki’s PhD thesis, supervised by Gries, provided the fundamental concept of interference freedom; a follow-up paper by Owicki and Gries received the ACM 1977 Programming Systems and Languages Award. Gries used the theory to give one of the first interesting formal proofs, of an on-the-fly garbage collector. CS professor Fred Schneider, Gries, and PhD

student Rick Schlichting developed algorithms for fault-tolerant broadcasts, while Schneider, with PhD student Bowen Alpern, developed rigorous definitions of liveness and safety properties and proved that any specification can be decomposed into a safety and a liveness property. Schneider’s text *On Concurrent Programming* (1997) provides a comprehensive and rigorous discussion of formal methods for proving concurrent programs correct.

At Cornell, one goal of the work on axiomatic semantics was to learn how a theory of program correctness could influence the programming process —and thus the teaching of programming. Edsger W. Dijkstra, in his monograph *A Discipline of Programming* (1976), gave basic principles and strategies for this. Gries’s text *The Science of Programming* (1981) amplified and brought the ideas down to the undergraduate level. There followed a period of intense activity in honing the principles and strategies for developing programs and in developing and presenting algorithms.

Today, correctness issues have not been integrated into the undergraduate computer science curriculum as much as some had hoped. However, these ideas are receiving renewed attention as trustworthy computing initiatives in industry and government turn the spotlight on questions of assurance (see e.g. pp.13-14). Further, formal program development techniques are more and more used routinely in companies concerned with building high-assurance systems. With a formal verification group and all this interest in security, Cornell’s faculty will undoubtedly remain leading players in this area.

Tom Reps receives the ACM Doctoral Dissertation Award for his PhD thesis, *Generating Language-Based Environments* (MIT Press). Reps, whose advisor was Tim Teitelbaum, is now a Professor at Wisconsin, Madison.

Gianfranco Bilardi, Alexandru Nicolau, John Solworth, Vijay Vazirani join.

1984

The Synthesizer Generator is distributed to over 330 institutions. Developed by Tim Teitelbaum and student Tom Reps, this tool for automating the construction of interactive language-based environments is based on Reps’s 1983 thesis prototype. The Synthesizer Generator was subsequently commercialized and is still in use.

Gene Golub and Charlie Van Loan publish *Matrix Computation* (Johns Hopkins Press).

The CS computing facility serves as the gateway for the entire university to Arpanet and CSnet. CS is instrumental in the university’s Project Ezra to increase the use of computers on campus, with a 5-year, \$8 million grant from IBM.

Prakash Panagaden, Dexter Kozen join.

1985

The Cornell Theory Center, founded in 1984, becomes one of four NSF supercomputer centers. IBM provides an additional \$30 million in hardware, software, and staff.

Ken Birman develops the first version of Isis, the first system for fault-tolerance in distributed systems. Isis has impacted the theory and practice of distributed computing. Two years later, the virtual synchrony model is defined and incorporated.

CS receives its second 5-year NSF CER (Coordinated Experimental Computing) grant.

David Gries receives the AFIPS Education Award for his contributions to computer science education.

Keith Marzullo, Alberto Segre, Keshav Pingali join.

1986

The Nuprl work reaches a milestone: Bob Constable and his students publish *Implementing Mathematics with the Nuprl Proof Development System* (Prentice Hall).

John Hopcroft shares the ACM Turing Prize with Bob Tarjan, “For fundamental achievements in the design and analysis of algorithms and data structures”. The work was Bob Tarjan’s PhD thesis at Stanford, advised by Hopcroft. Their major achievement was a linear algorithm for graph planarity testing, but many more ideas on algorithm design and data structures came out of their collaboration.

A grand challenge in computer networking

Network management — what could be less interesting? The phrase evokes images of hapless IT workers poring over manual pages and typing rows of cryptic management commands, while pulling out their hair trying to figure out why Fred can't connect to the server but Kathy, in the next office, can.

Network management has always been viewed as a black art, understood only by gurus. This guru-centric view hurts most in two areas: first, settings where even the best gurus aren't good enough to control all situations that might arise (such as critical infrastructures requiring reliability measured as "five-nines" or more), and second, settings where there are no gurus at all (home networks). Networks whose management is based on gurus cannot evolve much beyond the abilities of the available gurus, and networks that grow too complex become less reliable, less secure, and more expensive to operate.

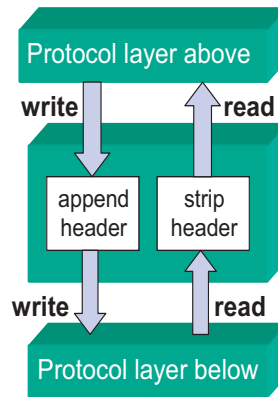
Network management is all about discovery: discovery of boxes and discovery of protocol layers supported by those boxes.

Self-managing networks are the obvious solution, but how to build such a network is far from obvious. IBM, which calls this grand challenge *autonomic computing*, has not reported much progress despite a now three-year corporate focus. The UPnP (Universal Plug and Play) industry forum defines standards for self-configuring devices and PCs, but the standards for basic connectivity alone cover 500 pages!

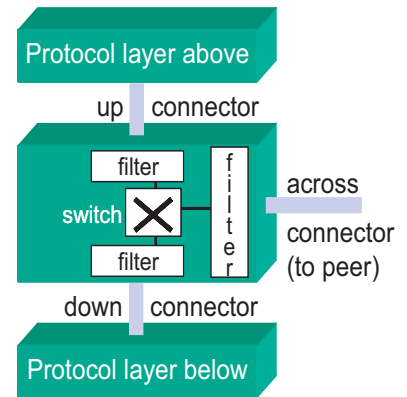
Why is the construction of self-managing networks so difficult? CS professor Paul Francis believes it is because there is no real architectural framework for network management. Architects of the early Internet devised routing protocols only to hold the network layer together. As a result, every vendor was left to implement a distinct style of management. "Network management research is like doing a 5000-piece jigsaw," says Francis. "There are many interesting sub-problems, but putting the whole

puzzle together is more about piecing together hundreds of evolving details and less about elegant solutions." Not surprisingly, many networking researchers have dismissed the problem area as hard but dull.

The Internet has always had an extensible architectural framework for data delivery services: the protocol layering model, which today is taken for granted. The simple idea that each layer adds and strips its own header,



Historical protocol layering abstraction for data delivery



Is there an analogous layering abstraction for network management?

and provides basic services to the layer above, is arguably responsible for much of the tremendous expansion and evolution of the Internet over the last two decades. Is there an analogous extensible framework for network management? Francis believes there might be and that it could lead to a new generation of self-managing networks.

Network management is all about discovery: discovery of boxes and discovery of protocol layers supported by those boxes. Historically, automatic network management protocols performed discovery using the data delivery services provided by protocol layers. "What if," asks Francis, "in addition to supporting data delivery, each protocol layer had some native support for discovery? What would this support look like, and how would it be used?"

All protocol layers appear to have a small number of fundamental structural components in common. These components include connectors to layers above and below as well as across to peers in other devices. They also include filters at each end of the connectors and internal switches that join the connectors together. Given this structural commonality, there should be a small number of operations to manipulate those common components: discovery of potential or realized components, connection of components, and testing of components. With the right set of abstractions, it should be possible to hide most of the gory details associated with the management of a protocol from network managers (human or machine). The network now can be seen as a simple (albeit large) graph of nodes, links, and filters, rather than as thousands of obscurely inter-related protocol parameters.

"This network management solution isn't going to be a quick hack like NAT was," says Francis, "although I hope its impact is at least as dramatic." Network Address Translation (NAT), invented by Francis in the early 90s, is widely acknowledged as having averted the IP address shortage crisis, allowing the Internet to expand beyond its role as an experimental research network. NAT allows a

CS Professor Paul Francis: Who would've thought that network management might be interesting after all?



router to act as an agent between the Internet and a local, or private, network, so that a single unique IP address can represent an entire group of computers. If you have a local network in your home, thank Francis for developing NAT, which made your network possible.

“Who would’ve thought that network management might be interesting after all?” quips Francis. That Francis did is what makes him the networking leader he is today.

Computing in the arts

The influence and potential of computer science extends to the realms of the imaginative and the aesthetic. To encourage students to learn about this role, CS and CIS have worked with faculty in other disciplines to create an undergrad minor or concentration in the College of Arts & Sciences, called Computing in the Arts. This minor is the latest example of how we are expanding opportunities for students by working with forward-thinking professors across the campus.

The concentration in Computing in the Arts: Requires CS 165 and five courses in one track (which may include two from another track to encourage interdisciplinary study). The topics of courses, given below, illustrate some of the infiltration of computing into the arts at Cornell.

Computer Science Track

- Visual Imaging in the Electronic Age
- Computer Game Design
- Computer Graphics
- Artificial Intelligence
- Natural Language Processing
- Machine Learning
- Computer Animation
- Data Mining
- Human-Computer Interaction Design
- Language and Technology

Music Track

- Computer Game Design
- Digital Music
- Computers in Music Performance
- Scoring the Moving Image
- Sound Design and Digital Audio
- Digital Performance
- Improvisational Theory
- Counterpoint
- Composition in Recent Styles
- 20th-Century Musical Languages
- Physics of Musical Sound

Psychology Track

- Visual Imaging in the Electronic Age
- Computer Graphics
- Cognitive Psychology
- Digital Music
- Visual Perception
- Auditory Perception
- Psychology of Music
- Human Perception: Applications to Computer Art and Visual Display

CS faculty member Graeme Bailey, educated as a mathematician and also as a performing musician, piloted the project, along with Steve Stucky (Music, composer, winner of a 2005 Pulitzer Prize) and Carol Krumhansl (Psychology, audio and music perception). The minor currently has tracks in music, psychology, and computer science. A parallel minor in Digital Arts has been approved in the College of Art, Architecture, and Planning.

Supporting the minor is Bailey’s CS 165 course on Computing in the Arts. An innovative mix of formal lecture and studio work, with minimal prerequisites, the course focuses on ideas rather than software packages. Working with poetry, visual art, sculpture, and music, students learn about randomness and stochastic processes, symmetry, structure and group theory, dynamical systems and embedded structures, shape, deformation, and topology—applying all this to actual artistic creations. Future versions will have enhanced perception/cognition content. In the studio environment, students learn to critique each others’ work in constructive ways. Creativity is an integral part of the course.

Cornell is putting resources into research in computing in the arts. Music has hired faculty in computer music. Psychology has freed resources so that Krumhansl can co-teach CS 165, imbuing it with her insights of a lifetime’s work in audio perception. Fine Arts has committed to hiring in digital art. Dance is planning productions involving computer response to visual capture.

Various faculty are distilling their active research into presentations aimed at both undergrads and experts from other disciplines. For example, CIS visiting faculty and CS PhD Fabio Pellacini, previously of Pixar Animation Studios, has given talks on making high-level computer graphics artist-friendly (used centrally in making *The Incredibles* and a new film, *Cars*). A week in April 2004 was *Digital Arts Graphics Week*, with the President and Vice President of Pixar, Senior Vice President of Sony Imageworks, and Mark Levoy of Stanford giving presentations (many of the participants were Cornell graduates). For two weeks in Sept. 2004, Cornell hosted *Perspectives on Digital Music in the 21st Century*, with major composers and leading academics from the US and Europe exploring the past and future of many contributions of computer science in the realm of music and sound.

CS moves into 22,000 additional sq. ft. of new space constructed on top of Upson Hall.

Former PhD student Kurt Mehlhorn and frequent visitor Wolfgang Paul receive the German Leibniz Prize.

David Gries publishes the first of five years of Taulbee Surveys, which give data on PhD-granting departments. The five years of surveys have an almost 100% completion rate.

Bruce Donald, Dave McAllester join. John Hopcroft becomes Chair. CS grows to 25 faculty members and 200 computers.

1987

David Gries chairs the Computer Science Board, the precursor to the Computing Research Association (CRA). This Board was formed in 1972 to provide a forum for the discussion of issues in research and education in computer science.

John Hopcroft chairs the NSF Advisory Committee for Computer Research.

Gerry Salton receives the Distinguished Science Award from the Humboldt Foundation. This foundation, created by the German government in 1953, enables scholars to do research in Germany.

Don Greenberg receives the ACM Steven Coons Award. This highest award in graphics honors lifetime contributions to graphics and interactive techniques.

Ramin Zabih and David McAllester receive the Best Paper Award at the AAAI Conference. McAllester is now Professor and Chief Academic Officer, Toyota Technological Institute at Chicago.

Devika Subramanian, Dan Huttenlocher join.

1988

Juris Hartmanis and John Hopcroft are elected to the National Academy of Engineering.

Gerry Salton is named a Pioneer of Computing in the *Annals of the History of Computing*. He receives the ACM Award for Best Review in Computing Reviews.

Don Greenberg receives the National Computer Graphics Association Academic Award.

Eva Tardos receives the 1988 Fulkerson prize for the paper *A strongly polynomial minimum cost circulation algorithm*.

Fred Schneider takes over as Editor-in-Chief of *Distributed Computing*, and David Gries becomes a Managing Editor of *Information Processing Letters*.

Data mining, today and tomorrow

We leave digital puddles wherever we go. Buy something at a supermarket, and your market basket gets added to the grocery chain's data warehouse for purchase-behavior analysis. Visit a Web site, and your interactions may be used to personalize future interactions. The amount of stored data grows about 30% every year. How can useful information be extracted from this ever-growing ocean of data?

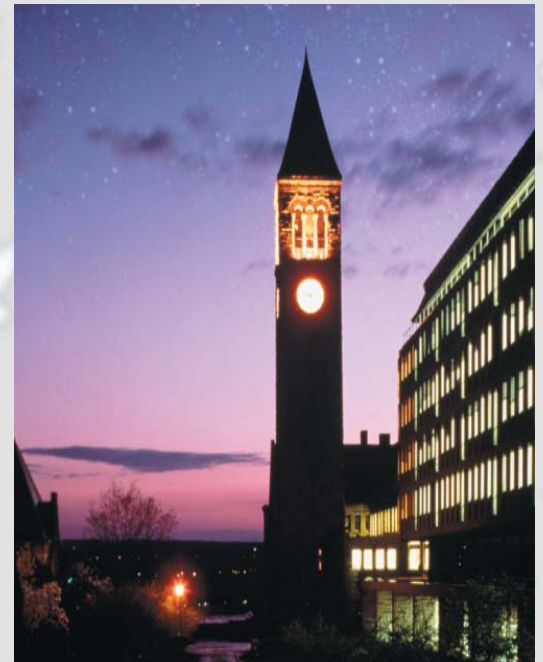
Data mining is the science of discovering new information, such as unknown patterns or hidden relationships, from huge databases; it is the science of finding knowledge that you were unaware of before. "Think of it as database queries on steroids," says CS professor Johannes Gehrke, whose group has developed some of the fastest data-mining algorithms. "Traditional database queries let you specify exactly what records to retrieve. In data mining, the computer finds interesting gold nuggets without you pointing a specific query at them."

Data mining is the science of finding knowledge that you were unaware of before.

Three ongoing projects illustrate different research challenges in data mining. First, data-mining algorithms most often search humongous combinations of possibilities. For example, consider finding out what items shoppers frequently purchase together in

Wal-Mart. Assuming that Wal-Mart stocks a few 10,000 items, there are about 10^{3000} possible combinations of items to investigate! Also, a 100-terabyte database does not fit into memory, and access to data on disk can be five orders of magnitude slower. Fast search and scalability of algorithms are needed; they have been the focus of research in the last decade, leading to much progress on scalable data mining algorithms.

Scalability today presents an enormous challenge. CS researcher Alan Demers and Gehrke are working with Jim Cordes of the Astronomy Department on the design and implementation of an analysis infrastructure for a new census of pulsars in the Milky Way Galaxy. The data will be collected at the Arecibo Observatory in Puerto Rico. "The data rates and processing requirements for the pulsar survey are truly astronomical," says Gehrke. The total raw data, which will take three to five years to acquire, will be about one petabyte—14 terabytes of data will arrive every two weeks via "Fed-Ex-Net" on USB disk

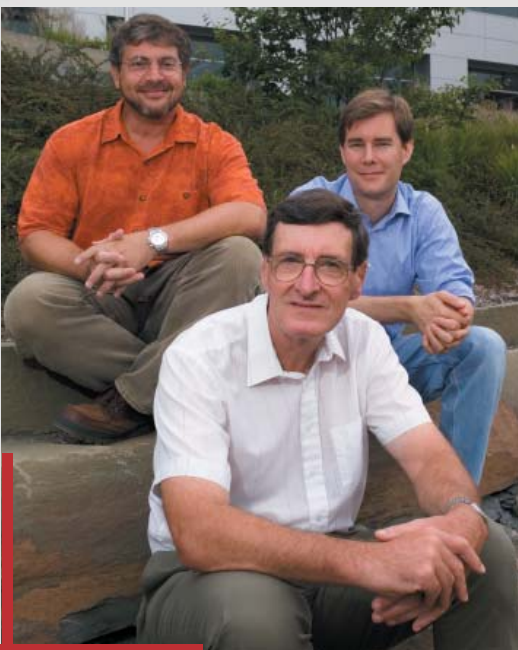


Cornell's Olin Library and McGraw Tower.

packs, requiring the processing of one TB of data per day. A recent \$2M research infrastructure award has allowed the team to build the necessary computing infrastructure at the Cornell Theory Center.

A second challenge is to mine data with missing or wrong entries. CS professor Rich Caruana and researcher Mirek Riedewald are working with scientists from the Cornell Lab of Ornithology on analyzing large citizen-produced datasets. Every year, tens of thousands of volunteers report sightings of birds to the Cornell Lab of Ornithology, creating one of the largest and longest-running resources of environmental time-series data in existence. Its analysis could reveal long-term changes in ecosystems due to human intervention; for example changes in farming practices have been shown to affect bird abundance over time. But mining the data is challenging. Volunteers often leave some entries in bird report forms empty, novice observers may confuse bird species, and other variables such as habitat, weather, human population, climate, and geography have to be considered when estimating the true abundance of a species. "Compensating for bias in the collected data is a major challenge, and each observation could be differently biased," says Caruana.

A third challenge is the enormous complexity of today's databases. For example, consider the Web. CS professors Bill Arms, Gehrke, Dan Huttenlocher, Jon Kleinberg, and Jai Shanmugasundaram are building a testbed that will enable the study of temporal dynamics of the Web over time. The team



Interests in data mining and the Web—itsself a humungous database—bring Rich Caruana (left), Bill Arms (center), and Johannes Gehrke together.

will obtain the 40 billion Web pages archived by the Wayback Machine, the time machine of the Internet. The team will also receive new 20-terabyte snapshots of Web crawls every two months. This collection will enable the research community, for the first time, to evaluate models of Web growth and evolution at a wide range of different time scales. “The combination of content, link structure, and temporal evolution creates an immensely complex dataset,” says Arms. “With this data and associated

data-mining tools, we will be able to tackle really big questions, for example how new technologies, opinions, fads, fashions, norms, and urban legends spread over time.”

“The beauty of working in this area is that you have discovery at two levels,” says Gehrke. “You develop interesting new computer science methods, and you find nuggets by applying these to real datasets.”

The marriage of structured and unstructured data

Most digital documents contain a mixture of structured and unstructured data. For example, online versions of congressional bills in the Library of Congress database contain not only the names, dates, and sponsors of these bills (structured data) but also the text of the bills and hyperlinks to related documents (unstructured data). The same is true for the Internet, which contains database-backed Web sites (structured) and static HTML pages (unstructured). Similarly, online versions of Shakespeare’s plays contain information about acts, scenes, and names of persona (structured) and the text of the play (unstructured).

Current data management systems such as relational database systems and information retrieval systems do not provide a unified way to handle both structured and unstructured data.

Recent advances led mostly by CS researchers are bringing together the separate worlds of structured and unstructured data. CS professor Jayavel Shanmugasundaram and grad student Chavdar Botev, in collaboration with Sihem Amer-Yahia at AT&T Research Labs, have developed a new language and system architecture for managing both kinds of data. The core of their contribution is a new query language, TeXQuery, which enables users to seamlessly query XML documents that contain a mix of structured and unstructured data.

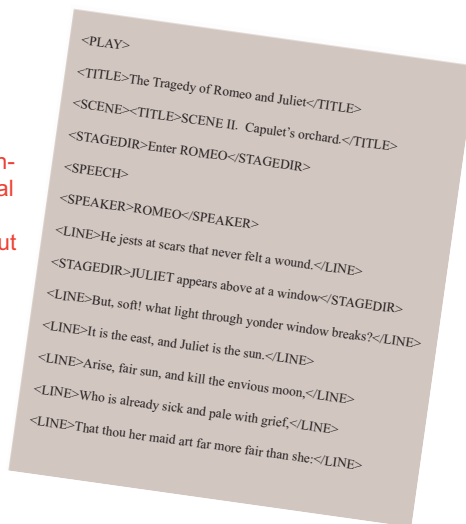
TeXQuery has had a rapid and profound influence on the data management industry. The World Wide Web Consortium (W3C), the body that developed fundamental standards such as HTML and XML, has adopted TeXQuery as the precursor to their standard XQuery Full-Text language for querying structured and unstructured XML data. Shanmugasundaram and Botev serve as invited experts at the W3C to help shape the evolution of the language. Rarely has a research idea been transferred to a standards body in such a short time.



The potential impact of TeXQuery is enormous. The Library of Congress has recently started an effort to convert its data into XML to make it searchable using XQuery Full-Text. All major data management vendors have announced plans to implement the language in future releases of their systems.

Structured and unstructured data are engaged, and their marriage date will be set soon. And CS researchers are the matchmakers!

Jai Shanmugasundaram and his colleagues are the matchmakers for structured and unstructured data.



Ken Birman starts a company based on Isis. Isis is used extensively on Wall Street and in telecommunications and VLSI FAB systems. Today, Isis is still the core technology in the New York Stock Exchange (every trade and every quote since 1993 ...), the Swiss Exchange, the French Air Traffic Control system, the US Navy’s AEGIS warship, and the Florida Electric and Gas SCADA system.

Tom Coleman and Charlie Van Loan publish the *Handbook for Matrix Computations* (SIAM).

Tim Teitelbaum and former student Tom Reps publish two books on the Synthesizer Generator, with Springer-Verlag.

Bard Bloom, Steve Vavasis join.

1989

The Computer Science Board, chaired by Gries, changes its name to the Computing Research Association (CRA), opens an office in Washington, and works to represent the national interests of computing research.

John Hopcroft authors a report for the NSF Advisory Committee for Computer Research (with Ken Kennedy). “Computer Science: Achievements and Opportunities” helps set the direction of the NSF computing research funding.

Gerry Salton is Chair-Elect of Section T of the AAAS (American Association for the Advancement of Science). Section T concerns Information, Computing, and Communication.

Tom Coleman becomes Director of the Cornell Advanced Computing Research Institute, a unit of the Cornell Theory Center. The interdisciplinary institute is concerned with scientific computation research and its application to engineering and scientific problems.

Gerry Salton receives the ASIS Award of Merit, the American Society of Information Science and Technology’s highest honor, bestowed annually to an individual who has made a noteworthy contribution to the field of information science.

John Hopcroft receives an honorary doctorate from Seattle University.

Bob Constable and student Doug Howe publish *Implementing Metamathematics as an Approach to Automatic Theorem Proving* (Elsevier Science).

Gerry Salton publishes *Automatic Text Processing* (Addison Wesley).

Grid computing and Web services

The next time you take a plane, thank a multi-disciplinary team led by CS professors Keshav Pingali and Steve Vavasis and Civil & Environmental Engineering professor Tony Ingraffea for making your flight a little safer.

Most commercial and military aircraft are flown well past their intended lifetime—the B-52 will be flown until 2040 when it will be 94 years old, three times its original planned lifetime. The most critical problem with aging aircraft is wear and tear, which causes microscopic cracks to form in the fuselage and engines, compromising airworthiness.

To simulate how cracks form and propagate in mechanical structures like airframes and rocket engines, the team is using grid computing. “The grid-computing metaphor represents many styles of distributed computing,” said Pingali. “We have shown that some of the more useful styles of grid computing can be done quite effectively using existing industry-standard protocols and software such as SOAP and XML.”

The benefits of using industry-standard Web services became evident while the team was exploring a simulation of fracture in engine components. These components transport high-pressure, high-velocity chemically reacting gases, which can create large thermo-mechanical stresses on component walls. To simulate fracture initiation and growth, the group had to integrate several software systems, including a finite-element mesh generation code developed jointly by Cornell and the College of William and Mary, a chemically reacting flow-simulation code developed at Mississippi State University (MSU) and the University of Alabama, and a linear elastic fracture code developed at the Cornell Theory Center.

“The traditional approach to integrating such software modules is to port them all to a single computing platform,” said Pingali. “Not only is this time-consuming, but every time a new release of a module becomes available, some poor soul has to repeat the entire process of downloading and porting the code, recompiling it, relinking the compiled code with the rest of the software, and so on.”

To simplify the job of integrating software components while respecting individual software developers’ choices of hardware platforms, operating systems, and



Structural failure in Aloha Airlines Boeing 737 (28 April 1988).

programming languages, the team decided to deploy each major component as a Web service running on a server at the institution where the component was developed. The flow-simulation code, for example, runs on an IBM x330 Linux server at MSU, while the fracture simulation code runs on the Cornell Theory Center’s Windows cluster. Industry-standard Web service implementations, such as Apache SOAP and XML-based data exchange formats developed by Vavasis, are used.

“We view the person running the simulation as a client who writes a few hundred lines of code to invoke the various Web services to orchestrate the simulation,” said Vavasis. “Our motto is ‘write once, run from anywhere.’” He said that the overhead of using geographically-distributed Web services for their simulation is about 10 percent.

Gordon Bell, senior researcher at Microsoft’s Bay Area Research Center, said, “This project demonstrates the potential for a new way to build applications and the potential for a new software industry structure based on delivering results. Users don’t have to buy apps programs and maintain a more complex software environment; they simply call a program or database. This is one of the few projects that I would call a Web service, and it is well beyond what is running on today’s experimental grid.”

The team’s next project, being done in collaboration with NASA, is to develop an adaptive control system to protect civilian airliners from attack by shoulder-fired missiles. “A network of sensors and processors embedded in the aircraft structure provides a nervous system for sensing and estimating damage,” said Pingali. “Given this information, the system must perform projections of aircraft performance and adjust flight controls appropriately.”

This is one of the few projects that I would call a Web service, and it is well beyond what is running on today’s experimental grid.

Gordon Bell
Microsoft Research



Keshav Pingali (center), Radu Rugina (left), and Steve Vavasis use grid computing to study fracture in mechanical structures like the portion of a Boeing 747 fuselage that Pingali is holding.

Another ongoing project is focused on providing transparent checkpoint-restart capability for long-running scientific programs in computational grids, which will enable applications to migrate between sites to take advantage of changing resource availability. CS professor Radu Rugina is working with CS senior research associate Paul Stodghill on approaches to reduce the amount of state saved at these checkpoints. This work builds on Rugina's successes in developing sophisticated compiler analyses and transformations to reduce the memory requirements of Java programs and to automatically find memory errors in C programs.

CS chair Charlie Van Loan believes that these projects exemplify the interdisciplinary research style made possible by the collegial atmosphere at Cornell. "Like many schools, we have a strong computational science and engineering program," said Van Loan, adding, "but what is unique here is the collaboration among numerical analysts like Vavasis, practitioners like Pingali and Rugina, and faculty in engineering disciplines like Ingraffea. I cannot think of many other places in the world where this kind of work can be done."

The Cornell Theory Center

The Center for Theory and Simulation in Science and Engineering, or Cornell Theory Center (CTC) as it is known unofficially, was founded in 1984 as one of five national supercomputing centers funded by the NSF to provide high-performance computing to researchers across the country. Ken Wilson, the 1982 Nobel laureate in Physics, was its founding director.

Large-scale delivery of cycles to researchers started on the IBM 3090 series of vector computers. A few years later, CTC deployed one of the largest IBM SP parallel computers in the world. More recently, CTC has pioneered the use of Windows clusters for high-performance scientific computing, with over 2000 processors. Substantial funding for this effort was provided by Microsoft, Dell, and Intel. The results from this project were highly influential in persuading Microsoft to enter the high-performance computing market. CTC is now an important unit of the Faculty of Computing and Information Sciences.

From its inception, the CTC has had strong ties with CS. The original proposal to the NSF to be one of four national supercomputing centers included contributions from CS professors John Hopcroft, Ken Birman, Bob Constable, and Charlie Van Loan.

As CTC's mission evolved under NSF's ten-year supercomputing centers program, the relationship with CS was formalized through establishment of the Advanced Computing Research Institute (ACRI), led by CS professor Tom Coleman. The ACRI's mandate was to apply CS expertise in parallel linear algebra methods, advanced optimization techniques and applications, parallel solution of PDEs, and the study of parallelizing compilers for scientific computing to a range of scientific applications. ACRI members Keshav Pingali, Steve Vavasis, and Charlie Van Loan collaborated with groups in biomechanics, Chemistry, and Civil & Environmental Engineering. The ACRI used one of the first Intel Hypercubes and solidified CTC's reputation among computational scientists and engineers as a pioneer in parallel computing.

Tom Coleman became director of CTC in 1997 and created even stronger CS linkages. During his tenure, Pingali, Ron Elber, and Johannes Gehrke were CTC Associate Directors. Pingali forged a strong alliance with Cornell professor Tony Ingraffea to create the Computational Materials Institute (CMI), which successfully competed for large-scale proposals from NSF's Knowledge and Distributed Intelligence and Information Technology Research programs. Elber led the center's NIH-funded Parallel Processing Resource for Biomedical Scientists and continues to lead CTC's Computational Biology Service Unit; both initiatives have contributed to Cornell's reputation as a powerhouse in computational life sciences.

A major new thrust is data-intensive computing. Led by Gehrke, this project is extending the expertise of the CS database group to an increasingly broad base of collaborators in fields from proteomics to astronomy.

Given this rich history of interdisciplinary work, it is not surprising that CTC plays a central role in the CIS mission to transform intellectual disciplines across the university by making them more computational.



Cornell's Rhodes Hall houses the Theory Center.

1990

With completion of Rhodes Hall, CS expands to 38,000 sq. ft. of space.

David Gries receives the ACM SIGCSE Award for Contributions to CS Education.

David Gries receives the CRA (Computing Research Association) Award for Service to the CS Community.

Juris Hartmanis is elected a Foreign Member of the Academy of Science of Latvia.

John Hopcroft receives an honorary doctorate from the University of Seattle in Washington.

Charlie Van Loan becomes a member of the SIAM Council.

Tom Coleman and Yuying Li publish *Large-scale Numerical Optimization* (SIAM Publications).

Paul Pedersen, Carlo Tomasi, Nick Trefethen join.

1991

The CS research budget tops \$6 million. CS receives an NSF grant on Revitalizing the Computer Science Curriculum and acquires an 8000-node CM-200 data parallel computer.

Don Greenberg is elected to the National Academy of Engineering.

John Hopcroft is appointed to the National Science Board, which oversees the National Science Foundation.

John Hopcroft becomes Chair of the Board of Trustees of SIAM.

Dexter Kozen publishes *The Design and Analysis of Algorithms* (Springer-Verlag).

Steve Vavasis publishes *Nonlinear Optimization: Complexity Issues* (Oxford Science).

Tom Henzinger, Ronitt Rubinfeld join. Juris Hartmanis becomes Chair. John Hopcroft becomes Dean of Engineering.

1992

Dick Conway is elected to the National Academy of Engineering.

Dexter Kozen receives a Prize from the Polish Ministry of Education.

Students Aravind Srinivasan and Alessandro Panconesi receive the Best Student Paper Award at the ACM Symposium on the Theory of Computing.

The science of networks

Complex networks are ubiquitous, and the study of networks is now central to many scientific disciplines. The explosive growth of the Internet and the World Wide Web has led computer scientists to seek ways to manage their complexity and help users navigate their vast information content. Social scientists are confronted by social network data on a scale previously unimagined: datasets on communication within organizations, on collaboration in professional communities, and on relationships in financial markets. Biologists have discovered that network structure that defines pathways of cell metabolism can provide insight into fundamental biological processes. The drive to understand such phenomena has resulted in a new “science of networks” as they arise in the physical world, the virtual world, and society.

Cornell has occupied a central position in this emerging discipline since the beginning.

Much of the action in the science of networks is occurring at the boundary between computer science and the social sciences. Sociology has long been concerned with the processes by which networks form and with the social processes that they support. Game theory and other branches of economics have studied how people behave in network settings. The interaction of these fields with computer science is leading to deep and surprising insights. To illustrate, we examine results in the study of traffic and in network searching.

Commuters create traffic on the network of roads and highways, much as Web surfers create traffic on the Internet. If a central authority were allowed to specify everyone’s route, it could make the average time in the network, accounting for slowdowns due to congestion, as small as possible. But neither commuting traffic nor the Internet has such a central authority, so how can the average time be reduced?

Game theory suggests we consider “selfish routing”, in which each person follows the path that is best for them, given what everyone else is doing. Selfish routing can be thought of as the only “stable” way for traffic to flow; it is the only kind of



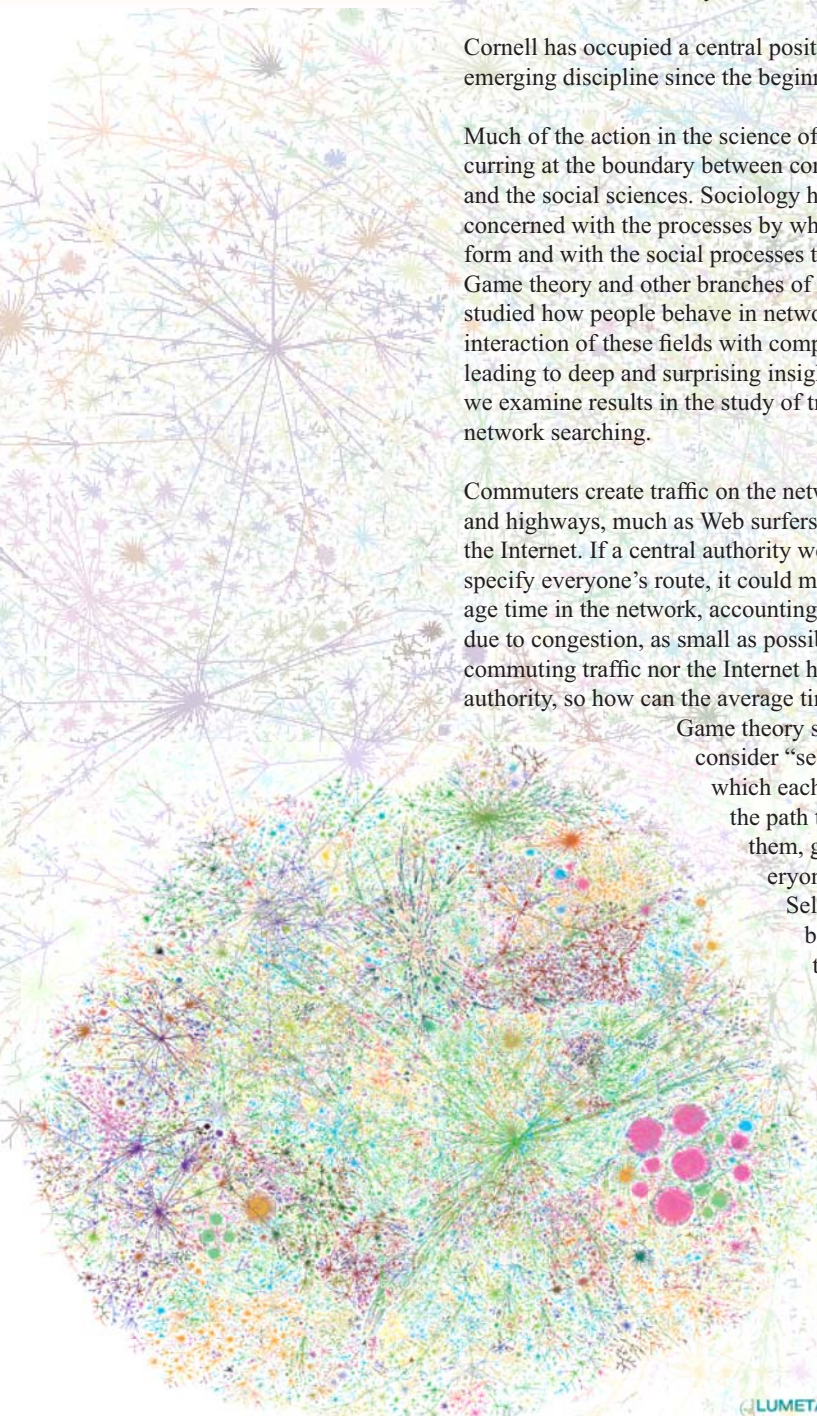
The Internet, pictured below, has inspired research in many areas of CS. Eva Tardos (left) and Jon Kleinberg have made important contributions.

configuration in which no one has an incentive to change routes.

On the Internet, routers mimic “selfishness” by computing shortest paths, but congestion still occurs. To alleviate congestion, is it better to devise more complex routing protocols or just to throw more bandwidth at the problem? CS professor Eva Tardos and student Tim Roughgarden (now a faculty member at Stanford) proved the striking result that travel time under selfish routing is never more than the optimal travel time with twice the traffic. The Roughgarden-Tardos result says something fundamental: simply doubling the bandwidth is at least as beneficial as any amount of central control.

Different kinds of questions arise in social networks—the networks of relationships among people. For example, a popular belief is that people are connected by “six degrees of separation”—an average of six hops in the network that connects people who know each other on a first-name basis. The origins of this notion are rooted in experiments by social psychologist Stanley Milgram in the 1960s. He asked participants to forward a letter to a distant “target person”, with each participant forwarding it to only a single acquaintance. The median length of the paths followed by the letters was six—the origin of the number six in this pop-cultural mantra. This striking experiment inspired researchers to develop network models designed to resemble real social networks in their profusion of very short paths.

CS professor Jon Kleinberg asked a natural question: how could the subjects in Milgram’s experiments, lacking any global knowledge of the social network, be so effective at routing letters, essentially running a collective algorithm for finding shortest paths? In studying this issue, Kleinberg considered a network model developed by Cornell applied math professor Steve Strogatz and his student Duncan Watts. Kleinberg found that, in a variation on the Watts-Strogatz model, it was possible to perform highly efficient message-routing using only local information. The basic structural features of



Kleinberg's networks have since been found in the social structure of large organizations and online communities, and his message-routing approach has appeared in proposals for decentralized search methods in several peer-to-peer file-sharing systems.

Ongoing research at Cornell and elsewhere is seeking answers to other questions such as how

new behaviors spread through large networks, how networks evolve over time, and how their structure shapes and is shaped by interactions among strategic agents. By grounding this research in models that are fundamentally computational, computer science is bringing a new perspective to long-standing questions in a broad array of fields.

CS undergrads do well on the Putnam Math Competition. The team of Kleinberg, Munoz, and Krosky places fifth out of 284, and Zhang places in the top ten individuals.

Juris Hartmanis is Chair of the NRC Committee that produces *Computing the Future* (National Academy Press). This influential report assesses academic computer science and engineering. It advocates a broader research and educational agenda that builds on the field's impressive accomplishments.

Charlie Van Loan publishes *Computational Frameworks for the Fast Fourier Transform* (SIAM).

Juris Hartmanis is elected to the American Academy of Arts and Sciences.

Monika Rauch Henzinger, Thorsten von Eicken join. Bob Constable becomes Chair.

1993

Juris Hartmanis shares the ACM Turing Award with Dick Stearns, "in recognition of their seminal paper, which established the foundations for the field of computational complexity theory" (see the entry for 1965).

Juris Hartmanis receives a Humboldt Foundation Award for Senior U.S. Scientists. This foundation, created by the German government in 1953, enables scholars to do research in Germany.

Researcher Yuying Li receives the 1993 Leslie Fox Prize in Numerical Analysis from the Institute of Mathematics and its Applications.

David Gries and Fred Schneider publish *A Logical Approach to Discrete Math* (Springer Verlag).

Stratus Computer acquires Ken Birman's Isis Distributed Systems, Inc. Isis technology is deployed in the NY and Swiss Stock Exchanges, the French Air Traffic Control System, and other places.

Brian Smith, Claire Cardie, Ramin Zabih join.

1994

Ken Birman becomes Editor-in-Chief of the *ACM Transactions on Computing Systems*.

David Gries receives the IEEE Computer Society Taylor Booth Education Award for his "commitment to education in CS and Engineering as demonstrated by a record of outstanding teaching and mentoring, writing of textbooks, curriculum development ..."

Algorithms at Cornell

At Cornell, research in algorithms began in the early 1970s with a focus on the concept of asymptotic complexity and its role in graph algorithms, efficient data structures, and the abstraction of techniques such as divide-and-conquer and dynamic programming. A prime example of such work is the linear algorithm for testing planarity of a graph, developed by CS professor John Hopcroft and his Stanford PhD student Bob Tarjan (also a former CS professor), for which they received the 1986 ACM Turing Award.

Hopcroft also had a major influence on education and research in the field. His classic text with Al Aho and Jeff Ullman, *The Design and Analysis of Computer Algorithms* (1974), essentially defined the field, and nearly every CS department in the country developed a course around this text.

Cornell faculty members have made important contributions to a broad range of algorithmic domains. Dexter Kozen is widely known for his work on algorithms in computer algebra and symbolic computation and for his text *The Design and Analysis of Algorithms* (1991). Eva Tardos won the Fulkerson Prize in 1988 for her work on network flow algorithms, resolving long-standing open questions of Edmonds and Karp on the efficient solvability of minimum-cost flow problems. David Shmoys's work on approximation algorithms for scheduling and other problem domains was one of the key influences that gave approximability the prominent role it currently enjoys in the area of algorithms. Cornell professor David Williamson won the Fulkerson Prize in 2000 for his joint work with Michel Goemans on the use of semidefinite programming in the design of approximation algorithms.

Today, the field of algorithms has matured and broadened to connect with many other areas, both within computer science and beyond. Jon Kleinberg has led this transformation with his early work on search and information extraction in large networks like the Web; he received the 2001 Award for Initiatives in Research from the National Academy of Sciences for his introduction of link analysis techniques into Web search. Over the years, Kleinberg, Shmoys, Tardos, and Williamson,



Over the years, John Hopcroft (left) and Dexter Kozen have made important contributions in algorithms and complexity.

and more recently Hopcroft, have been leaders in this continuing evolution of the field, through their work on algorithmic issues in social and information networks, on approximation algorithms in discrete optimization, and on algorithmic and game-theoretic foundations for network routing. The new textbook *Algorithm Design* by Kleinberg and Tardos (2005) emphasizes the perspective of this emerging focus through its efforts to situate algorithms at the center of the field of computer science more generally.

Algorithms are prominent in the research of Cornell faculty in many other areas of computer science. Ramin Zabih and Dan Huttenlocher have had a major impact in computer vision through their work using graph algorithms, especially using network flow techniques. These methods are also becoming increasingly influential in graphics. The work of Bart Selman and Carla Gomes has highlighted the key role of algorithms in artificial intelligence through an algorithmic approach to central problems in constraint satisfaction. Johannes Gehrke's work in data mining has established fundamental connections to algorithms. Work in computer security at Cornell includes algorithmic issues, led by Kozen's investigation of efficient code certification.

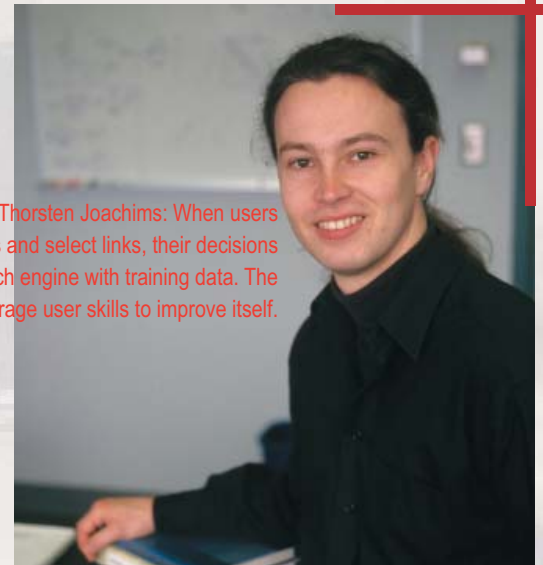
New faculty hire Bobby Kleinberg brings further strength in this area, with his already widely-recognized work on stochastic algorithms and learning-based models for networks and online economic systems.

Through its ongoing role in the evolution of algorithms as a discipline, Cornell moves forward with arguably the strongest algorithms group in the world.

Search engines that learn from experience

Some components of search engines that rank search results need periodic “tune-ups” when the environment changes. An exception is the search engine Osmot, developed by CS professor Thorsten Joachims and his student Filip Radlinski. When fielded on Cornell’s Library Web pages, Osmot tuned itself to this new collection and user base without expert intervention. To avoid making the same mistake twice, Osmot observes how users react to results and uses machine-learning to update its ranking function. For example, it quickly learned that users with the query “oed” wanted to visit the library gateway to the Oxford English Dictionary, even though this page does not contain the word “oed”.

Osmot is an example of how collaborations between CS and Cornell’s new Program in Information Science combine math topics traditionally pursued in computer science with research in human factors. “You can learn a lot about people by watching how they act and react,” says Joachims. “When users reformulate queries and select links, their decisions provide the search engine with training data. The system can leverage user skills to improve itself.”



CS professor Thorsten Joachims: When users reformulate queries and select links, their decisions provide the search engine with training data. The system can leverage user skills to improve itself.

Eye-tracking experiments have provided interesting insights in other computer science areas. In the late 1990’s, Eric Aaron, CS Professor David Gries’s PhD student, performed eye-tracking experiments with Professor Spivey of Psychology to analyze how students developed calculational proofs. The findings confirmed some expected behaviors, based on strategies and principles taught in the Gries-Schneider text *A Logical Approach to Discrete Math*, and uncovered other interesting patterns, such as the tendency to attend to particular premises despite their not being used in the proof under consideration.

The most intriguing property of using observable user behavior as implicit feedback is its availability. It directly reflects individual preferences, and it can be gathered without user effort. So, in principle, search engines need not be one-size-fits-all; instead, they can learn what each user means with their queries. For example, the word “keyboard” in a query from a user at cs.cornell.edu is less likely to refer to a musical instrument than for an average user. A search engine that knows its users from their reactions to the results of previous searches can make better guesses about the meaning of future queries and documents. The better the guess, the better the retrieval quality.

It is not always clear how to interpret user behavior reliably. For example, does a click on a link in the search results really mean that the link is relevant? The answer is “no”, says Joachims, who investigated the question with CIS professor Geri Gay and research associate Helene Hembrooke of Information Science, along with postdoc Bing Pan and grad student Laura Granka. They used eye-tracking experiments to analyze the decision process of search-engine users. Other factors influence clicking behavior, it turns out, most prominently the position in the ranking. “In Google We Trust,” said Hembrooke. “Whenever we moved a link to the top of the ranking, it received more clicks.” While clicks do not indicate relevance on an absolute scale, other interpretations of clicks do give highly accurate feedback. For example, if a user does not click on the top link but instead reformulates the query and clicks on a link there, then, with high probability, the clicked link is more relevant than the top link of the original query.

“We learned in these studies that we can get accurate relative preferences between links but no absolute relevance judgments,” says Joachims. However, most traditional machine learning algorithms can use only absolute feedback. To overcome this problem, they adapted the Support Vector Machine learning method to make use of relative feedback. Here, research on human factors in search-engine use uncovered and directed the need for research on machine learning methods.

The next challenge is to scale these methods to collections of the size of the Web. Retrieval functions that explicitly model all users and sites on the Web will be among the largest machine learning problems ever attempted, involving billions of features and millions of examples every day. But, given that tractability was pushed from hundreds of features and examples 15 years ago to hundreds of thousands of features and examples today, such problems are no longer beyond reach.



Eye-tracking experiments are used to analyze the decision process of search-engine users.

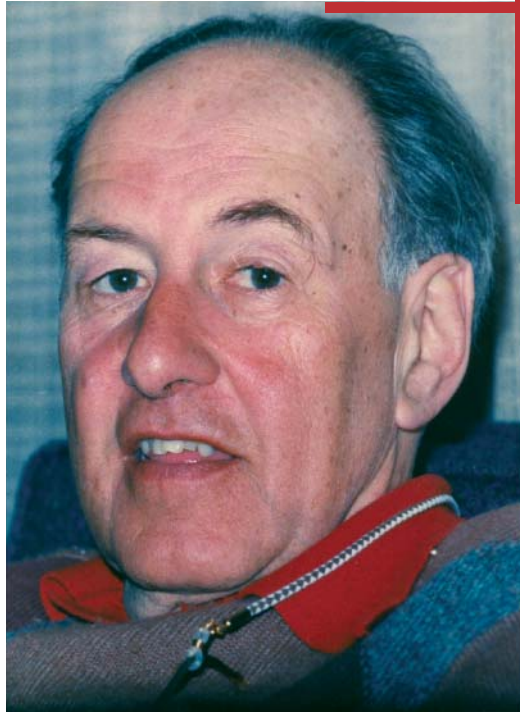
The father of information retrieval

CS professor Gerry Salton is the man most responsible for the creation and coming of age of information retrieval (IR).

Salton published more than 150 research articles and five texts on information retrieval. His honors are too numerous to mention. Among the most prestigious are a Guggenheim Fellowship (1962), ASIS Award for Best Information Science Paper (1970), Best Information Science Book (1975), the first ACM/SIGIR Award for Outstanding Contributions to Information Retrieval (1983), the Alexander von Humboldt Senior Science Award (1988), and the ASIS Award of Merit (1989). The ACM/SIGIR Award was subsequently renamed the Gerard Salton Award. He became an ACM Fellow in 1995.

Salton was information retrieval. At the heart of every IR system, Web-based or otherwise, is the set of keywords and phrases that are collectively used to describe, or index, each document. In stark contrast to the standard indexing approach requiring manual assignment of index terms to texts, he was a very early and vocal proponent of *automatic indexing*. He proposed a scheme in which every word in a document (except for the most common ones) would be used as an index term. This type of full-text indexing technique comprises the core technology in virtually all of today's Internet search engines. Salton's subsequent work addressed, in turn, the critical components of automatic full-text indexing retrieval systems: term weighting, relevance feedback, document clustering, extended boolean retrieval, term discrimination value, dictionary construction, term dependency, phrase indexing, semantic indexing via thesauri, text understanding and structuring, passage retrieval, and even document summarization.

Salton is best known for his vector space model of information retrieval, upon which modern retrieval systems are based; and for the SMART system, his



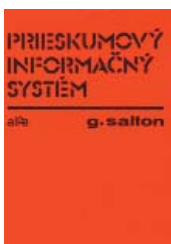
Gerry Salton, 1927-1995

Photo: Professor Edgar Rosenberg

publicly available automatic text processing system, which incorporates the vector space model. SMART, which was known as *Salton's Magical Retriever of Text* (later given the dull interpretation *System for the Manipulation and Retrieval of Text* by more pedantic professors), rapidly matured to the stage where it was the most advanced information retrieval system in the world for many years. It remains a powerful experimental vehicle. Individual, a news clipping service, licensed the technology directly. WAIS (Wide Area Information Server) and DOWQUEST (a tool for the Dow Jones news wire) and others use technology derived from SMART, and many new systems have leveraged his years of research. Today, with the World Wide Web and massive digital libraries, IR has come of age.

The epitaph for Sir Christopher Wren, the architect of St. Paul's Cathedral in London, reads, "If you wish to see his monument, look around you." If you wish to find Salton's monument, use any one of the many text-based search engines for navigating the World Wide Web.

The Salton library



Dan Huttenlocher is the CASE New York State Professor of the Year. The award covers all disciplines. It is given by the Council for Advancement and Support of Education for impact and involvement with undergraduates, scholarly approach to learning, and contributions to undergraduate education.

David Gries receives a Cornell Presidential Weiss Fellowship for his contributions to undergrad education. Three such awards are given each year; Cornell has 1600 faculty members.

T.V. Raman receives the ACM Doctoral Dissertation Award for his PhD thesis, *Audio System For Technical Readings* (Springer-Verlag, 1998). Raman's advisor was David Gries. Raman is now a researcher at Google.

Researchers Jim David, Dean Krafft, and Carl Lagoze release Dienst, which becomes the foundation for future digital library interoperability.

Eva Tardos, Joe Halpern, Jon Kleinberg join.

1995

CS mourns the passing of Gerry Salton, a founding member of the department and the father of information retrieval.

David Gries receives the ACM Karlstrom Outstanding Educator Award. The citation reads, "His visionary emphasis on critical thinking and mathematical precision has dramatically changed the face of computer science education"

David Gries receives an honorary doctorate from Daniel Webster College.

Fred Schneider becomes Professor-at-Large at the University of Tromso, Norway.

Juris Hartmanis receives the Bolzano Gold Medal of the Academy of Sciences of the Czech Republic for Merit in the Field of Mathematical Sciences.

Juris Hartmanis receives an honorary doctorate from the University of Dortmund.

Ken Birman chairs a DARPA ISAT study on survivability of critical infrastructure; Fred Schneider is on the committee. The study establishes a major DARPA effort in the area and lays the groundwork for a broader government engagement of the challenge.

Neil Immerman (former student of Juris Hartmanis) and Róbert Szelepcsényi get the Gödel prize for their paper showing that nondeterministic logarithmic space is closed under complement.

In the novel *Hard Times*, Charles Dickens described the fictional “Coketown” as follows:

Fact, fact, fact, everywhere in the material aspect of the town; fact, fact, fact, everywhere in the immaterial. The M'Choakumchild school was all fact, and the school of design was all fact, and the relations between master and man were all fact, and everything was fact between the lying-in hospital and the cemetery, and what you couldn't state in figures, or show to be purchasable in the cheapest market and salable in the dearest, was not, and never should be, world without end, Amen.

In real life, facts are important, but opinion also plays a crucial role. A computer manufacturer, disappointed with low sales, asks itself: Why aren't consumers buying our laptop? The Democratic National Committee, disappointed with the last election, wants to know on an on-going basis: What is the reaction in the press, newsgroups, chat rooms, and blogs to Bush's latest policy decision?

Answering these questions requires focusing on subjective judgments (e.g. the design is tacky, the administration ignored previous treaties) while taking into account misperceptions (e.g. updated device drivers aren't available), the effect of indirect reporting (e.g. Bush assured the crowd that European support was broad), and the existence of possibly conflicting opinions from the same person or organization.

CS professors Claire Cardie and Lillian Lee are working on sentiment-analysis technologies for extracting and summarizing *opinions* from unstructured human-authored documents. They envision systems that (a) find reviews, editorials, and other expressions of opinion on the Web and (b) create condensed versions of the material or graphical summaries of the overall consensus.



Indeed, the Cornell Natural Language Processing group has done seminal work in developing algorithms for sentiment classification and extraction problems, and its research has been widely recognized in the research community and in the scientific popular press as being, in large part, responsible for the recent huge surge of interest in the area.

CS Professors Claire Cardie (left) and Lillian Lee (above): Be cautious when you hear, “It is a fact that ...”; the phrase is highly correlated with the introduction of an opinion rather than a fact!

Over a dozen external groups have written papers using the so-called Cornell movie-review dataset as a benchmark.

Problems considered by the group include the following: determine whether a document or portion thereof is subjective, determine whether the opinion expressed is positive or negative, determine the strength of the sentiment (e.g. is France *really* or just mildly unhappy with Bush?), find the sources of the opinion (the person, group, report, etc.), and determine whether the opinion is being filtered through indirect sources (e.g. as “Bush” took the liberty of attributing an opinion to “Europeans” in the example above). At first glance, this might not appear so hard. For example, can't one just look for obvious *sentiment indicators* —words like “great”?

The difficulty lies in the richness of human language use. The amazingly large number of ways to say the same thing (especially, it seems, when that thing is a negative perception) complicates the task of finding a high-coverage set of indicators. Furthermore, the same indicator may admit several different interpretations, as the following sentences show:

- This laptop is a great deal.
- A great deal of media attention surrounded the release of the new laptop model.
- If you think this laptop is a great deal, I've got a nice bridge for you to buy.

Each of these sentences contains the phrase “a great deal”, but the opinions expressed are, respectively, positive, neutral, and negative. The first two sentences use the same phrase to mean different things. The last sentence involves sarcasm, which, along with related rhetorical devices, is an intrinsic feature of texts on blogs, newsgroup postings, and, more generally, opinionated text.

Researchers have adopted basically two approaches to meeting the challenges of sentiment analysis. Many

groups are working to incorporate linguistic knowledge; given the subtleties of natural language, such efforts will be critical to building operational systems.

Cardie and Lee pursue a different tack: they employ learning algorithms that can automatically infer from text samples what word-level indicators and phrase-based syntactic and semantic patterns are useful for sentiment analysis.

Learning systems are potentially more cost-effective, more easily ported to other domains and languages, and more robust to grammatical mistakes. Furthermore, they can discover indicators and patterns that humans might neglect. Lee's group found, for example, that, in certain types of text, the phrase "still," (comma included) is a better indicator

of positive sentiment than "good" —a typical use is, "Still, despite these flaws, I'd go with this laptop." And, Cardie and her collaborators at the Universities of Utah and Pittsburgh found that the pattern "It is a fact that ..." is highly correlated with the introduction of an opinion rather than a fact!

Given the multitude of potential applications, researchers like Cardie and Lee have been devoting more and more attention to sentiment analysis. If they continue to be successful, their systems could save information analysts from having to read and summarize potentially hundreds of documents for each topic of interest and would save analysts at the aforementioned laptop manufacturer from having to read potentially hundreds of versions of the same complaints. Surely that sounds like a great deal!

Srinivas Keshav, Greg Morrisett, Praveen Seshadri, David Shmoys join.

1996

Don Greenberg receives the ASCA Creative Research Award in Architecture.

Dan Huttenlocher receives a Cornell Presidential Weiss Fellowship for his contributions to undergraduate education. Three such awards are given each year; Cornell has 1600 faculty members.

David Gries receives an honorary doctorate from Daniel Webster College in New Hampshire.

Bruce Land gets first place in the instructional materials (Web-based) competition of the ACM SIGUCCS Use Services Conference XXIV. The award was for the Web site for his graphics programming course: <http://instruct1.cit.cornell.edu/courses/cs418-land>.

Joe Halpern becomes Editor-in-Chief of the *Journal of the ACM*.

Graeme Bailey, Lillian Lee, Bart Selman join. CS grows to 30 faculty and has over 500 computers.

1997

Juris Hartmanis takes a two-year leave to serve as Assistant Director of the NSF for CISE. During his tenure, he effectively positions NSF and CISE to assume a leadership role in response to the PITAC report, and he is instrumental in shaping the discussion that lead to NSF's playing the lead role in the Information Technology Research (ITR) program.

Joe Halpern shares the 1997 Gödel Prize with former student Yoram Moses. Their paper *Knowledge and Common Knowledge in a Distributed Environment*, says the citation, "provided a new and effective way of reasoning about distributed systems".

David Shmoys becomes Editor-in-Chief of the *SIAM Journal of Discrete Mathematics*.

The faculty publish six books: Ken Birman, *Building Secure and Reliable Network Applications* (Prentice Hall).

Srinivas Keshav, *An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network* (Addison-Wesley).

Dexter Kozen, *Automata and Computability* (Springer-Verlag).

Fred Schneider, *On Concurrent Programming* (Springer-Verlag).

Nick Trefethen and student David Bau, *Numerical Linear Algebra* (SIAM).

CURIE comes to CS

The CURIE Academy is a one-week Cornell residential program for high school girls who excel in math and science. The students work in teams on a carefully formulated project designed to develop their problem-solving skills and immerse them in an interdisciplinary topic. There is, of course, time for introductions to many other areas of Engineering at Cornell and for non-academic fun.

Over the past eight years, the CURIE program has drawn a strong and diverse applicant pool from across the country. Most of the students go on to top-ranked colleges for science and engineering. Many past CURIE members who come to study at Cornell act as undergrad facilitators and mentors to the new crop.

CS faculty Graeme Bailey and Daisy Fan are on the board that oversees CURIE. In 2000, Fan headed up a highly successful CURIE project on environmental systems modeling, with challenging engineering and ethical problems to engage them. In summer 2005, the focus was on computer graphics, and around 40 high school girls did a project with CS faculty Kavita Bala and Steve Marschner.

In 1865, Ezra Cornell said, "I would found an institution where any person can find instruction in any study." More girls than ever are studying engineering and science high above Cayuga's waters, thanks to the CURIE program.



I had a wonderful experience at the CURIE Academy. [It] enabled me to spend a week with the most incredible girls who share my passion for math and science.

~ Isabelle Puckette
16, Encinitas, CA

Daisy Fan (left) and Graeme Bailey are on the Board of Directors of the CURIE program.

Beauty is skin deep

You're standing in a swanky hotel ballroom next to a ten-foot Oscar statue, chatting with computer scientists, who seem a little uneasy in their tuxedos. TV crews and reporters gather behind velvet ropes waiting for the chance to get a sound bite from a famous guest. Cameras flash. It's two weeks before the Academy Awards, and you are at the presentation ceremony for the Academy's Scientific and Technical Awards, which honor the people who work magic behind the scenes. These people have revolutionized their particular niches of the film industry, devising new methods for cinematography, optical engineering, electronics, and film production. In recent years, a growing cohort of computer scientists have joined them, as computer graphics becomes fundamental to moviemaking.



CS professor Steve Marschner knows just how this feels. He accepted a Technical Achievement Award from the Academy of Motion Picture Arts and Sciences in 2004 for his research in simulating subsurface scattering of light in translucent materials. "Winning the academy award was an amazing experience," says Marschner. "To see our ideas picked up by industry and put to use in mainstream movies to solve real artistic problems is exciting and gratifying." The mathematical model devised by Marschner and his colleagues led to the realistic skin of characters like Gollum in *The Lord of the Rings* trilogy and Dobby in *Harry Potter and the Chamber of Secrets*; it also permitted computer-generated versions of characters to stand in for The Terminator in *Terminator 3: Rise of the Machines* and Agent Smith in *The Matrix Reloaded*.

According to Richard Edlund, chair of the Academy of Motion Picture Arts and Sciences' Scientific and Technical Awards Committee, "this is one of the holy grails of computer graphics. One of the difficulties of creating lifelike characters in the computer world is the problem that skin is not opaque. If you render a faithfully scanned or created character and the skin is opaque, it doesn't look real."

Top right, Marschner (right) and colleague Pat Hanrahan enjoy the Oscar awards ceremony. Below, Gollum, of *The Lord of the Rings*, as created using the award-winning realistic skin model.



In 2001 at Stanford, Marschner (then a postdoc at the Computer Graphics Laboratory) and his colleagues, Henrik Wann Jensen, Marc Levoy (a Cornell PCG alumnus), and Pat Hanrahan, developed a model for how light penetrates a translucent material like skin and scatters through the material below the surface before re-emerging. Because skin is translucent, this subsurface scattering model simulates the soft appearance of skin very successfully. Previous models, which tacitly assumed that skin was entirely opaque, resulted in characters with a plastic appearance. The new model was so important in bringing digital characters to life that, within two years after their paper appeared, all the major special-effects studios had incorporated it into their rendering systems.

The subsurface scattering model is based on mathematics that goes back many decades. "The basic math behind our model was originally developed in the 1940s and 1950s for astrophysics; more recently, it was extended by people working in medical physics and doing simulations of laser light in skin and other tissues," says Marschner. "We extended it again to make it work for graphics; now translucency is part of the basic toolkit everyone uses for representing materials."

At Cornell, Marschner has continued his work on realistic rendering of natural materials. Research on rendering human hair led to a model that is based on the realization that hairs are basically thin transparent cylinders. The model, which matches measurements of light scattering from individual hair fibers, is being used in the movie industry to create believable hair for computer-generated characters. In looking at the appearance of wood, Marschner considered its translucent, fibrous structure and implemented a rendering model to replace the old model that assumed wood was opaque.

Marschner is a member of Cornell's Program of Computer Graphics (PCG), a leader in research in synthesizing realistic images.

“It’s simply a pleasure to be part of such a strong lab,” says Marschner. “I feel lucky to be working in this niche. I’m a visual person, and to be able to spend my time scrutinizing the world around me, trying to understand why it looks the way it does, is very rewarding. And to find myself in front of an audience accepting an Oscar —well, that’s like something out of a movie.”

Don Greenberg (Director of the PCG, center), Kavita Bala (right), and Steve Marschner work on synthesizing realistic images.



Program of Computer Graphics

The Program of Computer Graphics (PCG) is an interdisciplinary center dedicated to the development of interactive computer modeling and rendering techniques and their applications. It enjoys close connections with the CS Department, making it easy for CS graduate students to do research in graphics. CS faculty involved in the PCG are Director Don Greenberg, Kavita Bala, and Stephen Marschner.

Established in 1973, the PCG has enjoyed NSF support for more than three decades. Prior to its founding, joint research in computer graphics was conducted with GE’s Visual Simulation Laboratory in the late 1960s, resulting in the movie *Cornell In Perspective*.

The major long-term goal has been to create simulations that are physically accurate and perceptually indistinguishable from real-world scenes. Currently, the research focus involves three-dimensional modeling and rendering of complex environments for realistic image synthesis. Research is being conducted on light reflection models, methods for rapidly determining the interaction between diffusely reflecting surfaces, parallel processing strategies, micro-geometry surface modeling, perceptual issues in graphics, and a host of other topics related to complex modeling and realistic image displays.

The high scene complexity required for realism makes scalability an important problem. Bala’s research addresses this challenge by developing scalable algorithms and representations for illumination, rendering, and modeling. One key insight is that limitations of the human visual system can be exploited to achieve scalability without compromising perceived image quality.

In architecture, research continues on developing methods for interactive computer-aided design, particularly at the preliminary design phase. In medicine, new methods are being developed to display real-time four-dimensional volumetric information and to extract mathematical surface representations of complex body organs.

The PCG serves as the primary focus for computer animation. Two courses are offered, and the PCG is involved in efforts to establish undergrad and graduate majors in digital arts and graphics. Animation research is developing better methods for character modeling and rigging. Many currently used strategies for physically-based motion were developed at the lab.

Close working relationships with industry have resulted in the donation of tens of millions of dollars worth of equipment, making the center one of the most advanced in the United States. The PCG provides a unique opportunity for scientific exploration in computer graphics and parallel processing, as well as interdisciplinary research in computer-aided design.

Throughout its more than three-decade history, the PCG has played a significant role in the computer graphics community. Two articles were published in *Scientific American* (1974, 1991). Greenberg won the prestigious Stephen Coons Award in 1987 and was the founding director of the five-university NSF Science and Technology Center for Computer Graphics and Scientific Visualization. Five faculty and students have won the SIGGRAPH achievement award, and five have received Hollywood’s technical Oscars. Many of its hundreds of graduate students are now faculty at the best universities in the world, and others have leadership roles in the software and animation industries.



“Lightcuts” (by Bala, Greenberg, and fellow researchers) enables highly scalable rendering of complex scenes even with millions of lights.

Charlie Van Loan, *Introduction to Scientific Computing: a Matrix Approach Using MATLAB* (Prentice Hall).

Bill Arms, Andrew Myers, Ron Elber join.

1998

With CS providing leadership, Cornell starts the Faculty of Computing and Information Science, to provide a home for interdisciplinary computing work of all kinds. CS, the Program for Computer Graphics, and Digital Libraries are part of it.

David Gries receives an honorary doctorate from Miami University.

Juris Hartmanis receives an honorary doctorate from the University of Missouri.

Pedro Felzenszwalb is the CRA Outstanding Male Undergraduate Awards Runner-up.

David Liben-Nowell receives an Honorable Mention in the CRA Outstanding Male Undergraduate Awards competition.

Bill Arms becomes Chair of the ACM Publications Board and Editor-in-Chief of *D-Lib Magazine*.

Joe Halpern is founder and administrator of CoRR (the ACM-sponsored Computing Research Repository).

Fred Schneider is Associate Editor-in-Chief of the IEEE journal *Security and Privacy*.

Fred Schneider is Chair of the NRC committee that produces the report *Trust in Cyberspace* (National Academy Press). This report assesses the state-of-the-art procedures for constructing trustworthy networked information systems and proposes directions for research in computer and network security, software technology, and system architecture.

Tom Coleman becomes the Director of the Cornell Theory Center.

Jon Kleinberg publishes his Web-search work on using hubs and authorities. The research is credited, together with the Brin-Page work on PageRank, with forming the basis for the current generation of Internet search tools.

Johannes Gehrke, David Schwartz join. Bob Constable becomes Dean of the Faculty of Computing and Information Science. Charlie Van Loan becomes Chair.

1999

Don Greenberg receives an honorary doctorate from the New Jersey Institute of Technology.

Structural fingerprints of molecular evolution

The DNA molecule encodes all information required to create and sustain life. Mutations to DNA can dramatically alter the appearance, adaptation to the environment, and health of organisms. Mutations have produced the fantastically large number of species—estimated at five to sixty million—we see on earth. How many more variations can there be of DNA molecules? Are there meaningful constraints on the diversity of the genetic code?

Computational biologists like CS Professor Ron Elber are attempting to answer such fundamental biological questions using computational methods, starting at the smallest scale of biological activity. DNA codes proteins, which are the prime molecular machines of the cell. Proteins are linear polymers of amino acids. Since the amino-acid sequence determines all protein properties, including its three-dimensional shape, most evolutionary studies of proteins have focused on comparing sequences of amino acids.

Elber's group is taking a different tack.

"Sequence comparison is effective in detecting closely related evolutionary changes, but it is not the best when remote evolutionary pairs are considered," says Elber. "An interesting empirical observation is that protein structures are better preserved than

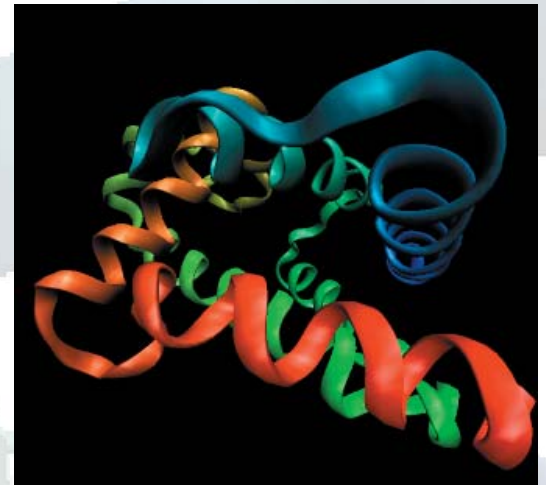
their sequences. Nature is using the same fold again and again to produce protein variants with comparable structures and biochemical properties, so my group uses structural information for detecting remote evolutionary relationships."

This idea of using protein structure rather than amino-acid sequences to uncover evolutionary patterns has led to some major breakthroughs.

In 2000, Steve Tanksley (Plant Breeding) and his co-workers found a gene that controls the size of the tomato fruit, but the evolutionary relationship of this gene with other known genes could not be identified based on sequence similarity. So they called Elber for help. Using his LOOPP software for matching protein sequences to shapes, Elber was able to determine within a few minutes that the tomato gene was remarkably similar to a human gene that controls cell division and growth (in fact, some human cancers result when this gene malfunctions).

"It is astounding that this kind of research could be done at the speed at which it was done," says Tanksley. "This would have been impossible just a few years ago."

Besides clarifying the molecular mechanisms that control tomato size, the study proposed an evo-



The evolutionary capacity of about 4,000 proteins was determined. The capacity of the above protein (*Ascaris hemoglobin*) is 10^{190} .

lutionary pathway from the wild tomato to the domestic fruit.

Such successes on empirical data have led Elber and CS Professor Jon Kleinberg to ask a more theoretical question: how many distinct protein sequences can fold into a particular 3-dimensional shape?

This number, which they call the *evolutionary capacity of the protein*, can in principle be very large (since there are twenty types of amino acids, the sequence space of a protein of length L , $L \approx 100-1000$, is 20^L), so it cannot be determined in reasonable time by direct enumeration.

Kleinberg, Elber, and their students Leonid Meyerguz and David Kempe have found that the problem is closely related to the well-known Knapsack problem, and this connection has led them to invent a fast randomized algorithm for estimating the evolutionary capacity of a protein. The figure on this page shows the protein *Ascaris hemoglobin*, which the

Nature is using the same fold again and again to produce protein variants with comparable structures and adjusted biochemical properties.



CS Professor Ron Elber: Sequence comparison is effective in detecting closely related evolutionary changes, but it is not the best when remote evolutionary pairs are considered. An interesting empirical observation is that protein structures are better preserved than their sequences.

team estimates has an evolutionary capacity of about 10^{190} .

“Using this algorithm, we have studied the evolutionary capacity of all known protein shapes,” says Kleinberg. “The capacity of existing shapes is vastly larger than sequence spaces already explored by nature, suggesting

that currently, it is not a limiting evolutionary factor.”

Interestingly, it appears that proteins occurring in nature are not optimal from a structural point of view —Elber and Kleinberg found alternative sequences that led to similar but more stable structures. Have they improved

on Mother Nature? Or are there additional factors that explain why Nature has chosen to construct proteins the way she has?

These kinds of fundamental questions about the nature of life on earth will keep computational biologists busy for many years.

Juris Hartmanis receives the CRA Distinguished Service Award for his service in the areas of government affairs, professional societies, publications, conferences, and leadership, which had a major impact on computing research.

Dexter Kozen is the Class of 1960 Scholar, Williams College.

Joe Halpern is named the Milner Lecturer at the University of Edinburgh.

Greg Morrisett and students Steve Zdancewic and Dan Grossman receive the Best Paper Award in the European Association for Programming Languages and Systems Conference on Principles, Logics, and Implementation of High-Level Programming Languages.

Keshav Pingali and his students receive the Best paper Award at the International Conference of Supercomputing.

Former students John Belzaira and Julian Pelenur sell their company, Theory Center, Inc. The one-year-old company, a leading provider of Java Beans, was sold to BEA Systems for \$100 million.

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

Johannes Gehrke publishes the second edition of *Database Management Systems* (McGraw Hill), with Ragu Ramakrishnan.

David Schwartz publishes *Introduction to UNIX* (Prentice Hall) and *Introduction to Maple* (Prentice Hall).

Under the leadership of Tom Coleman, the Cornell Theory Center opens the Financial Solutions Center on Broad Street in Manhattan.

Bart Selman's work on phase transitions and complexity is featured in *The New York Times*.

Gün Sirer, Golan Yona join.

2000

Ramin Zabih receives a joint appointment with the Cornell Medical School, the first such joint appointment at Cornell.

Jon Kleinberg receives the Best Paper Award, ACM Symposium on Principles of Database Systems.

Eva Tardos is elected to the American Academy of Arts & Sciences.

Juris Hartmanis receives the Lielo Medal from the Latvian Academy of Sciences. This highest award given by the Academy to scientists of Latvia and of foreign countries is for outstanding creative contributions.

Bridging the Rift: Promoting research and education and peace

On 9 March 2005, the cornerstone of the *Bridging the Rift* Center (BTR) was laid in the desert, 43 miles south of the Dead Sea, between Israel and Jordan. The Cornell president and others, including CS's Bob Constable, took part, as did the Israeli and Jordanian ministers of education, the Israeli finance minister, the Jordanian minister of planning and international cooperation, and Mati Kochavi of the Bridging the Rift Foundation, which is providing the seed money.

BTR will be a life sciences research complex, created to educate grad students from both sides of the border, on 150 acres donated by Israel and Jordan. Israeli and Jordanian students will study side by side, along with grad students from Cornell and Stanford. Cornell and Stanford, substantial partners in this venture, will offer doctoral degrees at BTR, and their faculty will participate along with faculty from Israel and Jordan.

The main research project of BTR is the *Library of Life*. The goal is to assemble a digital catalog and living samples of all microbe, fungi, plants, insects, vertebrates, and invertebrates in the region, creating a Library of the Desert. Cornell professor of Plant Breeding Steven Tanksley conceived the idea. CS professor Ron Elber is the Director of the Library of Life.

The Library is expected to be a global research and education resource, but this will require the development of novel search, analysis, and modeling tools. Because of this, other CS faculty will be involved,

including Rich Caruana, Johannes Gehrke, Dan Huttenlocher, Uri Keich, and Jayavel Shanmugasundaram.

BTR is becoming one of the most prominent and positive programs in the Middle East. King Abdullah II of Jordan described BTR as “bigger than Jordan and Israel”, and Prime Minister Sharon of Israel identified it as being of “first rank strategic

importance for the region”. By spearheading this project, our department is not only doing excellent research but is contributing in a small way to peace in a troubled part of the world.



BRIDGING THE RIFT Feb 23 2004

SOM



The impact of computing on medicine

The discipline of medicine, which is as old as history, is being heavily affected by a new-comer among disciplines: computer science. Simply keeping every patient's health records available in computer-readable form will have a major impact on society and the economy. The Institute of Medicine estimates that medical errors kill as many as 100,000 Americans per year, and many of these errors could be easily avoided using electronic health records.

Stored electronic health records would allow data mining techniques to be used to discover new truths. For example, suppose the majority of patients over 65 who show up in a given period in the ER with a fever above 102 turn out to have an infection that is vulnerable to a particular antibiotic. Data mining might be able to find this nugget of information and thus improve patient care.

CS professor Ramin Zabih, collaborating with radiologists at Cornell's Weill Medical College in New York City, is working on another area in which computing is having a significant effect: Magnetic Resonance (MR) and its use in radiology.

The output of an MR scanner is incomprehensible until it is transformed into an image by a computer program. In fact, most MR scanners are simply an I/O peripheral (which costs millions of dollars and weighs up to 10 tons) connected to a PC. So, even to create an MR image requires computers. But there is more.

The major limitation of MR is in handling motion. A three-minute MR scan can be used only on parts of the body that can stay stationary for that long, but body motion is surprisingly common. Even parts that would seem easy to stabilize, like the feet, turn out to be difficult to keep still for very long. Moreover, obvious motion, such as of the heart and lungs, induces motion in nearby areas. If MR could be used to image the heart, it could screen for coronary disease long before symptoms develop (40% of heart attack deaths happen to people who had no prior symptoms of coronary disease).

Ramin Zabih works with radiologists to replace intuition with evidence-based diagnosis of vascular disease and breast cancer.

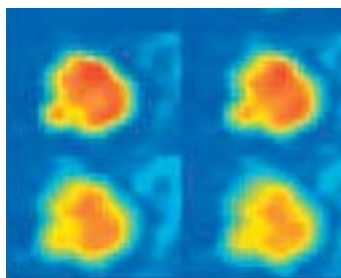


Dan Huttenlocher has made fundamental contributions in object recognition, including Hausdorff-based methods and a Bayesian approach, and in the development of end-to-end systems that apply visual matching and recognition techniques.

Zabih and his colleagues have created an automated computer algorithm for motion correction, and this algorithm has been surprisingly successful. A study found that their computer algorithm performed significantly better than an experienced board-certified radiologist could do with manual motion correction.

They are also investigating new computational techniques to substantially speed up MR imaging. These techniques are based on taking several images at the same time and using algorithms to create a kind of composite image. This approach has the potential to produce a roughly five-fold speed up, allowing a scan that currently takes three minutes to be performed in the time most patients can hold their breath.

In the long term, radiology will need to become much more quantitative, and computers will play a vital role in this transformation. Currently, a report by a radiologist may include phrases like "the gall bladder is somewhat enlarged". This differs from most medical tests, which produce numerical measures (imagine how useless a thermometer would be if it gave a vague opinion rather than a number). Zabih and his colleagues are developing sophisticated computational methods to produce accurate numerical measurements from images. By applying these kinds of new tools from computer science to long-standing problems in medicine, researchers expect to significantly improve patient care in the coming years.



Rate of change: Color enhanced data analysis reveals the diffusion of a contrast agent through a breast tumor and into surrounding tissue.



The Game Design Initiative at Cornell

A multidisciplinary approach to teaching game design is attracting students from all over the Cornell campus—and even across town from Ithaca College—into CIS’s introductory game-design course and an independent-study course.

GDIAC is the brainchild of CS professor David Schwartz and two CS alumni collaborators, Rama Hoetzlein and Mohan Rajagopalan. The interdisciplinary nature of the project has inspired collaborations with faculty across campus, like Todd McGrain and Xiaowen Chen from the art department and David Borden, former director of Cornell’s Digital Music Program.

The courses integrate technical, artistic, and cultural aspects of game design. Topics include almost everything that goes into a computer game—software engineering, game physics, digital art, sound and music, genre analysis, gender issues, game balance, and more. Student teams include programmers, writers, artists, and musicians, all of whom participate in the design and implementation of the game, bringing a variety of skills to bear on the game they are creating. A big contribution of the courses is the experience in working with multidisciplinary teams.

GDIAC started with funding from the GE Fund and Microsoft. The initiative has worked with Electronic Arts and Vicarious Visions on internships and has involved them in the Engineering Cooperative program (which enables students to spend one semester working in industry but still graduate in four years). A community service component, the GDIAC Intern Program, started in the Fall 2003 through participation of the Tompkins County Learning Web, which provides educational and occupational opportunities for youth in need of guidance.

No lab was flexible enough to meet the requirements of the course, so Schwartz worked with the Cornell University Library, the Faculty Advisory Board on Information Technology, and Cornell Information Technology to create a truly innovative lab: the Cornell Library Collaborative Learning Computer Laboratory, or just (CL)³. In a novel design, each of the eight computers in (CL)³ is mounted on a curved table with dual monitors and keyboards/mice. Groups of students collaborate at each station, taking turns typing. Students and instructors can rearrange the lab within minutes to accommodate groups of varying sizes. This shape-shifting lab, inaugurated by Cornell’s Provost in September 2004, offers the ultimate in flexibility. It now supports digital artists throughout the campus.



David Schwartz (top right) at the inauguration of the Cornell Library Collaborative Learning Computer Laboratory (CL)³.



Jon Kleinberg receives the 2001 National Academy of Sciences (NAS) Award for Initiatives in Research. Jon was cited for “his development of deep and innovative algorithms to solve fundamental problems in network, information extraction, and discrete optimization”.

Bart Selman is elected Fellow of the AAAI.

Fred Schneider chairs the International Review of UK Computer Science Research. The review was sponsored by The Engineering and Physical Sciences Research Council, the UK Government’s leading funding agency for research and training in engineering and the physical sciences.

The AFRL/Cornell Information Assurance Institute (IAI) is founded with a \$1M/year grant from AFOSR. See www.cis.cornell.edu/iai/about.htm.

Former undergrads Greg Pass and Frank Wood sell their company, ToFish, to AOL.

Bill Arms publishes *Digital Libraries* (MIT Press).

Intelligent Information Systems Institute (IISI) is established, with Carla Gomes as director.

Rich Caruana, Daisy Fan, Thorsten Joachims, Jai Shanmugasundaram, Jeanna Matthews, Radu Rugină join.

2001

The national organization Engineers for a Sustainable World is started at Cornell (with a different name) under the inspiration of Regina Clewlow (CS 2001). There are now chapters in 21 universities.

Allegra Angus receives the CRA Outstanding Female Undergraduate Award.

Andrew Myers and students Steve Zdancewic, Lantian Zhen, and Nathaniel Nystrom receive the Best Paper Award at SOSIP 2001 for their paper on secure program partitioning.

Kavita Bala, Steve Marschner join.

2002

Tim Roughgarden receives honorable mention in the ACM PhD thesis competition and receives the MPS Tucker Prize. His advisor was Eva Tardos.

PhD student Ioannis Vetsikas and his software “whitebear” wins first place in the Trading Agent Competition. Programs compete by bidding in over 25 simultaneous electronic auctions.

*E*pistemology, the study of knowledge, has a long and honorable tradition in philosophy, starting with the Greek philosophers. But what does it have to do with computer science? “Quite a lot,” says Joe Halpern. “People intuitively use the word ‘know’ all the time when thinking about distributed computing. For example, we are apt to say ‘this process can’t terminate yet because it doesn’t know that the other process has received the message.’”

In the 1980s, Halpern and his collaborators showed how reasoning about knowledge and common knowledge could help in understanding and verifying distributed algorithms. Consider the famous *coordinated attack problem*, introduced by Jim Gray, of Microsoft Research, as an abstraction of a commitment problem in distributed databases. Two generals must attack a village, but they want to attack together; if they attack alone, their armies will be wiped out.

There is a problem: they are miles apart, and the only way they can communicate is by means of a messenger, who might get lost or captured. General A sends a message to General B: “Attack at dawn!” General B gets the message, but will he attack? No, because he is worried that General A doesn’t know that B got the message. So B sends an acknowledgement. General A gets the acknowledgement. Will he attack now? Again, no, because A is worried that B doesn’t know that he (A) knows that B got the acknowledgement. So A acknowledges the acknowledgement. The messenger makes it again (although he’s getting tired!). But this still doesn’t work —General B is worried that A doesn’t know that B knows that A sent the message.

There is never *common knowledge* that the message was sent, where common knowledge is the state in which everyone knows that everyone knows that everyone knows that Indeed, as Halpern and his former student Yoram Moses showed, it is impossible to get common knowledge in a system where communication is not guaranteed. Moreover, as they also showed, common knowledge is a necessary and sufficient condition for coordination. Halpern and Moses were awarded the 1997 Gödel Prize for the paper describing this work, which included a general approach for using epistemic logic to analyze distributed systems.

The work on knowledge has found application in other areas. In AI, robots trying to coordinate have to reason about what one robot knows that the other knows. More recently, Halpern and his students Riccardo Pucella and Kevin O’Neill showed that knowledge is critical in reasoning about security. For example, what does it even mean to say that a message is sent anonymously? Roughly speaking, it should mean that no one knows who the sender is. This has typically been taken to mean that all possible senders are equally likely to have sent it. But it’s a bit more subtle than that. Somehow, probability has to be involved. A contribution of \$5,000,000 to Cornell may be anonymous, but not everyone is equally likely to have made it. Prior beliefs must be taken into account.

Knowledge and belief play a critical role in economics, particularly game theory. Here, the celebrated notion of *Nash equilibrium* is not always applicable because it requires that all agents be aware of all moves. Halpern, his student Leandro Rego, and Larry Blume and David Easley in Economics are exploring models for decision making and game theory that take this lack of awareness into account.

In 1986, Halpern organized a conference on Theoretical Aspects of Reasoning about Knowledge (later renamed Theoretical Aspects of Rationality and Knowledge) in order to bring together computer scientists, economists, and philosophers interested in issues of knowledge. Now, almost 20 years later, the conference is still going strong. The area continues to be active, and everyone *knows* that Cornell is at the forefront of these activities.



CS Professor Joe Halpern and his former student Yoram Moses received the 1997 Gödel Prize for their paper that showed that common knowledge is a necessary and sufficient condition for coordination.

Forty years of numerical analysis and scientific computing

Numerical analysis (NA) intersects applied math and computer science. NA has flourished at Cornell since the 1960s because it has always been a strong component of the faculties in Computer Science and Mathematics and because of easy participation in the Center for Applied Mathematics. For us, it is a best-of-both-worlds history, with advanced algorithmic ideas on the one hand and rigorous mathematical analysis and applications on the other.

Roland Sweet and Jim Bunch launched very productive research careers at Cornell. The Fast Poisson solvers of the early 1970s dramatically widened the class of solvable elliptical PDEs, and Sweet's work on cyclic reduction was a key factor. Bunch established our tradition in numerical linear algebra with fundamental work on symmetric indefinite systems. He also was one of four co-authors of LINPACK, a landmark in the history of numerical software and a precursor to LAPACK.

Faculty members Jorge More and John Dennis established a new way of thinking in the field of nonlinear equation solving and optimization. Their work on quasi-Newton methods showed just how well one could live with approximate Jacobians and Hessians. The sparse optimization research of Tom Coleman since the 1980s represents a continuation of this thread of NA. Incorporating ideas from graph theory and other areas, Coleman produced a steady stream of advanced numerical algorithms that could utilize fully the new generation of parallel machines. Coincident with his tenure as Director of the Cornell Theory Center, Coleman moved into computational finance and established the Financial Industry Solutions Center (FISC) in Manhattan. As an example of outreach to the applications community, FISC is one of the most remarkable realizations of Cornell's Land Grant Mission.

CS professor Steve Vavasis, with an interest in both differential equations and optimization, has brought advanced computer science ideas to several classical application areas. His book, *Nonlinear Optimization: Complexity Issues* (1991), reflects the department penchant for subareas-without-borders, in this case, NA and theory. The use of computational geometry and sophisticated data structures makes Vavasis's automatic mesh generator (QMG) one of the most important software tools for the solution of boundary value problems over irregular domains. The rigorous analysis of the matrix problems that arise in applications has been a hallmark of his work.

Similarly, former colleague Nick Trefethen's segue into the world of pseudo-spectra evolved from profound observations about the non-normal matrices that arose in various partial differential equation settings. This work is very much a tribute to the singular value decomposition (SVD), a matrix factorization that has come to dominate much of the numerical linear algebra scene. Charlie Van Loan's generalized SVD was used to solve several key real-time signal processing applications in the days of Reagan's "Star Wars" program, as were techniques developed by former colleague Frank Luk. Van Loan's SVD method with Gene Golub for total least squares created a new framework for doing least squares fitting when there are errors in the data, and his cache-friendly block Householder representation, developed with former PhD student Chris Bischof, is today the standard way of organizing orthogonal matrix computations like the SVD. According to Google Scholar, Van Loan's book with Golub, *Matrix Computations* (1983), is the most widely-cited text in the computing and mathematical sciences.

In the future, NA will play a critical role in the broader field of computational science and engineering. Typical of the new era of interdisciplinary research is an NSF-sponsored ITR project concerned with the simulation of turbulent combustion. Researchers in computer science, mechanical engineering, and math are combining their talents to address the problems associated with a high-dimensional numerical integration that is at the heart of the computation. The idea is to reduce the effective dimension through a table look-up scheme. To say something about the accuracy of the overall method requires physical modeling of the underlying phenomena (MechE), computational geometry (CS), and convex analysis (Math). The three-way collaboration makes it possible to track each table entry's domain of validity in "reaction space".

The intersection between NA and other departmental research areas is also greater now than ever before. For example, CS professor Uri Keich works on statistical and algorithmic problems that arise in biological sequence analysis. Researching the significance of matches of DNA sequences, Keich identified a problem in the extremely popular BLASTn program, a problem he is now working to fix. And let's not forget that PageRank is an eigenvector computation. The futures of NA and CS are coupled—and that is a good thing!

Uri Keich works on statistical and algorithmic problems that arise in biological sequence analysis.



Researcher Donna Bergmark receives the Best Paper Award for Collection Synthesis in the ACM Joint Conference on Digital Libraries.

Fred Schneider chairs the NSF ITR Program Review.

PhD '92 Daniela Rus and BA '93 Sendhil Mullainathan win MacArthur Genius Award.

Researcher Carl Lagoze, with three others, defines the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH). The work has led to renewed interest in shared metadata and increased ability to locate relevant digital assets regardless of geographical location.

Student Tim Roughgarden wins the "Danny Lewin Best Student Paper Award" at STOC 2002.

Ramin Zabih and student Vladimir Kolmogorov receive the Best Paper Award in the European Conference in Computer Vision. Their paper dealt with minimizing energy functions via graph cuts.

Uri Keich joins. David Gries becomes Assoc. Dean of Engineering.

2003

CS offers an undergrad Information Science major in Arts & Sciences.

Bob Constable is elected to the CRA Board.

Fred Schneider receives an honorary doctorate from the University of Newcastle upon Tyne, U.K.

Jon Kleinberg, Eva Tardos, and student David Kempe receive the Best Research Paper Award in the ACM SIGKDD Intl. Conference on Knowledge Discovery and Data Mining. Their paper, *Maximizing the spread of influence through a social network*, is one of a series of papers on social networks produced at Cornell.

The Cornell Game Design Initiative is formed, under the direction of Dave Schwartz.

Steve Marschner shares a Technical Achievement Award from the Academy of Motion Picture Arts and Science with Henrik Jensen and Pat Hanrahan for their model of subsurface scattering of light in translucent materials. The model has been used often, including for Gollum in *The Lord of the Rings* trilogy.

Lillian Lee's work with postdoc Regina Barzilay on a system that learns to paraphrase is featured in *The New York Times*.

The CS Programming Team wins honorable mention in the world finals in the ACM meeting at the Czech Republic.

Automated reasoning: The impossible made indispensable

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

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.

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.

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



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

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

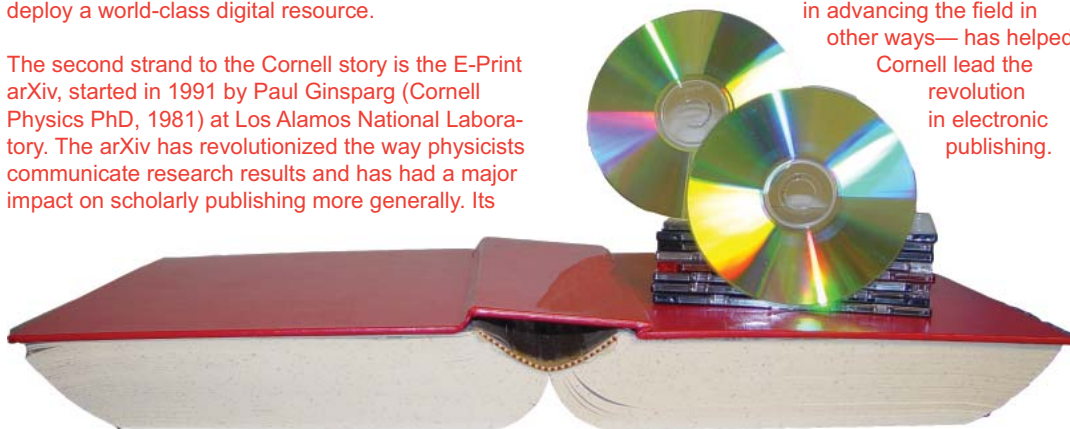
innovations have been emulated by other online literature and database systems, both academic and commercial. Unlike articles submitted to paper journals, articles submitted to arXiv.org are immediately available online, at no cost to the user. Ginsparg received a MacArthur Genius Award for this work.

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 led 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.

Computational complexity

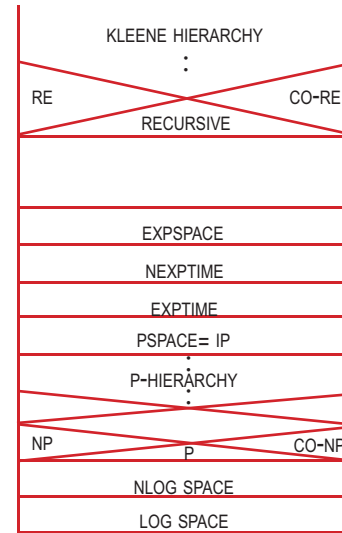
In 1965, the year Juris Hartmanis became Chair of the new CS Department at Cornell, he and his colleague Richard Stearns published the paper *On the computational complexity of algorithms* in the *Transactions of the American Mathematical Society*. The paper introduced a new field and gave it its name. Immediately recognized as a fundamental advance, it attracted the best talent to the field. Theoretical computer science was immediately broadened from automata theory, formal languages, and algorithms to include computational complexity.

As Richard Karp said in his Turing Award lecture, “All of us who read their paper could not fail to realize that we now had a satisfactory formal framework for pursuing the questions that Edmonds had raised earlier in an intuitive fashion — questions about whether, for instance, the traveling salesman problem is solvable in polynomial time.”

Hartmanis and Stearns showed that computational problems have an inherent complexity, which can be quantified in terms of the number of steps needed on a simple model of a computer, the multi-tape Turing machine. In a subsequent paper with Philip Lewis, they proved analogous results for the number of tape cells used. They showed that, given sufficiently more time, Turing machines can always compute more functions. This theorem revealed the existence of a rich hierarchy of complexity classes and provided a framework on which modern complexity theory is built.

Hartmanis and Stearns had the insight to consider both deterministic and nondeterministic models of computation. Their exploration of this relationship laid the groundwork for the celebrated P vs. NP question: whether the class of problems solvable by nondeterministic Turing machines running in polynomial time is strictly larger than the class of problems solvable by their deterministic counterparts. The subsequent work of Stephen Cook, Leonid Levin, and Karp revealed how this question lies at the heart of computationally hard problems throughout computer science and other fields. Nondeterministic Turing machine computation has turned out to be very effective at exposing the subtle interactions between the notions of computing a function and simply verifying its result. Due to its fundamental nature, the P vs. NP question is today widely viewed as one of the most important open questions in all of mathematics.

Studying the structure of complexity classes defined by bounding the time or space allowed for computation has led to a surprisingly rich theory, with striking



This diagram shows how the field believes complexity classes look. It is known that P is different from ExpTime, but there is no proof that $NP \neq P$ and $PSPACE \neq P$.

equivalences and separations among complexity classes, fundamental and hard open problems, and unexpected connections to distant fields of study. An early example of the surprises that lurk in the structure of complexity classes is the Gap Theorem, proved by Hartmanis’s student Allan Borodin and by Boris Trakhtenbrot; essentially, it says that surprisingly large “gaps” appear in the hierarchy. Sometimes, the surprises come from the equivalence of two complexity classes that had been long imagined to be different: in 1988, Hartmanis’s student Neil Immerman and Róbert Szelepcsényi showed that nondeterministic space is closed under complementation, resolving a question that had withstood 25 years of research. They were awarded the 1995 Gödel prize for this work.

Many fundamental questions in complexity theory remain open, despite efforts of researchers in the past 40 years. One of the characteristics of Hartmanis has been his ability to come up with questions that are astoundingly hard to answer. For example, he and his student Ted Baker conjectured that all NP complete sets are isomorphic, under isomorphism computable in polynomial time — a conjecture that is still open. If this conjecture is true, then there is essentially only one NP-complete set, which appears in many guises.

The concepts and methods of complexity theory apply widely and unexpectedly in many parts of math and computer science, from conjectures about the complexity of computable real numbers to reformulations of such basic notions as inductively defined sets. For instance, CS professor Dexter Kozen, a former student of Hartmanis, who introduced alternating machines combining existential and universal quantifiers, recently generalized this award winning work to provide a computational characterization of inductively defined sets that captures the hyperelementary relations over arbitrary structures — the runtime of his machines is measured by ordinals. Even more far afield, Bob Constable and Kurt Mehlhorn defined the computational complexity of higher-order functions and complexity classes

Hartmanis (pictured) and colleague Richard Stearns showed that computational problems have an inherent complexity.



of such functions; these concepts have been useful in studying the runtime of programs in functional languages such as ML and Haskell.

In this short article, we have barely touched the surface of a field that is mathematically deep, rich, and beautiful. Over the years, Hartmanis and his students have been in the thick of the research in this field and, together with other Cornell faculty, have helped it reach out to influence other disciplines. For example, the connections between phase transitions

and computational complexity, discussed below in the “ice cube” highlight, is drawing physicists into the study of computational complexity.

In 1993, Hartmanis and Stearns were awarded the ACM Alan M. Turing Award, the highest prize given in computer science, “In recognition of their seminal paper, which established the foundations for the field of computational complexity theory.” That paper was the start of 40 years of research by some of the brightest and most curious minds in computer science.



Bart Selman uses SAT to study phase transitions.

CS professor Bart Selman, with physics colleagues Remi Monasson in France and Riccardo Zecchina in Italy and Scott Kirkpatrick at the IBM T. J. Watson Research Center, has been in the thick of this research, dealing mainly with the SAT (satisfiability) problem. Consider a boolean formula that is the **and** of a set of clauses, each of which is the **or** of variables or their negation. Here is a formula with two clauses: $(x \text{ or } y) \text{ and } (\text{not } x \text{ or not } y)$. This formula can be satisfied (made true) by making x **true** and y **false**. The SAT problem is to determine whether such a formula is satisfiable.

Consider a collection of formulas, and group them by the ratio r of the number of clauses to the number of variables. For small enough r , most formulas in the group will be satisfiable. This makes sense; each of the clauses in a formula restricts possible satisfying assignments, and the fewer restrictions, the more likelihood of being able to find an assignment. But as r get large, more and more formulas become unsatisfiable. A *phase transition* takes place at the point r where suddenly most formulas become unsatisfiable. For the collection of formulas with three variable clauses (3-SAT), r is about 4.25.

Selman and his colleagues found the following surprising result for mixtures of “random” 2-SAT and 3-SAT. With up to a certain percentage of 3-SAT in the mixture, the formulas can be solved in average polynomial time, and the phase transition is smooth. But with a higher fraction of 3-SAT, search procedures for satisfying assignments scale exponentially at the phase transition, and this transition is abrupt, as when water freezes.

They also discuss the *spin-glass* model as a way of explaining why phase transitions work the way they do—perhaps not only for computational problems but for problems in physics as well. The spin glass, a basic model of a magnetic system, starts with an array of magnetic particles, each oriented either up or down. The orientation of each particle affects the orientation of its neighbors, and a particle can flip from one state to the other with certain probability. Getting the model to settle into a lowest-energy state, in which there is no more flipping, is equivalent to solving a satisfiability problem.

In the past, the flow of information has gone from physics to computing. Computing may now provide insights that deepen the understanding of physics and the physical world.

Computational complexity and the ice cube

Water goes through a phase transition when it freezes. So does a metal when it melts. Some of these transitions are smooth; others exhibit anomalies when the critical point of transition is neared, as in water freezing. In 1982, Cornell physics professor Ken Wilson was awarded the Nobel Prize for his 1971 theory that helped explain how these phase transitions worked.

Now, computer scientists are finding similar phase transitions in computational problems. This connection has stimulated collaboration between statistical physicists, studying disordered systems; mathematicians, studying random combinatorial structures; and computer scientists, interested in complexity and algorithms. The collaboration has led to novel algorithms and to a detailed understanding of phase transition phenomena in computational problems and the underlying combinatorial search spaces.



Johannes Gehrke receives a Cornell University Provost's Award for Distinguished Scholarship.

David Gries publishes *Multimedia Introduction to Programming Using Java* (Springer-Verlag), with his son, Paul.

Once again, PhD student Ioannis Vetsikas and his software “whitebear” wins first place in the Trading Agent Competition. From 2001 to 2005, his worst finish is third.

Bobby Kleinberg joins.

2005

John Hopcroft receives the 2005 IEEE Harry Goode Memorial Award for “fundamental contributions to the study of algorithms and their applications in information processing”.

Technology Research News magazine, in its “Top Picks: Technology Research Advances of 2004”, includes work by two CS groups: Jon Aizen, Dan Huttenlocher, Jon Kleinberg, and Tony Novak devised a way to measure users' reactions to an item description; and Lillian Lee and Regina Barzilay developed software that picks up the topic structure of whole documents to generate more accurate automatic summaries.

Fred Schneider is named chief scientist of TRUST (Team for Research in Ubiquitous Secure Technologies) a new five-university NSF Science and Technology Center.

Student Filip Radlinski receives the Best Student Paper Award at the ACM SIGKDD Conference.

Student Alexandru Niculescu-Mizil receives a Distinguished Student Paper Award at ICML.

Student Thomas Finley receives a Distinguished Student Paper Award at ICML.

Thorsten Joachims receives the Best Paper Award at ICML.

Jon Kleinberg, Jure Leskovec, and Christos Faloutsos receive the Best Research Paper Award at the 11th Conf. on Knowledge Discovery and Data Mining.

Jon Kleinberg and Eva Tardos publish *Algorithm Design*. (Addison-Wesley).

Rafael Vinoly architects begin a feasibility study for a CIS information campus signature building; CS is included as a core unit of CIS.

Computer science chairs

1965–71	Juris Hartmanis
1971–77	Gerry Salton
1977–82	Juris Hartmanis *
1982–87	David Gries *
1987–92	John Hopcroft
1992–93	Juris Hartmanis
1993–99	Bob Constable
1999–	Charlie Van Loan

* 1978-79 & 1983-84 *Dick Conway, Acting Chair during Sabbaticals*



Past chairs of CS: (from left) Juris Hartmanis, David Gries, Bob Constable, John Hopcroft, Dick Conway, and Gerry Salton, on the occasion of Juris Hartmanis's Turing Award celebration.

Computer science faculty

William Arms
Graeme Bailey
Kavita Bala
Ken Birman
Claire Cardie
Rich Caruana
Tom Coleman
Bob Constable
Ron Elber
Daisy Fan
Paul Francis
Johannes Gehrke
Carla Gomes
Don Greenberg

David Gries
Joe Halpern
Juris Hartmanis
John Hopcroft
Dan Huttenlocher
Thorsten Joachims
Klara Kedem
Uri Keich
Jon Kleinberg
Bobby Kleinberg
Dexter Kozen
Lillian Lee
Steve Marschner
Andrew Myers

Keshav Pingali
Radu Rugina
Fred Schneider
David Schwartz
Bart Selman
Jai Shanmugasundaram
David Shmoys
Gün Sirer
Eva Tardos
Tim Teitelbaum
Charles Van Loan
Steve Vavasis
Ramin Zabih

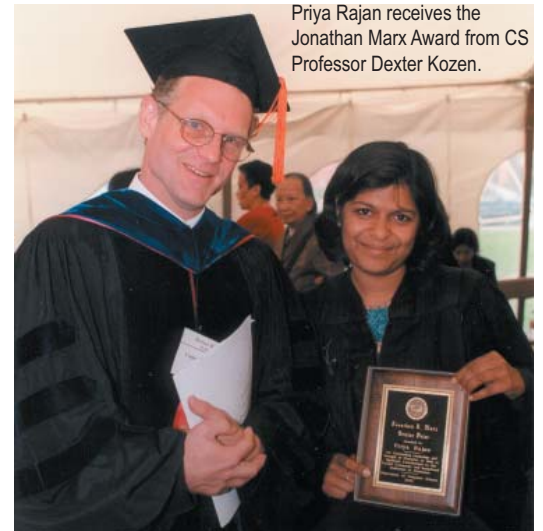
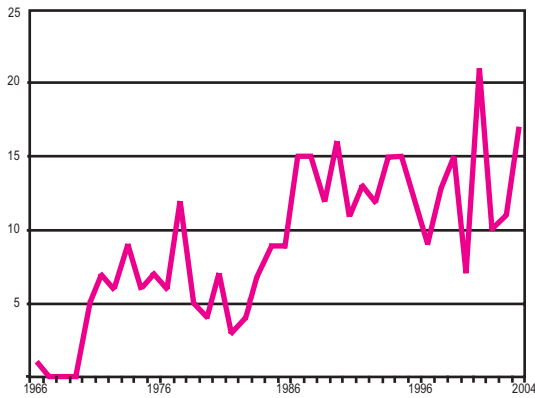
Senior research associates

Paul Chew
Alan Demers
Dean Krafft
Christoph Kreitz
Carl Lagoze
Yuying Li
Paul Stodghill
Robbert van Renesse



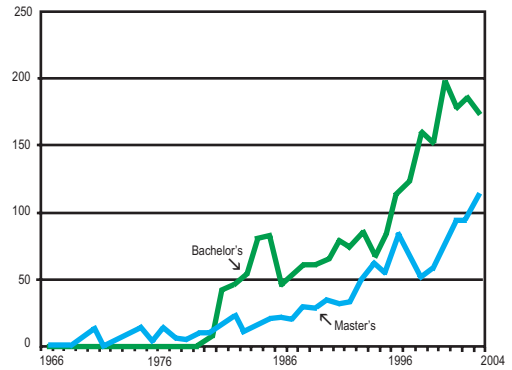
Not pictured: Rich Caruana (see p. 18), Ron Elber (p. 30), Don Greenberg (p. 29), Klara Kedem, Lillian Lee (p. 26), Andrew Myers (p. 15), David Shmoys, Tim Teitelbaum (p. 15), and Steve Vavasis (p. 20).

PhD's Granted



Priya Rajan receives the Jonathan Marx Award from CS Professor Dexter Kozen.

Bachelor's/Master's Granted



PhD's Granted	360
Master's Granted	1403
Bachelor's Granted	2408



ACM Turing Award

1993: Juris Hartmanis
1986: John Hopcroft

National Academy of Engineering

1992: Dick Conway
1991: Don Greenberg
1988: Juris Hartmanis
1988: John Hopcroft

American Academy of Arts & Sciences

2000: Eva Tardos
1992: Juris Hartmanis
1987: John Hopcroft

Honorary doctorates

2003: Fred B. Schneider, University of Newcastle upon Tyne, UK
1999: Don Greenberg, New Jersey Institute of Technology, NJ
1998: David Gries, Miami University, Oxford, OH
1998: Juris Hartmanis, University of Missouri, MO
1996: David Gries, Daniel Webster College, NH
1995: Juris Hartmanis, University of Dortmund, Germany
1990: John Hopcroft, University of Seattle, WA

Grand Medal of the Latvian Academy of Sciences (Lielo Medalu)

The highest award given by the Academy for outstanding creative contribution.

2001: Juris Hartmanis

Latvian Academy of Science

1992: Juris Hartmanis

Bolzano Gold Medal, Academy of Science, Czech Republic

1995: Juris Hartmanis

Humboldt Senior U.S. Scientist Award

1993-94: Juris Hartmanis

Humboldt Distinguished Scientist Award

1987-88: Gerry Salton

ACM Coons Award

The highest award given in graphics.

1987: Don Greenberg

National Computer Graphics Association Academic Award

1988: Don Greenberg

National Academy of Sciences Award for Initiatives in Research

Given to one US scientist under the age of 35.

2001: Jon Kleinberg

Guggenheim Fellowships

2001-2002: Joe Halpern
1999-2000: Eva Tardos
1996-1997: Steve Vavasis
1991-1992: Dexter Kozen
1990-1991: Bob Constable
1983-1984: David Gries
1961-1962: Gerry Salton

Fulbright Scholar

2002: Joe Halpern

SOCIETY FELLOWS

Fellow, IEEE

1987: John Hopcroft

Fellow, ACM

Program began in 1993.

2003: Dexter Kozen
2002: Joe Halpern
2002: Bart Selman
2001: David Shmoys
1998: Ken Birman
1997: Eva Tardos
1994: Bob Constable
1994: Gerry Salton
1994: Fred Schneider
1993: David Gries
1993: John Hopcroft
1993: Juris Hartmanis

Fellow, AAAS

2002: Don Greenberg
2002: Bart Selman
1995: Fred Schneider
1990: David Gries
1987: John Hopcroft
1981: Juris Hartmanis

Fellow, AAAI

2000: Bart Selman
1993: Joe Halpern

NATIONAL/INTERNATIONAL PRIZES FOR PAPERS

Fulkerson Prize

Given triennially by the AMS and the Mathematical Programming Society for a paper published in the previous six years in discrete mathematics.

1988: Eva Tardos

ACM/EATCS Gödel Prize

Given annually for an outstanding paper(s) in the past six years in theoretical computer science.

1997: Joe Halpern

Fox Prize

Given to a person under 31 in the field of numerical analysis.

1993: Yuying Li

ACM Programming Systems and Languages Paper Award

1977: David Gries and Susan Owicki

SERVICE AWARDS

Computing Research Association Distinguished Service Award

2000: Juris Hartmanis
1991: David Gries

NATIONAL EDUCATION AWARDS

New York State Professor of the Year

Given by CASE and the Carnegie Foundation for the Advancement of Teaching. Universities and colleges submit nominations in any discipline.

1994: Dan Huttenlocher

AFIPS (American Federation of Information Processing Societies) Education Award

1986: David Gries

ACM-SIGCSE Education Award

1991: David Gries

ACM Karl V. Karlstrom

Outstanding Educator Award
1995: David Gries

IEEE Taylor L. Booth

Education Award

1994: David Gries

SELECTED CORNELL TEACHING AND ADVISING AWARDS

Cornell Weiss Presidential Fellow

Cornell instituted this award in 1993 to recognize outstanding contributions to undergraduate education. Given annually to three faculty (out of 1600).

1997: Dan Huttenlocher
1995: David Gries

A&S Clark Award for Excellence in Undergraduate Teaching

Given annually to one or two faculty (out of ~ 600).

1989: Juris Hartmanis
1987: David Gries

A&S Russell Distinguished Teaching Award

Given annually to one or two faculty--out of ~ 600.

2001: Dexter Kozen
1998: Keshav Pingali
1994: Dan Huttenlocher

A&S Robert Paul Advising Award

Given annually to one or two faculty (out of ~ 600).

1998: Charles Van Loan

Cornell Carpenter Memorial Advising Award

Recognizes distinguished contributions to undergraduate advising. Given annually to one to four faculty (out of 1600).

2002: Graeme Bailey

AWARDS TO GRADUATE STUDENTS

2002: Tim Roughgarden: Honorable mention in the ACM Doctoral Dissertation Award competition. Advised by Eva Tardos.

1994: T.V. Raman: ACM Doctoral Dissertation Award for his PhD thesis, *Audio System For Technical Readings*. Advised by David Gries.

1983: Tom Reps: ACM Doctoral Dissertation Award for his PhD thesis, *Generating Language-Based Environments*. Advised by Tim Teitelbaum.

AWARDS TO UNDERGRADUATE STUDENTS

2003: Eugene Lee: First place in the national Intel Student Research Contest. Advised by Kavita Bala.

2003: Omar Khan: CRA Outstanding Male Undergraduate Award.

2001: Allegra Angus: CRA Outstanding Female Undergraduate Award.

1998: Pedro Felzenszwalb: CRA Outstanding Male Undergraduate Award Runner-up.

1998: David Liben-Nowell: Honorable Mention, CRA Outstanding Male Undergraduate Award.

1992: Team of Kleinberg, Munoz, and Krosky: Fifth out of 284 in the Putnam mathematics competition.

1992: Zhang: In the top 10 individuals in the Putnam mathematics competition.

The senior faculty in a department are a stabilizing force; they provide experience, a wealth of knowledge, and a perspective that younger faculty cannot have. Wisdom, some might call it. On the other hand, excellent young faculty bring fresh innovation, excitement, a sense of the new. A department without continual rejuvenation through outstanding new faculty will lose its vitality.

CS at Cornell has consistently been able to attract star young faculty members. One sees this in the number of young faculty who have received special

awards to further their careers. Some of these, like Keshav Pingali, Dan Huttenlocher, Jon Kleinberg, Fred Schneider, and Eva Tardos, have become themselves distinguished senior faculty at Cornell. Many others, like Greg Andrews, Tom Henzinger, Greg Morrisett, Ronitt Rubinfeld, and Bob Tarjan, have excelled elsewhere.

Since such research awards and grants started (about 1984), three to seven of our young faculty have received such awards every year, many of them for several years. In 2003-2004, for example, eight young faculty held 11 such awards.

NSF Presidential Young Investigator

John Gilbert (84-88)
 Vijay Vazirani (87-89)
 Bruce Donald (88-92)
 Keshav Pingali (88-95)
 Dan Huttenlocher (90-97)
 David Shmoys (86-92)
 Stephen Vavasis (90-97)
 Eva Tardos (91-96)
 Paul Pederson (92-94)

NSF Faculty Early Career Development

Monica Rauch-Henzinger (94-96)
 Tom Henzinger (94-96)
 Jon Kleinberg (96-02)
 Ronitt Rubinfeld (96-00)
 Praveen Seshadri (96-00)
 Thorsten von Eicken (96-00)
 Bart Selman (97-03)
 Claire Cardie (96-00)
 Johannes Gehrke (01-04)
 Greg Morrisett (01-03)
 Andrew Myers (01-04)
 Golan Yona (01-03)
 Thorsten Joachims (03-05)
 Steve Marschner (03-05)
 Jai Shanmugasundaram (03-05)
 Rich Caruana (04-06)

Presidential Early Career Award for Scientists and Engineers (PECASE)

Greg Morrisett (00-02)

IBM Faculty Development Award

Kevin Karplus (83-86)
 Fred Schneider (83-85)
 Vijay Vazirani (84-87)
 Keshav Pingali (86-88)
 Johannes Gehrke (00-02)
 Jai Shanmugasundaram (03-06)

Sloan Research Fellowship

Eva Tardos (91-93)
 Thorsten von Eicken (97-99)
 Ronitt Rubinfeld (97-00)
 Srinivas Keshav (97-99)
 Jon Kleinberg (97-99)
 Greg Morrisett (97-03)
 Brian Smith (97-98)

Bart Selman (99-04)
 Lillian Lee (01-03)
 Johannes Gehrke (03-07)
 Andrew Myers (01-03)

Lily Foundation Teaching Fellow

Claire Cardie (96-97)

Packard Foundation Fellowship

Eva Tardos (90-95)
 Jon Kleinberg (00-05)

ONR Young Investigator

Ronitt Rubinfeld (92-97)
 Tom Henzinger (94-96)
 Jon Kleinberg (99-01)

“One of the very best things about Cornell is the way the senior faculty work to ensure the success of the junior faculty. They help the new faculty with funding, write unsolicited letters for fellowships and awards, and of course, provide a lot of one-on-one help through mentorship. In my own case, it was clear that a number of senior people worked hard behind the scenes to clear a path so that I could concentrate on the important things: students and research.”

~ Greg Morrisett
 Allen B. Cutting Professor of Computer Science
 Harvard University



Over the years, CS has contributed internationally to education and research through its texts and monographs. The department is proud to have faculty-authored texts that set the standard for the field in algorithms, automata theory and languages, computa-

tional complexity, compiler construction, information retrieval, numerical analysis, programming methodology, theory of scheduling, and more. Two NRC reports, by Hartmanis and by Schneider, have had a national impact. Only the first edition of each book is listed.

2005

Bala, K., P. Dutre (eds.). *Rendering Techniques 2005*. Springer-Verlag

Birman, K. *Reliable Distributed Systems: Technologies, Web Service, and Applications*. Springer-Verlag

Kleinberg, J., E. Tardos. *Algorithm Design*. Addison-Wesley.

2004

Gries, D., P. Gries. *Multimedia Introduction to Programming Using Java*. Springer-Verlag

Van Renesse, R. *Understanding Ukulele Chords*. Mel Bay Pub.

2003

Bala, K., P. Dutre, P. Bekaert. *Advanced Global Illumination*. AK Peters, Ltd.

Halpern, J. *Reasoning About Uncertainty*. MIT Press

2001

Hopcroft, J.E., R. Motwani, and J.D. Ullman, *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley

2000

Arms, W. *Digital Libraries*. MIT Press

Gries, D., P. Gries. *ProgramLive. Data Description*

Kozen, D., D. Harel, J. Tiuryn. *Dynamic Logic*. MIT Press

Van Loan, C. *Introduction to Scientific Computing: A Matrix Approach Using MATLAB*. Prentice Hall

1999

Gries, D., W. de Roever (eds.). *Programming Concepts and Methods, PROCOMET '98*. Chapman and Hall

Ramakrishnan, R., J. Gehrke. *Database Management Systems*, 2nd edition. McGraw-Hill

Schneider, F.B. *Trust in Cyberspace*. National Academy Press

Schwartz, D. *Introduction to Maple*. Prentice Hall

Schwartz, D. *Introduction to UNIX*. Prentice Hall

1997

Birman, K. *Building Secure and Reliable Network Applications*. Prentice Hall

Keshav, S. *An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network*. Addison-Wesley

Kozen, D. *Automata and Computability*. Springer-Verlag

Schneider, F.B. *On Concurrent Programming*. Springer-Verlag

Trefethen, N., D. Bau III. *Numerical Linear Algebra*. SIAM

1996

Van Loan, C. *Introduction to Computational Science and Mathematics*. Jones & Bartlett

1995

Halpern, J., R. Fagin, Y. Moses, M. Vardi. *Reasoning About Knowledge*. MIT Press

1994

Birman, K., R. Van Renesse. *Reliable Distributed Computing with Isis Toolkit*. IEEE Computer Society Press

1993

Gries, D., F.B. Schneider. *A Logical Approach to Discrete Math*. Springer-Verlag

Gries, D., F.B. Schneider. *Instructor's Manual for "A Logical Approach to Discrete Math"*. CS, Cornell.

1992

Donald, B.R., D. Kapur, J.L. Mundy. *Symbolic and Numerical Computation for Artificial Intelligence*. Academic Press

Hartmanis, J.H. *Computing the Future*. National Academy Press

Van Loan, C. *Computational Frameworks for the Fast Fourier Transform*. SIAM

Zippel, R.E. *Computer Algebra and Parallelism*. Kluwer

1991

Kozen, D. *The Design and Analysis of Algorithms*. Springer-Verlag

Vavasis, S. *Nonlinear Optimization: Complexity Issues*. Oxford Science

1990

Coleman, T.F., Y. Li. *Large-Scale Numerical Optimization*. SIAM

Donald, B.R., et al. *Robotics*. American Mathematics Society

Feijen, W.H.J., A.J.M van Gasteren, D. Gries, J. Misra (eds.). *Beauty is our Business*. Springer-Verlag

1989

Salton, G. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley

1988

Coleman, T.F., C. Van Loan. *Handbook for Matrix Computations*. SIAM

Teitelbaum, T., T.W. Reps. *The Synthesizer Generator: A System for Constructing Language-Based Editors*. Springer-Verlag

Teitelbaum, T., T.W. Reps. *The Synthesizer Generator Reference Manual*. Springer-Verlag

1986

Constable, R.L., S.F. Allen, et al. *Implementing Mathematics with the Nuprl Proof Development System*. Prentice Hall

Trefethen, N. *Numerical Conformal Mapping*. Elsevier

1985

E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys (eds.). *The Traveling Salesman Problem*. Wiley

1984

Van Loan, C., G. Golub. *Matrix Computations*. John Hopkins University Press

1983

Hopcroft, J., A. Aho, J.D. Ullman. *Data Structures and Algorithms*. Addison-Wesley

Salton, G., M.J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill

1982

Coleman, T.F. *Large Sparse Numerical Optimization*. Springer-Verlag

Constable, R.L., S.D. Johnson, C.D. Eichenlaub. *An Introduction to the PL/CV2 Programming Logic*. Springer-Verlag

1981

Gries, D. *The Science of Programming*. Springer-Verlag

1979

Conway, R., C. Bass, M. Fay, D. Gries. *An Introduction to Microprocessor Programming*. Winthrop

Conway, R., J. Archer. *Programming for Poets: Using Basic*. Winthrop

Conway, R., J. Archer. *Programming for Poets: Using Pascal*. Winthrop

Conway, R., D. Gries. *An Introduction to Programming: A Structured Approach Using PL/I and PL/C*. Winthrop

Gries, D. (ed.). *Programming Methodology: a Collection of Articles by Members of IFIP WG2.3*. Springer-Verlag

Hopcroft, J.E., J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley

1978

Hartmanis, J. *Feasible Computations and Provable Complexity Properties*. SIAM

Conway, R., J. Archer. *Programming for Poets: A Gentle Introduction using FORTRAN*. Winthrop

Conway, R. *Programming for Poets: A Gentle Introduction using PL/I*. Winthrop

Constable, R.L., M.J. O'Donnell. *A Programming Logic*. Winthrop

Constable, R.L., S. Johnson. *PL/CV2 Program Verifier Reference Manual*. CS, Cornell.

1977

Conway, R., D. Gries, D. Wortman. *Introduction to Structured Programming, Using SP/k*. Winthrop

Conway, R. *A Primer on Disciplined Programming*. Winthrop

1976

Conway, R., D. Gries, E.C. Zimmerman. *Primer on PASCAL*. Winthrop

Conway, R., D. Gries. *Primer on Structured Programming*. Winthrop

1975

Salton, G. *Dynamic Information and Library Processing*. Prentice Hall

1974

Hopcroft, J., A. Aho, J.D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley

1973

Conway, R., D. Gries. *An Introduction to Programming: A Structured Approach Using PL/I and PL/C*. Winthrop

1971

Gries, D. *Compiler Construction for Digital Computers*. Wiley

Salton, G. *The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice Hall

1969

Hopcroft, J.E., J. D. Ullman. *Formal Languages and their Relation to Automata*. Addison-Wesley

1968

Wegner, P. *Programming Languages, Information Structures and Machine Organization*. McGraw-Hill

Salton, G. *Automatic Information Organization and Retrieval*. McGraw-Hill

1967

Wegner, P. *The Structure of Programming Languages*. McGraw-Hill

Conway, R., W.L. Maxwell, L.W. Miller. *Theory of Scheduling*. Addison-Wesley

1966

Hartmanis, J., R.E. Stearns. *Algebraic Structure Theory of Sequential Machines*. Prentice Hall



Our faculty members have been giving invited lectures at conferences every year since CS was formed. It is not just the senior people who are receiving invitations; many young people are being recognized as well.

2005

- Bala. ACM Student Computing Conf., Urbana, IL.
- Cardie. Plenary. Jt. Euro. Conf. Machine Learning/Euro. Conf. Principles and Practice in Knowledge Discovery in Databases, Porto, Portugal.
- _____. Keynote. Conf. Information Extraction. Antwerp, Belgium.
- Francis. Keynote. Web Caching Workshop (WCW). Nice, France.
- Gehrke. Keynote. Intl. Conf. Machine Learning, Bonn, Germany.
- _____. Keynote. SAS Data Mining Tech. Conf., Las Vegas, NV.
- Gries. Keynote. Mid-South College Computing Conf., Oxord, MS.
- Gomes. Plenary. Inaugural Conf. Northwestern Institute on Complex Systems, Evanston, IL.
- _____. AAAS Annual Meeting, Washington, DC.
- Joachims. Conf. German Classification Soc., Magdeburg, Germany.
- Kleinberg. Natl. Academies Conf. Statistics on Networks, Washington, DC.
- Myers. Keynote. European Symp. Prog. Edinburgh, Scotland.
- _____. Keynote. ACM Program Analysis for Software Tools and Engineering, Lisbon, Portugal.
- _____. 3rd Ann. ACM Workshop Formal Methods in Security Eng., Alexandria, VA.
- Pingali. Keynote. 25th Anniv. Salishan Conf. High Speed Computing, Glen Eden Beach, OR.
- Selman. Plenary. Intl. Jt. Conf. Artificial Intelligence (IJCAI), Edinburgh, Scotland.
- _____. Plenary. 8th Biennial Israeli Symp. Foundations AI, Haifa, Israel.
- Shmoys. Plenary. New Horizons in Computing, Recent Trends in Theoretical Computer Science, Kyoto, Japan.
- Tardos. First Spain-Italy-Netherlands Meeting on Game Theory, Maastricht, The Netherlands.
- _____. Found. Computational Math., July 2005: Santander, Spain.
- _____. 12th Intl. Conf. Random structures and Algorithms, Poznan, Poland.
- #### 2004
- Birman. Intl. Conf. Software Engineering (ICSE), Edinburgh, Scotland.
- Gries. Keynote. Consortium CS in Colleges in Northeast.
- _____. Keynote. ACM Mid-Southeast Conf. Gatlinburgh, TN.
- _____. Plenary. Consortium Computing Small Colleges, Schenectady, NY.

- Halpern. IEEE Symp. Multi-Agent Security and Survivability, Phil., NJ.
- _____. Irish Conf. Math. Found. Computer Science and Information Technology, Dublin, Ireland.
- _____. World Congress Game Theory Soc., Marseilles, France.
- Joachims. Keynote. Belgium/Netherlands Conf. Machine Learning, Brussels, Belgium.
- _____. Intl. Colloq. Grammatical Inference, Athens, Greece.
- _____. IEEE Intl. Conf. Data Mining, Brighton, UK.
- Kleinberg. Plenary. SIAM Conf. Discrete Math., Nashville, TN.
- _____. Plenary. Conf. Uncertainty in AI. Banff, Canada.
- _____. Santa Fe Inst. Business Network Topical Meeting, New York, NY.
- _____. AAAS Ann. Meeting, Seattle, WA.
- Kozen. Logic and Computation, Nelson, NZ.
- _____. Latin American Theoretical Informatics, Buenos Aires.
- Lee. Conf. Uncertainty in AI. Banff, Canada.
- _____. IBM Arch. of On-Demand Bus. Yorktown Heights, NY.
- Schneider. Keynote. Search Leaders' Summit. New York, NY.
- _____. IBM Arch. of On Demand Bus. Yorktown Heights, NY.
- Selman. Plenary. Intl. Conf. Principles and Practice of Constraint Programming, Toronto, Canada
- Tardos. American. Math. Soc., Phoenix, AZ.
- _____. Symp. Theory of Computing, Chicago, IL.

2003

- Arms. Keynote. Assoc. Learned & Prof. Soc. Pubs., London.
- Birman. IEEE Intl. Conf. Autonomic Computing (ICAC), Seattle, WA.
- Constable. Keynote. Math Knowledge Management Symp. Edinburgh, Scotland.
- Halpern. Italian Meeting Game Theory and Applications, Urbino, Italy.
- Kleinberg. Natl. Academies Arthur M. Sackler Colloquium, Irvine, CA.
- _____. AAAS Annual Meeting, Denver, CO.
- _____. Center for Nonlinear Studies Conf. Networks, Santa Fe. NM.
- Kozen. Int. Conf. Logic for Programming, Artificial Intelligence and Reasoning (LPAR), Almaty, Kazakhstan.
- Morrisett. UK Memory Management Workshop, Canturbury, England.
- _____. ACM Conf. Progr. Lang. Des. & Impl., San Diego, CA.
- Schneider. Keynote. Intl. Assoc. Science & Tech. for Development. NY.

Below, we list selected invited lectures at conferences. Omitted are invited lectures at universities and companies as well as workshops that consist mainly of invited lectures. Keynote lectures, banquet speeches, and plenary session talks are so marked.

- Tardos. Plenary. Ann. European Symp. on Algs., Budapest.

2002

- Arms. Keynote. DLESE Ann. meeting, Ithaca, NY.
- Constable. Math Knowledge Management Symp., Hamilton, ON, Canada.
- _____. 35 Years of Automath, Edinburgh, Scotland.
- Gries. Banquet. ITiCSE, Aarhus, Denmark.
- Halpern. Conf. on Dimension in Epistemic Logic, Roskilde, Denmark.
- Hartmanis. Descriptive Complexity and Formal Systems, Univ. Western Ontario, Canada.
- Kleinberg. IFIP Congress, Montreal, Canada.
- Kozen. Weighted Automata (WATA'02), Dresden, Germany.
- _____. Mathematics of Program Construction (MPC'02), Dagstuhl, Germany.
- _____. Fixed Points in Computer Science, Copenhagen, Denmark.
- _____. Int. Symp. Formal Techniques in Real-Time and Fault Tolerant Systems, Oldenburg, Germany.
- Lee. Informatics Jamboree, Univ. of Edinburgh. Scotland.
- Morrisett. Keynote. Conf. Program Analysis for Software Tools & Eng. Charleston, SC.
- _____. European Symp. Progr., Grenoble, France.
- _____. Intel Research Professor Forum, Santa Clara, CA.
- Schneider. Keynote. Formal Aspects of Security, British Computer Soc.
- Shmoys. Plenary. Constraint Programming 2002 (CP 2002), Ithaca, NY.
- #### 2001
- Birman. Keynote. ICDCS, Phoenix, AZ.
- _____. First Conf. Network Computing & Applications (NCA '01), Boston, MA.
- Cardie. Plenary. Intl. Conf. Machine Learning, New Haven, CT.
- Gehrke. Symp. Interface of CS & Statistics, Costa Mesa, CA.
- Gries. Banquet. Hartmanis Retirement Symp., Cornell.
- Halpern. Intl. Jt. Conf. AI (IJCAI 2001), Seattle, WA.
- Kleinberg. Plenary. North Amer. Chap. Assoc. for Computational Linguistics, Pittsburgh, PA.
- _____. Plenary. Neural Inf. Proc. Systems, Vancouver, Canada.
- _____. AMS Jt. Mathematics Meetings, San Diego, CA.
- _____. Plenary. The Learning Workshop, Snowbird, UT.

- Kozen. Symp. Theor. Aspects of Comp. Sci. (STACS), Dresden, Germany.
- _____. Logic, Language, Information and Computation (WoLLIC), Brasilia, Brazil.
- Morrisett. New England Progr. Lang. Seminar. Boston, MA.
- Myers. Intl. Static Analysis Symp., Paris, France.
- Schneider. Intl. Static Anal. Symp., Paris, France.
- _____. Intl. Sem. Teaching of Computing Science. Newcastle, England.
- _____. Keynote. IEEE Intl. Conf. Mobile Agents, Atlanta, GA.
- _____. Intel Res. Prof. Forum, Santa Clara, CA.
- _____. Keynote. Symp. Cyber Security & Trustworthy Software, Stevens Inst. of Tech., Hoboken, NJ.
- 2000**
- Arms. Keynote. European Conf. Digital Libraries, Lisbon, Portugal.
- _____. Kyoto Intl. Conf. Digital Libraries, Japan.
- Birman. Keynote. Middleware 2000, NY.
- Constable. Comp. Continuum, San Francisco, CA.
- _____. Computer Science Celebration. Beersheva, Israel.
- Gomes. Plenary. Natl. Conf. Artificial Intelligence (AAAI), Austin, TX.
- Halpern. Games 2000, Bilbao, Spain.
- _____. Conf. Logic & Found. Game & Decision Theory, Torino, Italy.
- _____. Decision Sciences 2000, Kyoto, Japan.
- Kleinberg. Internet Archive Coll. 2000, San Francisco, CA.
- _____. Santa Fe Inst. Meeting on Complex Interactive Networks, Santa Fe, NM.
- _____. Natl. Academies Meet. Interface of Three Areas of Computer Science with the Math Sciences, Washington, DC.
- Kozen. 5th Conf. Relational Methods in Computer Science (ReLMiCS), Quebec, Canada.
- Morrisett. DARPA ISAT Study Group on Mobile Code.
- Selman. Plenary. IEEE Symp. Logic in CS, Santa Barbara, CA.
- Shmoys. Plenary. CO 2000, Greenwich.
- _____. Plenary. CONF 2000, Saarbrücken, Germany.
- 1999**
- Constable. Logic & Computation, Edinburgh, Scotland.
- Gehrke. Institute for Op. Res. & Management Sciences, Philadelphia, PA.
- Halpern. Australasian CS Conf., Auckland, New Zealand.
- _____. AAAS Conf., Los Angeles, CA.
- _____. IEEE Conf. Logic in CS (LICS), Trento, Italy.
- Kleinberg. Foundations Computational Math., Oxford, U.K.
- _____. ACM Symp. Principles Database Systems, Philadelphia, PA.
- Kozen. Int. Congress of Logic, Methodology and Philosophy of Science, Krakow, Poland.
- _____. Math. Found. Comput. Sci., Szklarska Poręba, Poland.
- Morrisett. OpenSIG Conf., Pitts., PA.
- _____. INFOSEC Research Council Study.
- Shmoys. Plenary. Conf. Comp. Learning Theory, Santa Cruz, CA.
- _____. Oberwolfach Workshop Comb. Opt. Germany.
- 1998**
- Constable. CADE 15, Lindau, Germany.
- _____. Calculemus and Types, Eindhoven, Netherlands.
- _____. Implicit Comp. Complexity, Baltimore, MD.
- _____. Reflection (ASL), Stanford, CA.
- Gries. PROCOMET 98, Shelter Island, NY.
- _____. Banquet. Consortium Computing Small Colleges, Fairfield, CT.
- Halpern. Agent's World, Paris, France.
- _____. Intl. Symp. AI and Mathematics, Fort Lauderdale, FL.
- _____. PARCON98 Symp. New Directions in Parallel & Concurrent Computing, NY.
- Kozen. Amer. Math. Soc. Joint Mathematics Meetings, Baltimore, MD.
- Morrisett. Workshop Security and Languages, Palo Alto, CA.
- _____. Intl. Workshop Computer Science Logic, Brno, Czech Republic.
- _____. DARPA Workshop Behavioral Descr. Software Comp., St. Thomas, Virgin Islands.
- Schneider. Predinner. Natl. Res. Council, Wash., DC.
- _____. ACM Conf. Computer & Commun. Security, San Francisco, CA.
- Shmoys. APPROX 98 (ICALP satellite), Aalborg, Denmark.
- Tardos. Symp. Integer Prog. & Comb. Opt.
- 1997**
- Birman. High Performance Distributed Computing (HPDC 6), Portland, OR.
- Constable. TpNet, Murray Hill, NJ.
- _____. ECCAD, Northwestern, Boston, MA.
- _____. ASL, Madison, WI.
- Halpern. Scandinavian Conf. AI (SCAI '97), Helsinki, Finland.
- Morrisett. ACM Intl. Conf. Functional Progr., Amsterdam, Netherlands.
- Schneider. Keynote. Intl. Workshop W DAG '97. Saarbrücken, Germany.
- Smith. Keynote. IDMS '97, Darmstadt, Germany.
- Tardos. Semiplenary. Math. Prog. Symp., Lausanne, Switzerland.
- 1996**
- Coleman. INFORMS, Wash., DC.
- Constable. CADE, New Brunswick, NJ.
- Halpern. Natl. Conf. AI (AAAI-96), Portland, OR.
- _____. Dutch Assoc. Logic, Utrecht, Netherlands.
- _____. SIAM Conf. Computational Differentiation, Santa Fe, NM.
- Kozen. Keynote. 25th Anniv. Celebration, CS Department, Aarhus Univ., Aarhus, Denmark.
- _____. Tools & Algorithms for the Construction and Analysis of Systems (TACAS'96), Passau, Germany.
- Trefethen. Ann. Conf. Canadian Appl. Math. Soc. Winnipeg, Canada.
- 1995**
- Constable. 25 years of Type Theory, Venice, Italy.
- _____. Automath, Eindhoven, The Netherlands.
- Gries. Irish Math. Soc., Limerick, Ireland.
- _____. Banquet. ZUM '95, Limerick, Ireland.
- _____. Intl. Conf., AMAST '95, Montreal, Canada.
- Halpern. Intl. Coll. Cognitive Science, San Sebastian, Spain.
- Kozen. Intl. Conf. Theory & Practice of Software Dev., Aarhus, Denmark.
- Trefethen. Soc. Natural Philosophy, Blacksburg, VA.
- 1994**
- Constable. Bern, Logic Programming. Switzerland.
- _____. Types Conference, Nancy, France.
- Halpern. Intl. Conf. Epistemic Logic and Theory of Games and Decisions, Marseilles, France.
- Hartmanis. Banquet. Intl. Logic Prog. Symp., Cornell, Ithaca, NY.
- Kozen. Conf. Constraints in Computational Logics, Munich, Germany.
- _____. Conf. Found. Software Tech. & Theoretical CS, Madras, India.
- Schneider. Banquet. Intl. Summer School, Marktberdorf, Germany.
- 1993**
- Bloom. CONCUR '93, Hildesheim, Germany.
- Halpern. Kurt Godel Coll., Brno, Czechoslovakia.
- _____. Conf. Found. Software Technology and Theoretical Computer Science, Bombay, India.
- Hartmanis. Conf. Found. Software Tech. & Theoretical CS, Bombay, India.
- Kozen. Plenary. Symp. Assoc. CS Logic, Swansea, Wales.
- Schneider. ACM Symp. Operating Systems Principles. Asheville, NC.
- Subramanian. IJCAI 93, Chambery France.
- 1992**
- Constable. Mid-Atlantic Math. Logic Symp., Phil., PA.
- _____. Metalogical Frameworks, Bloomington, IN.
- _____. Metalogical Frameworks, Gothenberg, Sweden.
- _____. 25th Anniv. of Inria, Paris, France.
- Halpern. Conf. Theoretical Aspects of Reasoning About Knowledge, Monterey, CA.
- Kozen. Symp. on Logical Methods, Ithaca, NY.
- Schneider. School on Formal Techniques in Real-time & Fault-tolerant Systems, Nijmegen, Netherlands.
- 1991**
- Gries. Keynote. ACM CS Conf. San Antonio, TX.
- _____. Keynote. Swiss Inform. Meeting, Lausanne, Switzerland.
- _____. Keynote. CSEE '90 Fourth SEI Conf. Software Eng. Pitts., PA.
- Halpern. Intl. Jt. Conf. Theory and Practice of Software Development, Brighton, England.
- Van Loan. SIAM Meet. Linear Algebra & its Application, Minneapolis, MN.
- 1990**
- Constable. Types Conference, Espirit Sophia-Antipolis, France.
- _____. LF Conference. Edinburgh, Scotland.
- _____. ILPC, San Diego, CA.
- _____. Computer Alg. Intl. Meeting, Montreal, Canada.
- Hartmanis. SWAT 90, Bergen, Norway.
- _____. 75th Anniv. Celebration of MAA, Columbus, OH.
- Kozen. Math. Found. Comput. Sci., Banská-Bystrica, Slovakia.
- 1989**
- Constable. TACS, Iowa City, IA.
- Gries. Banquet. Fingerlakes Workshop, Cornell, Ithaca, NY.
- _____. Plenary. Conf. Software Eng., Pitts., PA.
- Halpern. Bar Ilan Symp. Found. AI, Bar Ilan, Israel
- Toueg. Second Colloques Jacques Cartier, Lyon, France.
- 1988**
- Gries. Keynote. Computer Curricula Conf., Mercer Com. College, NJ.
- Halpern. Conf. Theoretical Aspects of Reasoning About Knowledge, Monterey, CA.
- _____. Concurrency-88, Hamburg, West Germany.
- Marzullo. ANSA Workshop, Cambridge, UK.
- Schneider. Workshop Formal Tech. in Real-time & Fault-tolerant systems. Warwick, UK.
- 1987**
- Constable. Categories & Logic, Boulder, CO.
- Gries. Banquet. Year of Prog. Inst., Univ. Texas, Austin, TX.
- Halpern. Luncheon. Ann. Elect. Materials Symp., Santa Clara, CA.
- _____. Keynote. SEAS/Share Spring Meeting, Montpellier, France.
- _____. Amsterdam Colloquium, Amsterdam, Netherlands.

Hopcroft. Banquet. NSF CER Conf., Amherst, Mass.
Schneider. Intl. Sem. Teaching of Computing Science. Univ. Newcastle upon Tyne, U.K.
Van Loan. Army Conf. Applied Math., West Point, NY.

1986

Gries. Luncheon. Dept. Chairs Prog., CS Conf., Cincinnati, OH.
Halpern. National Computer Conf., Las Vegas, NV.
Hartmanis. Keynote. Workshop "Towards a Science of Parallel Prog.", Glendon Beach, OR.
Schneider. IFIP Congress, Dublin, Ireland.
_____. Finnish CS Soc., Finland.
_____. Adv. Sem. Real-time Local Area Networks. Bandol, France.

1985

Constable. Assoc. Symbolic Logic, Stanford, CA.
Halpern. Symp. Complexity of Approximately Solved Problems, NY.
Hopcroft. Keynote. RPI-SIAM Conf. Albany, NY.
Salton. Keynote. RIAO-85 Meeting, Grenoble, France.
Van Loan. SIAM Conf. Parallel Computation, Norfolk, VA.

1984

Van Loan. Gatlinburg Conf. Num. Lin. Alg. Waterloo, Canada.

1983

Birman. European USENIX Conf., Cologne, West Germany.
Constable. GI Conf., Dortmund, West Germany.
_____. FCT '83, Borgholm.
Gries. Keynote. Australian CS Conf., Sydney, Australia.
_____. ACM CS Conf., Indianapolis, IN.
Hartmanis. IBM Symp. Found. CS, Hakone, Japan.
Hopcroft. Keynote. Conf. Found. Software Tech. & Theoretical CS, Bangalore, India.
Salton. Keynote. Intl. Conf. Computer Applications in Documents & Libraries, Tel Aviv, Israel.
_____. Keynote. SIGIR Conf. Research & Development in Information Retrieval, Wash., DC.
Schneider. COMPCON '83, San Francisco, CA.
_____. First Natl. Congress Informatics & Telecommunications, Buenos Aires, Argentina.

1982

Gries. Keynote. Conf. Found. Software Tech. & Theoretical CS, Bangalore, India.

1981

Hopcroft. GI Conf. Theoretical CS, Karlsruhe, West Germany.
_____. SIAM Natl. Meet., RPI, NY.

1980

Hartmanis. Intl. Symp. Math. Found. CS, Rydzine, Poland.

Luk. Conf. Appl. Num. Anal. & Special Functions in Statistics, MD.

1979

Conway. Conf. VLDB, Rio de Janeiro, Brazil.
Gries. Intl. Conf. Software Eng., Munich, Germany.
Hartmanis. Math. Found. CS Symp., Olomouc, Czechoslovakia.
_____. IEEE FOCS, Puerto Rico.
Kozen. Second Symp. Fund. Comput. Theory, Berlin, Germany.
Schneider. Keynote. Prog. Meth. Workshop, Polytechnic Inst. of New York, Brooklyn, NY.

1978

Constable. Kleene Symposium, Madison, WI.
Gries. GI-8 Jahrestagung, Berlin, Germany.

1977

Han. TMS/ORSA Meeting, San Francisco, CA.
Hopcroft. Keynote. Second IBM Symp. Math. Found. CS, Kansai, Japan.

1976

Constable. American Math Society, New York, NY.
Dennis. State-of-the-Art Conf., York Univ., England.
Gries. Fourth GI Fachtagung ueber Programmiersprachen, Erlangen-Nuernberg, Germany.
Han. IEEE Conf. Decision & Control, Clearwater, FL.

1975

Dennis. Mini-Conf. Matrix Theory & Num. Anal., SUNY-Binghamton, NY.
_____. Op. Res. & Management Sci.

Meeting, Las Vegas, NV.
_____. American Math. Soc.

1974

Constable. Conf. Computing Theory. Montova, Italy.
_____. Semantics Conference, Saarbrucken, Germany.
Gries. Keynote. ACM Regional Conf., Detroit, MI.
Hopcroft. IFIP Congress, Stockholm, Sweden.

1973

Dennis. Intl. Symp. Math. Prog., Stanford, CA.
_____. NSF-CBMS Reg. Conf., Pitts., PA.
Gries. Natl. Sys. Conf., Bangalore, India.
Hartmanis. AMS Symp. Complexity of Real Computations. New York, NY.
More. Intl. Symp. Math. Prog., Stanford, CA.

1972

Bunch. Hawaii Intl. Conf. System Sci., Honolulu, HI.
_____. Symp. Complexity Sequential & Parallel Num. Algs., Pitts., PA.

1971

Dennis. ACM 1971, Chicago, IL.

1970

Dennis. Symp. Nonlinear Functional Analysis & Appl., Madison, WI.
Gries. IBM Sem. Advances in Software Tech., Germany.
Hartmanis. Conf. Board of Math. Sciences, San Antonio, TX.
Salton. Ann. Meet. Deutsche Gesellschaft fuer Dokumentation, Bad Reichenhall, Germany.
Shaw. Intl. Conf. Frontiers of Pattern Recog., Honolulu, HI.

1969

Salton. Third Cranfield Conf. in Automatic Documentation, Cranfield, England.
_____. Database Symp., New York, NY.

1968

Salton. IFIP Congress, Scotland, U.K.
_____. Spring Jt. Computer Conf., Atlantic City, NJ.

1967

Salton. Ann. Meeting Amer. Psych. Assoc., Wash., DC.
_____. NY State Assoc. Educational Data Systems, Syracuse, NY.
_____. Intl. Conf. on Content Anal., Phil., PA.

1966

Salton. Princeton Eng. Summer Conf., Princeton, NJ.
_____. Fourth Ann. SUNY Computer Conf., Binghamton, NY.
_____. Swiss Assoc. for Operations Research, Zurich, Switzerland.
_____. Assoc. of Special Libraries & Information Bureaux, London, England.

1965

Hartmanis. Jt. Symp. Logic, Computability, & Automata, Oriskany, NY.
_____. Systems & CS Conf., Univ. Western Ontario, Canada.
_____. Am. Math Soc. Symp., Mathematical Aspects of Computer Science, NY.
Salton. Conf. on Education Inf. Sci., Airlie House, VA.
_____. Intl. Fed. for Documentation, Intl. Congress, Wash., DC.
_____. AAAS Natl. Meeting, Berkeley, CA.

The 40th anniversary committee

Editor-in-chief

David Gries

Editorial board

Johannes Gehrke, Juris Hartmanis, Keshav Pingali, Fred B. Schneider, Eva Tardos

Editorial assistant and Project manager

Nora Balfour

Writers

This booklet is the collective effort of the faculty of the Department of Computer Science.

Designer and Producer

Stephanie Specchio

Copyeditors

Laurie J. Buck, Linda Callahan, Stephanie Specchio

Web site

Orlando Johnson, Una Money Penny

Symposium logistics

Nora Balfour, Kathy Carpenter, Amy Fish, Bill Hogan, Beth Howard, Orlando Johnson, Bonnie Maine