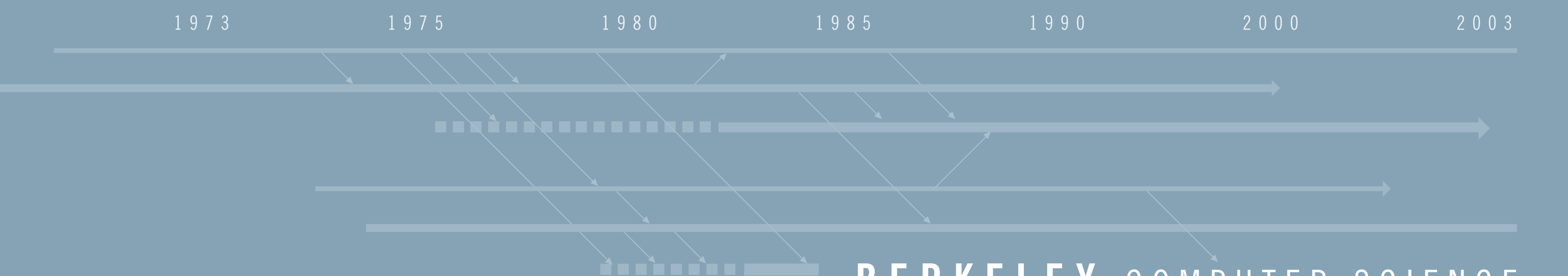
30

THIRTY YEARS OF INNOVATION



BERKELEY COMPUTER SCIENCE

30 YEARS OF INNOVATION

1973–2003

KELEY COMPUTER SCIENC

CONTENTS

THE COMPUTER SCIENCE DIVISION OF THE DEPARTMENT OF EECS AT UC BERKELEY WAS CREATED IN 1973. *THIRTY YEARS OF INNOVATION* IS A CELEBRATION OF THE ACHIEVEMENTS OF THE FACULTY, STAFF, STUDENTS AND ALUMNI DURING A PERIOD OF TRULY BREATHTAKING ADVANCES IN COMPUTER SCIENCE AND ENGINEERING.

THE FIRST CHAIR OF COMPUTER SCIENCE AT BERKELEY was Richard Karp, who had just shown that the hardness of well-known algorithmic problems, such as finding the minimum cost tour for a travelling salesperson, could be related to NP-completeness—a concept proposed earlier by former Berkeley mathematics professor Steven Cook. The resulting P vs. NP question has since been accepted as one of the ten most important open problems in mathematics, along with such classics as the Riemann Hypothesis. Berkeley computer scientists continue to lead the field of computational complexity, with work such as that on probabilistically checkable proofs and the hardness of approximation problems by Sanjeev Arora and Madhu Sudan in the early 1990s, and on quantum complexity theory by Ethan Bernstein and Umesh Vazirani a few years later. Two Turing Awards (Richard Karp, Manuel Blum) and four ACM Ph.D. Dissertation Awards (Eric Bach, Noam Nisan, Madhu Sudan, and Sanjeev Arora) are just a few of the honors garnered by the research in theoretical computer science at Berkeley.

The impact of Berkeley research on the practical end of computer science has been no less significant. The development of Reduced Instruction Set computers by David Patterson and Carlo Sequin, the Redundant Array of Inexpensive Disks project led by Randy Katz and David Patterson, and the INGRES relational database system led by Mike Stonebraker, Larry Rowe and Eugene Wong, can be directly connected to multi-billion dollar industries. In the area of system software, the impact of Berkeley Unix on minicomputers and subsequently on workstations and, through LINUX, on personal computers, is self-evident. Nor can we forget the role of Berkeley alumni in sparking the workstation and personal computer industry—pioneers such as Butler Lampson (Xerox PARC), Bill Joy (Sun), and Steve Wozniak (Apple). Numerical computations would not have been reliable had it not been for adoption of the IEEE 754 floating point standard, largely due to William Kahan, who received a Turing Award in 1989 for this work. In the area of programming languages and software engineering, Berkeley research has been noted for its flair for combining theory and practice, as exemplified in these pages by George Necula's research on proof-carrying code.

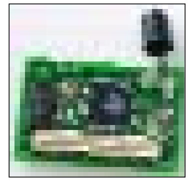
While Berkeley has always led research in theory and computer systems, it was not a central player in the development of symbolic artificial intelligence, or "Good Old-Fashioned AI," in the 1960s and 70s. Berkeley's AI effort grew largely in the 80s and 90s, at a time when problems with this paradigm were becoming evident, and researchers at Berkeley played a major role in developing the new, more probabilistic and learning-oriented AI. This new synthesis brought traditional AI together with control theory, pattern recognition, neural networks, and statistical learning theory. Stuart Russell and Peter Norvig's best-selling textbook has become the canonical exemplar of this synthesis, and research at Berkeley in fields such as vision, robotics and learning is bringing us ever closer to the dream of truly intelligent machines.

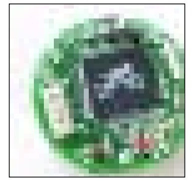
Berkeley's was the one of the first top computer science programs to invest seriously in computer graphics, and our illustrious alumni in that area have done us proud. We were not so prescient with respect to the field of human computer interaction, even though a Berkeley alumnus, Douglas Engelbart, with his 1967 demonstration, did more than any one else to lay out a futuristic vision for the field. Over the last few years, we have made up lost ground by assembling a vibrant, interdisciplinary research group in this area. Finally, computational biology, truly the newest kid on the computer science block, will grow rapidly at Berkeley in the coming years under the visionary leadership of Gene Myers and Richard Karp.

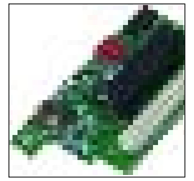
This publication is the result of an outstanding team effort, ably led by Randy Katz. I am proud to share it with you.

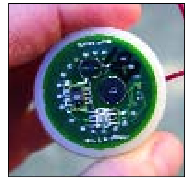
Jitendra Malik
Chair, Computer Science Division, UC Berkeley

CITRIS AND MOTES









*^ **Top:** Prototype motes developed at Berkeley are hardly larger than a small coin.*

^ In a collaborative project involving faculty and graduate students from Computer Science, Integrative Biology, and the Intel Lab, a technician installs microweather stations in a study redwood tree in the UC Berkeley Botanic Garden.

In the spring of 2000, Berkeley EECS faculty tramped across campus to offer researchers a glimpse of a tantalizing future: Imagine that you had at your fingertips hundreds of tiny, cheap 'Smart Dust' sensors to scatter anywhere you chose. Once in place, the sensors would automatically assemble into a network to measure physical quantities, process data, and relay collected data to the outside world. What could you do with this new computational tool?

The responses were as imaginative as they were diverse. Civil engineers envisioned buildings that reported structural damage after an earthquake, and smart offices that switched off light and heat in empty rooms. Other researchers proposed pulse-monitoring wristwatches to alert emergency vehicles in case of heart attack, pollution monitors near roads to assess the environmental impact of traffic, and soil-moisture sensors in vineyards to track optimal growing conditions.

The question was no trivial exercise. Sensor nets were being proposed as the centerpiece of a multi-campus research project, the Center for Information Technology Research in the Interest of Society (CITRIS), which Berkeley was creating in response to a challenge from then-California Governor Gray Davis to apply the University's scientific prowess to the state's environmental, health and energy problems. Today CITRIS, cosponsored by the state and 11 corporate partners, boasts more than 100 affiliated academic researchers, who are already taking smart dust out of the lab into the real world.

The creation of CITRIS marks a sea change in the questions Berkeley computer scientists are tackling, says James Demmel, CITRIS scientific director and a Berkeley computer science and mathematics professor. "For a long time, computer science goals have been set by the internal standards of Silicon Valley," Demmel says. "The questions were how to make a computer widget faster, cheaper, more reliable. Now we're asking questions like how to make the electricity market function better in the state of California. That's how our goals have expanded."

In choosing to focus on sensor networks, CITRIS organizers had picked a technology solidly under development. Berkeley professors Kristofer Pister and David Culler were already building prototype networked arrays of small wireless sensors. Dividing his time between the Berkeley computer science division and the collaborative Berkeley-Intel Research Laboratory in Berkeley, Culler now works on the software for the network sensors, called motes, in a lab strewn with motes of various shapes and sizes and context-aware toy cars that trundle along beneath webs of suspended motes.

The motes range in size from the oatmeal-sized SPEC, just 2.5 millimeters on a side, to motes the size of their power source, two AA batteries. Each mote comes with memory, wireless communication, and power managing capabilities. Customized sensor circuitry snaps on top, allowing the motes to measure a range of environmental conditions, such as temperature, pressure, motion and light. Miniature programs—whose size is measured in tens of kilobytes—run on an operating system called TinyOS, developed by Berkeley and Intel to manage data processing and the self-organization of the sensor networks.

Culler's team has already deployed motes in the field. In 1999, the researchers used a remotely controlled aircraft to scatter motes along a desert highway, where they recorded the speed and direction of passing vehicles. And last year, Culler's group and biologist John Anderson, of the College of the Atlantic in Bar Harbor, Maine, placed motes among petrels on Great Duck Island off the coast of Maine. Studying petrels is challenging, as the presence of people disturbs the nesting birds. By planting motes equipped with light, humidity and temperature sensors into petrel burrows, the researchers were able to peek into the petrels' lifestyle over the course of six months. "This kind of sensor network will have a profound effect on how we do field ecology," Anderson told CITRIS.

Other mote projects are taking off in a number of Berkeley departments. Computer scientists, mechanical engineers and architects are creating mote-centered hardware to relay energy prices to California homes in real time, in the hopes of developing a more workable energy market. Geoengineer Steve Glaser has used motes to study the impact of earthquakes and ground liquefaction on buildings. Culler is involved in an effort to measure the structural stability of the Golden Gate Bridge on windy days and in potential earthquake scenarios, and he is working with botanist Todd Dawson to use motes to explore the ecosystem sustained in the branches of a single redwood tree. Mechanical engineer Paul Wright is using motes to engineer "smart stairways" that could guide firefighters through burning buildings.

Beyond Berkeley, Culler estimates that about 150 research groups worldwide have adopted mote technology.

Although the technology is up and running, many questions remain about how to make smart dust even more useful and versatile. Culler's team is working on more efficient ways to transmit software through the sensor network. Meanwhile, Professor David Wagner and his students are investigating how to block intruders from hijacking motes, adding false motes to a network, or reading sensitive data being beamed from mote to mote.

Despite the challenges, Culler is optimistic that motes will fulfill their promise. "We've made enough progress that we can get serious about deploying real applications," he says. As more and more research projects start using these embedded networks, Culler believes the full implications of this new computing paradigm will begin to sink in. "I think in the next year or so, we will start to get scientific studies that can resolve data they've never been able to perceive before," he says. "As the technology starts being applied, I think we're going to see the revolution happen."

^ Motes support new kinds of socially relevant information processing-intensive research. They have enabled the close monitoring of weather conditions on Great Duck Island, a bird habitat off the coast of Maine, where biologists are trying to understand the nesting behavior of the difficult to observe Kestrel.

TIMELINE

PRE-1973

BERKELEY PH.D. STUDENT *(1954) and faculty member **Al Hoagland** developed the enabling technologies for the first magnetic disk, later commercialized as the RAMAC by himself and others at IBM in San Jose in the mid-1950s.*

IN 1971, SUSAN GRAHAM *joins the Berkeley faculty in Computer Science. When the Computer Science Division of EECS is formed two years later, she becomes the first woman faculty member in the College of Engineering*

IN 1968, *during a time of great campus unrest, three future Turing Award winners join the Berkeley faculty: Manuel Blum, William Kahan, and Richard Karp.*

IN THE LATE 1960S-EARLY 1970S, *many leaders of the field of Computing Systems, such as Butler Lampson (Turing Award), Jim Gray (Turing Award), and Bruce Lindsay (IBM Fellow), developed the Berkeley Time Sharing System, implemented on the SDS 940 series of computers, later acquired by Xerox Corporation.*

THE FUNDAMENTAL PAPERS OF COMPLEXITY AND REDUCIBILITY *were written by Berkeley faculty: S. A. Cook, "The complexity of theorem proving procedures," Proceedings of the 3rd Annual ACM Symposium on Theory of Computing (1971), pp. 151–158; R .M. Karp, "Reducibility among combinatorial problems," Complexity of Computer Computations, Plenum, New York (1972), 85-103; M. Held, R.M. Karp, "The traveling-salesman problem and minimum spanning trees." Operations Research, 18:1138–1162, 1970.*

LOTFI ZADEH *develops the field of fuzzy sets with his paper in Information and Control, 8:338-353, 1965. This lays the foundation for a worldwide fuzzy control systems industry.*

1973

THE COMPUTER SCIENCE DIVISION *of the Electrical Engineering and Computer Science Department within the College of Engineering is formed from an independent CS faculty in Letters and Science and the existing computer science and engineering activity in EECS. The size of the Division stands at 19.90 (fractional FTE due to joint appointments with other departments)*

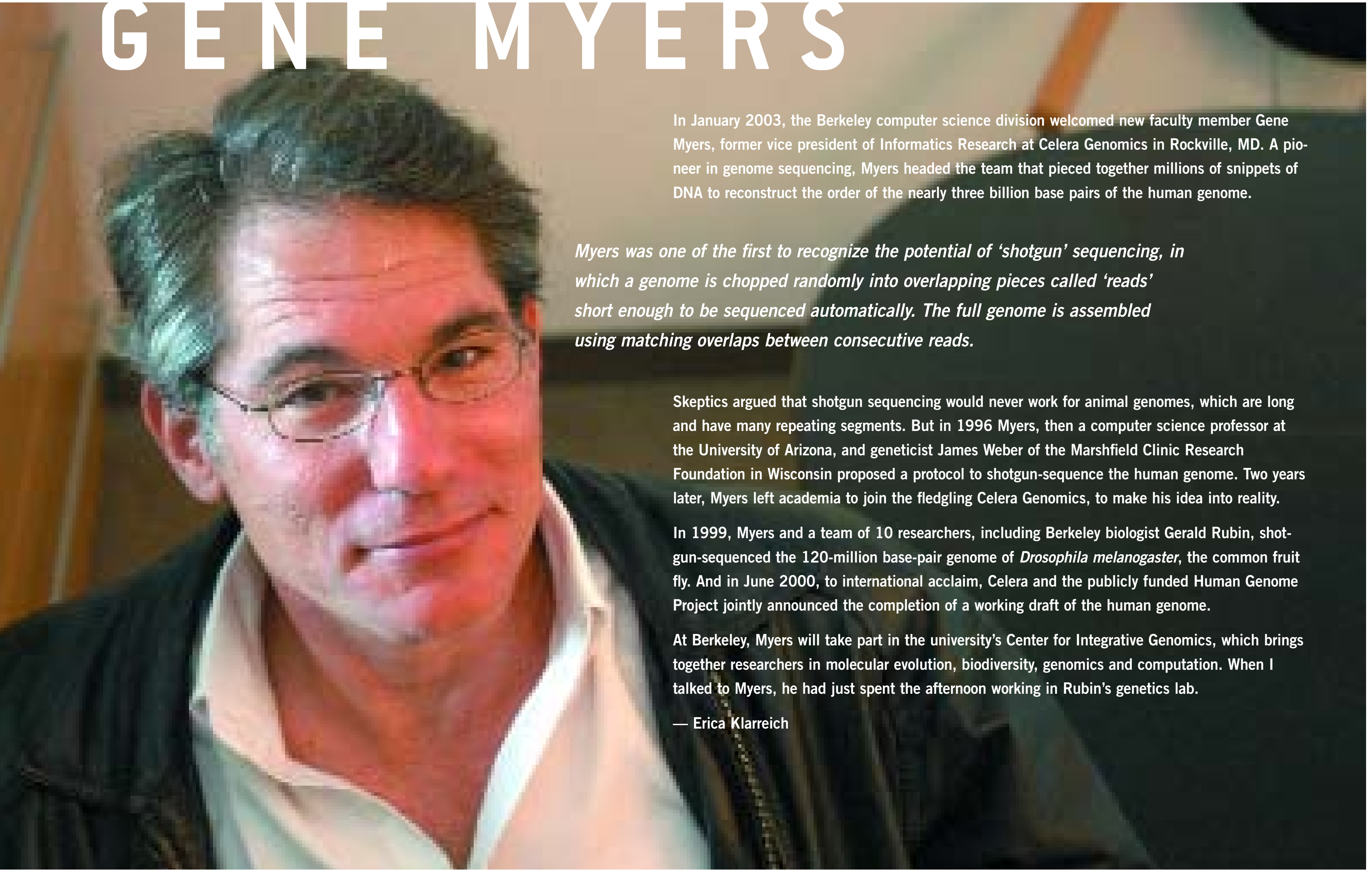
1974

UNDER THE LEADERSHIP OF PROFESSOR ROBERT FABRY, *with help of former Berkeley student **Ken Thompson** (Turing Award), Berkeley leads the university research community with one of the first deployments of UNIX outside Bell Labs.*

1975 1976

INGRES DATABASE SYSTEM PROJECT *(1973-1979), led by Mike Stonebraker and Eugene Wong, is first implemented on UNIX. It becomes the first widely distributed software package developed by Berkeley CS faculty. An influential National Academy study states: "Although the relational model was originally proposed and developed at IBM, it was a government-funded effort at UC-Berkeley that disseminated the idea widely and gave it the intellectual legitimacy required for broad acceptance and commercialization."*

GENE MYERS



In January 2003, the Berkeley computer science division welcomed new faculty member Gene Myers, former vice president of Informatics Research at Celera Genomics in Rockville, MD. A pioneer in genome sequencing, Myers headed the team that pieced together millions of snippets of DNA to reconstruct the order of the nearly three billion base pairs of the human genome.

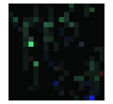
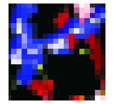
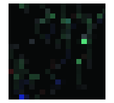
Myers was one of the first to recognize the potential of 'shotgun' sequencing, in which a genome is chopped randomly into overlapping pieces called 'reads' short enough to be sequenced automatically. The full genome is assembled using matching overlaps between consecutive reads.

Skeptics argued that shotgun sequencing would never work for animal genomes, which are long and have many repeating segments. But in 1996 Myers, then a computer science professor at the University of Arizona, and geneticist James Weber of the Marshfield Clinic Research Foundation in Wisconsin proposed a protocol to shotgun-sequence the human genome. Two years later, Myers left academia to join the fledgling Celera Genomics, to make his idea into reality.

In 1999, Myers and a team of 10 researchers, including Berkeley biologist Gerald Rubin, shotgun-sequenced the 120-million base-pair genome of *Drosophila melanogaster*, the common fruit fly. And in June 2000, to international acclaim, Celera and the publicly funded Human Genome Project jointly announced the completion of a working draft of the human genome.

At Berkeley, Myers will take part in the university's Center for Integrative Genomics, which brings together researchers in molecular evolution, biodiversity, genomics and computation. When I talked to Myers, he had just spent the afternoon working in Rubin's genetics lab.

— Erica Klarreich

Q&A

What made you decide to leave Celera and return to academia?

I want to learn and incubate new ideas. When I went to Celera, it was to execute a particular vision, to sequence the genome, and now we've done it. My feeling was that coming back to a university was the right thing to do, because it's a place where I'll have time to think deeply about things and pursue exactly the ideas I want to. In the commercial world, when you've got the wind at your back and the cash behind you, you can move mountains. And we moved a mountain at Celera. But companies also have to make money. They have to produce reagents, or diagnostics, or drugs, and at the moment I'm not particularly interested in designing drugs. I want to understand how it all works. So I'm going back to being a basic scientist. I want the freedom to speculate. Take today, for instance—I got to spend three hours just hanging out, learning how to program the PCR thermocycler, how to pipette, how to turn on the centrifuge. In industry I wouldn't get the chance to do that.

You described your afternoon's activities as "biology bootcamp." What are you trying to accomplish in the lab?

I'm trying to get a feel for the Zen of experimental work. I've gotten to the point where I understand what the experiments are about intellectually, but now I want to actually feel them and touch them. You see, the thing that's really important in experimental work is to understand the nature of error and what kinds of things could go wrong. I want to think about designing experimental protocols, and to do that effectively I have to understand the details much better than I do today. I'm not under the illusion that I'm ever going to be a great experimentalist—at this stage I'm not going to develop 'great hands', as they say. But I do want to set up a lab of my own when I grow up, and design my own experiments and execute them. I want to see how far I can get with being a biologist and still being a computer scientist at the same time.

What do you feel that computer science can contribute to genetics?

I don't think computation is going to drive the discovery of how cells work, but it will be extremely important for interpreting experimental results. Often you do an experiment and you don't get exactly what you need. Take X-ray crystallography, for instance: if you send an X-ray through a crystal, you get a diffraction pattern, and then it's a computer that actually figures out what shape would produce that pattern. Or take sequence assembly: you shred a genome into a gazillion pieces, sequence about 40 million snippets, and then you have to figure out what was the genome those snippets were sampled from.

Computation is also extremely important for modeling and holding information. It's clear that computation is essential to biology simply because of the scale of the system we're trying to think about. We're talking about a system with 200,000 different agents floating around in a cell, and that's just types of agents. We're talking about a system with millions of particles. It's too big for a human mind, so we'll have to use computers to model this stuff and help us think about it.

Why have you chosen Berkeley?

There are lots of reasons. Berkeley has a great computer science department, one of the best in the country, so that's kind of a no-brainer. Biology is also extremely strong here, and there are very good *Drosophila* people here. Then there's the whole group at Lawrence Berkeley Laboratory, great people who can do systems-type engineering projects. And there's the Joint Genome Institute in Walnut Creek, that's affiliated with us in a loose kind of way. Berkeley is a very good place to be positioned in terms of the kinds of activities going on around me. I feel that I'm in the right place.

TIMELINE

1977

MANUEL BLUM *wins the CS Division's first Distinguished Teaching Award, conferred by the UC Berkeley Academic Senate.*

FIRST AND SECOND BERKELEY SOFTWARE DISTRIBUTIONS *(1BSD/2BSD) OF UNIX, incorporating the seminal pieces of software developed by graduate student **Bill Joy**: Pascal and the VI visual editor.*

1978

WOMEN IN COMPUTER SCIENCE AND ENGINEERING *(WICSE) founded by CS grad students, including **Susan Eggers**, **Deborah Estrin**, **Paula Hawthorn**, **Diane McEntyre**, **Mary Anne Niemat** and **Barbara Simons**. Among innovations WICSE spearheads is the CS Re-entry Program.*

DECVAX 11/780 ARRIVES ON CAMPUS, *one of the first to be delivered to a Computer Science Department.*

1979

RICHARD KARP *wins Fulkerson Prize.*

FORMER BERKELEY UNDERGRADUATE STEVE WOZNIAK *wins the ACM Grace Murray Hopper Award for his invention of the Apple Personal Computer.*

UNDER THE LEADERSHIP OF DOMENICO FERRARI AND ROBERT FABRY, *3BSD UNIX adds virtual memory support for the Digital Equipment VAX minicomputer. This hardware/software becomes the standard research platform for DARPA research community.*

EARLY IMPLEMENTATIONS OF ORACLE'S DB SYSTEM *were based on the distributed code of INGRES. INGRES Alumnus Bob Epstein founds Britton-Lee to build a commercial database machine. Epstein later goes on to found Sybase, a leading client-server database supplier in the late 1980s and early 1990s.*

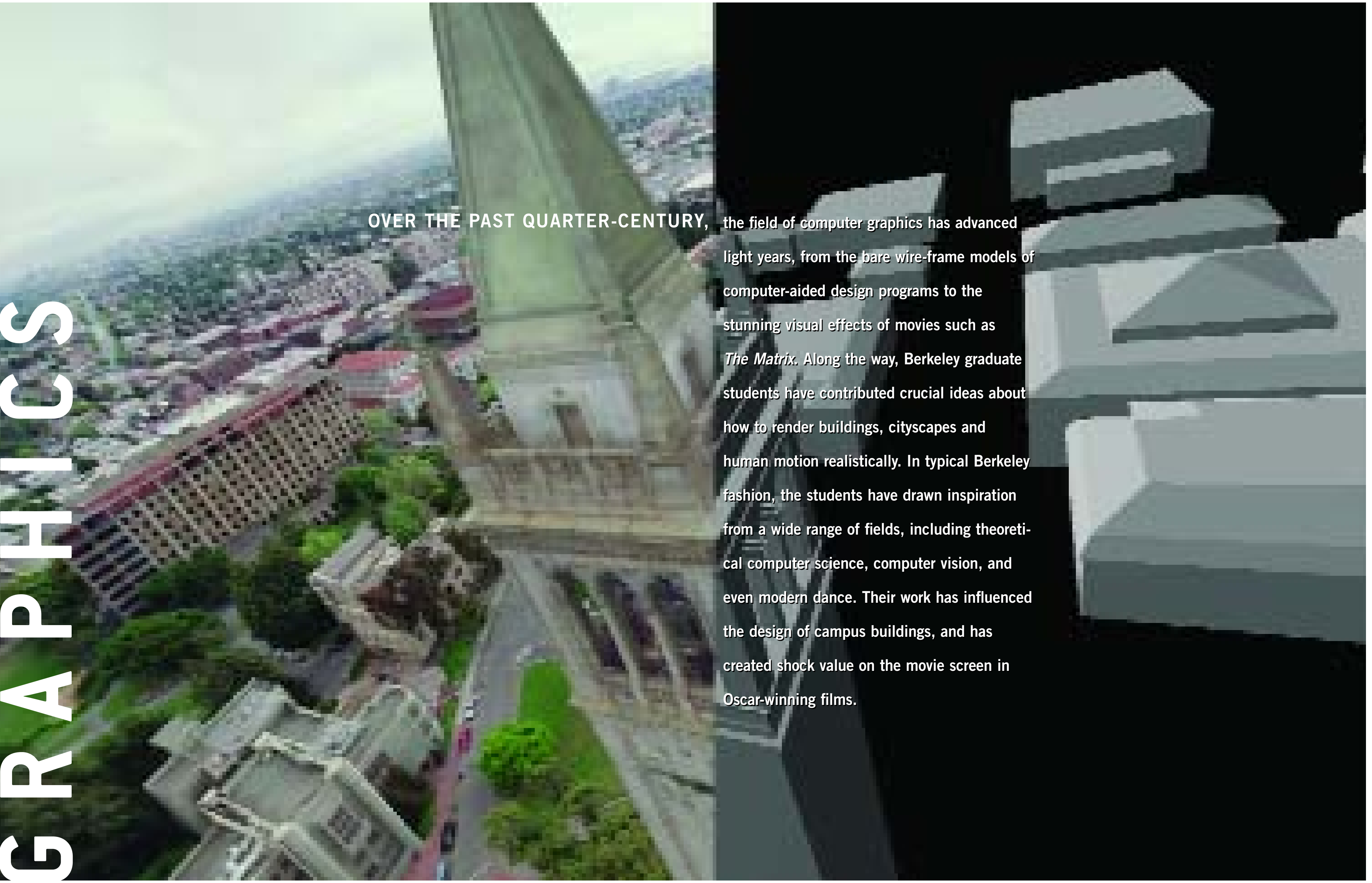
RICHARD FATEMAN *leads the Berkeley development of the VAX version of the Macsyma Computer Algebra System.*

DURING THE DECADE OF THE 1970s, *ten new faculty members join the CS Division.*

1980

MIKE STONEBRAKER, LARRY ROWE, AND EUGENE WONG *found Ingres to commercialize relational database technology. Eventually acquired by Computer Associates.*

GRAPHICS



OVER THE PAST QUARTER-CENTURY, the field of computer graphics has advanced light years, from the bare wire-frame models of computer-aided design programs to the stunning visual effects of movies such as *The Matrix*. Along the way, Berkeley graduate students have contributed crucial ideas about how to render buildings, cityscapes and human motion realistically. In typical Berkeley fashion, the students have drawn inspiration from a wide range of fields, including theoretical computer science, computer vision, and even modern dance. Their work has influenced the design of campus buildings, and has created shock value on the movie screen in Oscar-winning films.

THE CAMPANILE MOVIE

On April 20, 1997, a team of graduate and undergraduate students ushered Professor Jitendra Malik into the lab to be the first to see the fruit of months of round-the-clock labor: a 'fly-through' movie of the Berkeley campus that swooped dizzyingly from ground to air. It morphed seamlessly from computer-generated images of the famous Campanile tower to film footage of a small architectural model of the tower resting in the palm of graduate student Paul Debevec, who stood atop the real Campanile. At the end of the three-minute movie, Malik wore a broad grin. "He asked us, 'Which parts are real and which are synthetic?'" Debevec recalls.

The team had accomplished a novel feat in computer graphics: they had modeled the campus to a dazzling degree of precision using image-based rendering, which pieces together a three-dimensional model from two-dimensional photographs of a scene. At the time, most architectural graphics used preconstructed polygonal models, and the few efforts in image-based rendering were mostly unconvincing. "In so many of the architectural graphics, the geometry was rough, the lighting was strange, and the buildings looked like shoeboxes with photos glued on," says Debevec, now a professor at the University of Southern California in Los Angeles. "We wanted to do something that would have people fooled for a moment."

Debevec teamed up with Camillo Taylor, a University of Pennsylvania professor who was a Berkeley postdoc at the time, to write a computer program called Façade that could reconstruct a three-dimensional model from photographs of an architectural scene. Automatically constructing accurate three-dimensional models of scenes from photographs is still well beyond the reach of computer vision, so instead of trying to fully automate Façade, the pair programmed it to start with input from a human who matches up the edges of the buildings in different photos.

The next step was to use the photographs to add texture to the bare three-dimensional model. The Façade program determined the vantage point from which each photo had been taken, and Ph.D. student Yizhou Yu, now a professor at the University of Illinois at Urbana-Champaign, wrote a program to calculate which parts of the buildings were occluded by others in the photos; put together, these algorithms figured out exactly which parts of the buildings were visible at any moment, and which pixels from the photographs should be used to texture them. Masters student George Borshukov implemented efficient algorithms to add the texture in real time. Finally Debevec, Yu, Borshukov and a team of undergraduates put together the Campanile movie. A few months later, the film premiered to great acclaim at the prestigious Electronic Theater at the annual meeting of SIGGRAPH, the main association for computer graphics professionals. "It opened the eyes of a lot of people in the industry," Borshukov says.

One audience member at the SIGGRAPH showing was John Gaeta, the visual effects supervisor for the film *The Matrix*, which was scheduled to come out a couple of years later. Gaeta had been puzzling over how to film the movie's 'bullet time' sequence, in which Keanu Reeves dodges bullets in slow motion as the camera shots range over a cityscape. When Gaeta saw the Campanile movie, he realized that it was just what he needed. Gaeta hired Borshukov to import the Campanile movie technology to *The Matrix*, which went on to win an Academy Award for visual effects in 1999.

In 2000, Borshukov and two colleagues were awarded a Scientific and Technical Academy Award "for the development of a system for image-based rendering allowing choreographed camera movements through computer graphic reconstructed sets." Borshukov, who has made a career in the visual effects industry, says, "It's been great to be involved in a technology that's allowed filmmakers to do things they couldn't do before. The high point is for technology to be used by artists."

< *Solid models of the structures of the Berkeley Campus are generated from photographic images with a combination of automated tools and human assistance.*

TIMELINE

1980 CONTINUED 1982

MORE DEC VAXS *sold up to this time are running 3BSD UNIX than DEC's own VMS operating system.*

UNDER THE LEADERSHIP OF PROFESSOR SUSAN GRAHAM *and her students, gprof is developed. After 20 years, this remains a widely used performance profiling tool.*

4.2 BSD UNIX *incorporates the Internet's TCP/IP protocol suite—its first open source code implementation. It is distributed to over 1000 sites.*

BILL JOY LEAVES BERKELEY TO CO-FOUND SUN MICROSYSTEMS, *where much of the company's initial software architecture is inherited from Berkeley. Joy goes on to become a computing visionary, championing novel computer architectures, Network File Systems, and more recently Java. He is a Distinguished Alumnus of the CS Division.*

DAVID PATTERSON AND CARLO SEQUIN *pioneer simple architectures and exposing pipeline implementation to compiler (delayed branches) in their RISC-I Project. They coin the term Reduced Instruction Set Computer (RISC).*

SENIOR LECTURER MIKE CLANCY *publishes his essential textbook on introductory programming:* ***Oh! Pascal!***

MANUEL BLUM AND HIS STUDENT SILVIO MICALI *publish the landmark paper laying the foundations of pseudorandom generators.*

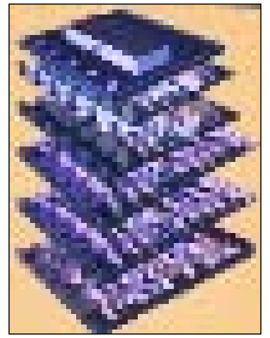
DAVID PATTERSON *wins the campus' Distinguished Teaching Award.*

1983

KEN THOMPSON *(Berkeley B.S. and M.S. in Computer Science in 1965-66) wins the ACM Turing Award: "For their (with Dennis Ritchie) development of generic operating systems theory and specifically for the implementation of the UNIX operating system." Ritchie and Thompson also win the National Medal of Technology in 1998.*

DAVID PATTERSON AND CARLO SEQUIN *take pipelining RISCs to the next level with their RISC-II Project, an architecture that is the direct ancestor of Sun's SPARC processor architecture, still in use today.*

THE SODA HALL WALKTHRU



^ *Professor Carlo Sequin and his students built a virtual Soda Hall long before the CS Division moved into the real building in 1994. His rendering made it possible for the future occupants of the building to influence its "look and feel" while also enabling cutting edge research in fast rendering of complex scenes.*

In the late 1980s, the Berkeley campus was gearing up to build a much-needed building for the computer science division. Carlo Séquin, the computer science professor heading the building committee, had already caught many flaws in the two-dimensional architectural floor plans. But, he thought, he could do his job much better if it were possible to take a virtual walk through a three-dimensional model of the building.

At the time, that was a tall order. A detailed computer model of the new building, Soda Hall, would contain millions of polygons. State-of-the-art walk-through programs could handle only on the order of tens of thousands of polygons in real time—any more and the succession of images would become jerky and intermittent.

However, Seth Teller, a Ph.D. student working with Séquin, realized that only a tiny fraction of the polygons in the Soda Hall model are visible from a given vantage point. So at any moment in a walk-through, the rendering program may need only about 1% of the model's polygons, Séquin says. "If you can find a way to avoid sending the other 99% through the graphics pipeline, you can make models that run 100 times faster or are 100 times more detailed than before," he says.

To figure out what is visible from any vantage point, Teller broke down the building's design into cells, the rooms and corridors, and portals, the windows, doors, and other openings through which parts of the building can be seen. Drawing on his background in theoretical computer science and computational geometry, Teller came up with an efficient algorithm to calculate which objects are visible at any moment in the walk-through. Previously, Teller says, graphics algorithms decided what was visible one polygon or even one point at a time. "The idea here was to make decisions about visibility on a chunk-by-chunk basis," he says.

Another challenge remained: to get the polygons to the rendering program quickly enough to refresh the scene 30 times every second, which creates a fluid, realistic animation. The Soda Hall model was too large for a computer's memory, so the team was storing it on the RAID disk system created by Randy Katz and David Patterson's architecture group. But data had to be fetched from the disk each time it was needed. To avoid slowdowns, Tom Funkhouser, another of Séquin's Ph.D. students, came up with an algorithm that predicts where a user is likely to move next, and brings the relevant portions of the building into memory before they become visible. Funkhouser sped the Walkthru further by creating models of the furniture at multiple levels of detail, and developing an algorithm to determine the appropriate level of detail for each item in a frame in real time.

The final program could render detailed animations of Soda Hall, complete with rooms, staircases and furniture, in real time. "Seth and Tom created a system that demonstrated to the world how this could be done efficiently and robustly, and this general approach has now been widely adopted," Séquin says.

Teller, now a professor at MIT, recalls that some of the most important lessons he learned at Berkeley were not specific techniques but a general approach to research. "Carlo gave me a lot of freedom to follow my interests, and had the philosophy that you should find a really hard problem, tackle it, and see what you learn," he recalls. "A lot of my own philosophy in advising my students has come from that." Funkhouser, now a professor at Princeton, agrees: "At Berkeley I learned a way of doing research where you pick a very hard problem, and then you put your stake in the ground where you want to be, and work towards that goal," he says. "That has served me well."

CAPTURING HUMAN MOTION

Of all the challenges animators face, perhaps the most elusive is human motion. A film audience may not notice if the geometry of a building is subtly off, but if an animated character walks in an awkward manner, they will pick up on it instantly. "Our perception is highly specialized for looking at human and animal motion," says Chris Bregler, who completed a Ph.D. in graphics at Berkeley in 1998. "If something is not natural there, we notice it right away."

When Bregler came to Berkeley as a graduate student, he started out working on object recognition problems. But motionless things couldn't hold his interest. "I wanted to work with things that live," he says. Animation was the obvious choice, he says, since it is "the part of graphics that brings things alive."

To tackle animation, Bregler departed from the traditional course of study. He enrolled in classes on hand-drawn animation and observed motion in many settings. "To bring things alive, you have to understand living things," he says. "To learn animation you have to watch movies, analyze Charlie Chaplin and Buster Keaton, go to the zoo and watch the animals, and watch random people in a plaza, even when they think it's weird that you're watching them. Once you get motion into your system, then you can start creating it."

Most animations of human motion are made in one of two ways: they are drawn out laboriously by hand, frame by frame, or they are assembled from "motion capture" data, obtained by tracking markers on a human being moving in a special studio. But suppose you want to animate the motion of a real human being, perhaps a celebrity or historical figure, who is no longer alive or is unavailable for time-consuming motion capture sessions? Bregler became fascinated by this problem, which lies on the boundary of graphics and computer vision, since it involves capturing information about motion from archive video footage instead of from motion capture data.

If you try to track the motion of a human body in a video pixel by pixel, then a movement will be a path in an extremely high-dimensional space, with three dimensions for each pixel. Bregler realized, however, tracking each pixel isn't necessary. Instead, it's possible to encode a walk using only about 20 free parameters that represent how the body's joints twist and turn, and how the body is being translated in space. Given a video of a human motion, Bregler used a combination of computer learning techniques and mathematical analysis to estimate the values of these parameters.

Once the parameters of a motion have been determined, it's possible to tweak the motion in a wide variety of ways—rearrange it into a new motion, for instance, or map it onto a new character. Bregler's first project dealt with analyzing human lip motions for lip-reading and realistic mouth animation. With colleagues at Interval Research Corp. in Palo Alto, Calif., Bregler turned that project into a full-face animation system called "Video Rewrite" that took old video footage of John F. Kennedy and rearranged it into a new video in which Kennedy ostensibly said, "I never met Forrest Gump."

Bregler next turned to a more complex challenge: analyzing full-body motions. Bregler's advisor, Jitendra Malik, steered him towards the famous photographic plates taken by Eadweard Muybridge, who in the mid 1880s captured the first photographic recordings of human and animal walk cycles. Bregler's program recovered the motions of a man and woman Muybridge had photographed, and mapped the motions onto animated figures. "We were able to take the way this woman and man walked 120 years ago, and bring it alive today," Bregler says. "No one had thought it was possible to process the Muybridge plates and animate them with that level of quality."

Bregler recalls fondly the relaxed, convivial atmosphere in the computer science division. "At Berkeley, half the research happens in [Café] Nefeli, with artificial intelligence people and graphics people and vision people and theory people all talking to each other," he says. "You're all just excited about certain ideas and you push them through. It's not the kind of department where there's one guy who builds a big empire."

Now a professor at New York University, Bregler is collaborating with experts in Laban movement analysis, a language for recording dance steps, to try to get to the heart of the myriad different styles with which the same gesture can be performed. He is also setting up a new motion capture laboratory in Manhattan. "We are currently the only motion capture lab in the New York area, and all these content producers and artists in the city want to use our lab, which is another great opportunity for interdisciplinary research," he says. "It's a really exciting time."

TIMELINE

1983 CONTINUED

WILLIAM KAHAN AND STUDENTS *develop a new C math library supplanting AT&T's in 4.2 BSD UNIX. It ran faster and more accurately on multiple VAX models with modern arithmetics conforming to the proposed IEEE Floating Point Standard. This evolved into fdlibm, the Freely Distributed Math Library promulgated by SUN before its initial release of Java in 1994.*

AT THE END OF THE FIRST DECADE OF THE CS DIVISION, FACULTY HAS GROWN TO 27.25 FTE.

1984

MANOLIS KATEVENIS WINS ACM DISSERTATION AWARD *for "Reduced Instruction Set Computer Architecture for VLSI." Currently at the University of Crete.*

ERIC BACH WINS ACM DISSERTATION AWARD *for "Analytic Methods in the Analysis and Design of Number-Theoretic Algorithms." Currently Professor of Computer Science at the University of Wisconsin-Madison.*

NIKLAUS WIRTH, *then a Professor at ETH Zurich (Berkeley CS Ph.D., 1963), receives the ACM Turing Award "For developing a sequence of innovative computer languages, EULER, ALGOL-W, MODULA and PASCAL. PASCAL has become pedagogically significant and has provided a foundation for future computer language, systems, and architectural research."*

ELWYN BERLEKAMP AND CHITOOR RAMAMOORTHY *win the IEEE Centennial Medal*

RICHARD FATEMAN'S GRADUATE STUDENTS *found Franz, Inc. to market Franz Lisp, developed at Berkeley in 1979. Franz, Inc. emerges by 2000 as the leading Lisp vendor.*

1985

DAVID PATTERSON LEADS THE SOAR PROJECT *to build a 32-bit RISC chip designed to run Smalltalk, demonstrating value of RISC architecture to support higher order programming languages.*

RICHARD KARP RECEIVES THE ACM TURING AWARD *"For his continuing contributions to the theory of algorithms including the development of efficient algorithms for network flow and other combinatorial optimization problems, the identification of polynomial-time computability with the intuitive notion of algorithmic efficiency, and, most notably, contributions to the theory of NP-completeness. Karp introduced the now standard methodology for proving problems to be NP-complete which has led to the identification of many theoretical and practical problems as being computationally difficult."*

INTELLIGENT SYSTEMS RESEARCH

TWENTY YEARS AGO, THE WORDS ARTIFICIAL INTELLIGENCE (AI) WOULD PROBABLY CONJURE UP IMAGES OF SUPERCOMPUTERS PLUGGING AWAY AT CHESS PUZZLES OR RIGIDLY FOLLOWING A SET OF RULES FOR MEDICAL DIAGNOSIS. BUT TODAY, INTELLIGENT MACHINES AT BERKELEY ARE MORE LIKELY TO TAKE THE FORM OF SELF-PILOTING HELICOPTERS OR VISION SYSTEMS THAT CAN TELL THE DIFFERENCE BETWEEN USING AN ATM AND ROBBING AN ATM. THE TREND AWAY FROM PURE COGITATION TOWARD MACHINES THAT MOVE AND PERCEIVE IS NOT MERELY COSMETIC—AT BERKELEY, IT HAS RESULTED IN A BREAKING DOWN OF BARRIERS BETWEEN PREVIOUSLY ISOLATED SUBFIELDS AND A RETHINKING OF THEIR FOUNDATIONS.

Take a look at the roster of a typical project at Berkeley, and you will find expert collaborations among all the formerly independent fiefdoms of control theory, machine learning, knowledge-based systems, and vision research. To a degree unmatched at other universities, Berkeley researchers are finding new ways to combine the subdisciplines, as they, for example, mesh machine learning with control theory to teach a robot how to fly, or combine ideas from vision research with statistics to sort and index image collections.

One aspect of this interdisciplinary ecumenism is mathematical and conceptual. For example, Peter Bartlett and Michael Jordan have led the unification of statistics and machine learning; Stuart Russell and his former postdoctoral scholar Daphne Koller (now at Stanford) have defined frameworks that combine logical and probabilistic reasoning; and the aptly named field of hybrid systems—a marriage of discrete automata theory and continuous control theory—was also developed here at Berkeley by Professors Shankar Sastry and Tom Henzinger.

Another important aspect of Berkeley's approach is the grounding of theoretical developments in concrete problems. For example, the world-leading vision group, headed by Jitendra Malik and David Forsyth, has pioneered a broad array of mathematical techniques and models. But these have emerged hand-in-hand with real systems ranging from the world's best reader of handwritten digits to a startlingly effective detector of naked people.

And chances are that if you learn about a particular area of AI, you'll learn it from a Berkeley-authored text. Berkeley's "whole-agent" view of AI is embodied in the text *Artificial Intelligence: A Modern Approach*, by Stuart Russell, coauthored with former Berkeley student and current Google executive Peter Norvig. *AIMA*, as the book is known, is considered essential reading for AI students in virtually all of the top computer science programs in the United States. Also dominant in its field is *Computer Vision: A Modern Approach* by David Forsyth and Jean Ponce. And Michael Jordan's textbook-in-progress on machine learning is already in wide use.

In 2002, Berkeley launched the Center for Intelligent Systems, to continue work on the conceptual underpinnings and encourage collaboration not only among the subdisciplines of AI, but also with biologists, cognitive scientists, and engineers. For inspiration, the Center looks to the past and present models of collaboration and subdiscipline cross-fertilization that have occurred at Berkeley, examples of which are presented here.

MOVING ABOUT: VEHICLES THAT PILOT THEMSELVES

"We started doing flying robots somewhat on a lark, in 1996," says EECS department chair Shankar Sastry. "Mobile robots on the ground were getting too passé." People told him, "That's pie in the sky, you'll never fly," he recalls.

Two years later, robots were flying off the roof of Cory Hall, and campus safety police booted the project off campus. Undaunted, the team pressed on. Now, much of the technology behind the increasingly common Unmanned Aerial Vehicles (UAVs), including the U.S. military's Predator aircraft, has come out of Sastry's lab.

Flying robots were not Berkeley's first foray into unmanned vehicles. In the mid-nineties, Jitendra Malik and students participated in a state-sponsored PATH (Partners for Advanced Transport and Highways) project to build an autonomous car. They designed an automated driving system that used stereo vision from two cameras mounted on a car to maneuver inside lanes and keep track of other vehicles. "The system has to identify cones and lane markers under all conditions, in bad lighting and bad weather—but at the same time, you cannot afford to have computationally expensive algorithms," says Malik. The team mastered the challenge, and after demonstrating autonomous driving in 1997 on a freeway in San Diego, Malik closed the book on self-piloting cars at Berkeley. "The basic science of freeway driving under autonomous control has been demonstrated, and now it's up to industry to carry on," Malik says.

Keeping a robot in the air, on the other hand, presents a host of challenges not faced by vehicles on the ground. Like the car, an airborne vehicle needs to be able to understand visual images, so that it can, for example, judge the distance to the pitching deck of a ship in the dark. But unlike the car, it needs to keep moving constantly for stability, and react and respond to shifting wind speeds and unexpected obstacles. For complex maneuvers, the robot can find itself in a place where the next move is not obvious, and it needs to be taught sequences of actions to achieve its end goal. "In flying, everything is much more challenging," Sastry says. "You have to unify all the elements to make them work."

All the elements came together for former Berkeley graduate student Andrew Ng in early 2002, when a helicopter trained to fly by Ng and Sastry's postdoc and former Berkeley graduate student Jin Kim, piloted itself through a series of competition-quality acrobatic maneuvers and pirouettes. The robot's skill exceeded even that of an expert human pilot. "As it was flying," says Ng, now an assistant professor at Stanford University, "we were all just standing there watching it fly. None of us was controlling it."

Even with a human manipulating the controls, helicopters are notoriously difficult to keep in the air. After hours and hours of practice, human pilots need unflagging vigilance to monitor the endless shifts in wind direction and speed, and the touch of a surgeon to tweak the controls to keep the helicopter air-

^ *Berkeley researchers have made use of a variety of robot helicopters to stress test their revolutionary algorithms for unmanned controlled flight.*

TIMELINE

1985 CONTINUED

SPRITE PROJECT, *led by Professor John Ousterhout, pioneers the development of Client-side Caching File Systems and Log-structured File Systems, in stark contrast to the existing "stateless" network file systems then available. CFSs are now widely used.*

THE IEEE 754 FLOATING POINT STANDARD *is adopted, based primarily on the design by* ***William Kahan****.*

THE THEORY WORLD COMES TO BERKELEY *when the Complexity Theory Year at the Mathematics Science Research Institute (MSRI) is held.*

1986

JOHN OUSTERHOUT *wins the UC Berkeley Distinguished Teaching Award.*

DAVID UNGAR WINS ACM DISSERTATION AWARD *for "The Design and Evaluation of a High-Performance Smalltalk System"*

RICHARD KARP *wins the campus' Distinguished Teaching Award.*

BILL JOY *wins the ACM Grace Murray Hopper Award for his contributions to Berkeley UNIX.*

MIKE STONEBRAKER AND LARRY ROWE'S POSTGRES PROJECT *pioneers the object-relational database model.*

1987

JOHN CANNY WINS ACM DISSERTATION AWARD *for his MIT work on "The Complexity of Robot Motion Planning." Canny is now the Jacobs Distinguished Professor of Computer Science at Berkeley.*

JOHN OUSTERHOUT WINS THE ACM GRACE MURRAY HOPPER AWARD *for his innovative VLSI design tools.*

RICHARD KARP *gives the SIAM von Neumann Lecture, honoring his contributions to mathematics and its applications.*

SUN RELEASES ITS SUN-4 SERIES OF COMPUTERS, *sporting the company's new SPARC architecture, which was based on the Berkeley RISC project.*

^ *Stills of UAV autonomous collision avoidance experiments (before and after the potential mid-air collision).*

borne. Because conditions shift constantly and helicopters are so sensitive to the environment, the robot can't be given a fixed control plan in advance. The machine needs to learn to respond in real time to any gusts or complications that come its way.

When Ng and Kim launched into the project in 2001, they already had in hand the tools to tutor the helicopter. The general-purpose PEGASUS program (Policy Evaluation-of-Goodness and Simulation Using Scenarios), developed earlier by Ng and his adviser Michael Jordan, was a reinforcement learning algorithm with the ability to teach controllers—the motion-directing brain of a robot—to deal with environments of arbitrary complexity. Ng and Kim used the program to teach the helicopter's controller how to respond to any situation. For eighteen hours, the controller was subjected to computer simulations of potential scenarios. When the controller responded well—righting a tipped over helicopter, for example, or lifting and hovering at the appropriate height, PEGASUS rewarded the controller. When the controller spun the simulated helicopter out of control by overreacting to an input, it was penalized. As the training progressed, the controller learned just how much it needed to adjust the helicopter's controls to obtain rewards and minimize punishments—and it carried those skills into the real world.

Helicopter acrobatics are among the more difficult tasks ever attempted by a machine learning system. In order to pull off the maneuvers and keep down the training time, Ng and Kim had to apply yet another theorem from their toolbox—this time, a theorem on reward shaping that Ng had derived earlier with Stuart Russell and graduate student Daishi Harada.

Shaping is an old idea from animal training: if you want to train a horse to jump over three fences, you give it a lump of sugar after each fence instead of waiting for it to jump over all three by chance. Sometimes, however, the little extra rewards may guide the horse into doing the wrong thing—such as running around in a circle so that it can jump over the first fence again and again to collect more and more sugar. What Ng, Harada, and Russell proved was that the horse will learn the right behavior if the shaping rewards correspond to the gradient of a conservative potential; that is, if the horse is penalized for going the wrong way as well as rewarded for going the right way. In the case of the helicopter, the shaping rewards guided it through the complex maneuvers and reduced the training time from millions of hours to just eighteen.

WORD AND PICTURES: NAVIGATING A SEA OF INFORMATION

The Internet contains pretty much everything we know—a librarian's dream—but it comes with a nightmare problem. How to unearth the gems from the detritus of the information age, the legal documents, newspaper articles, committee reports and general babble?

Berkeley computer scientists are assembling intelligent systems to pull order out of the digital chaos. The Digital Library Project is using evolving techniques in computer vision, machine learning and automatic ranking algorithms to streamline the discovery and organization of digital information, both text-based and visual.

On the Internet, information is cheap, says computer science Professor Robert Wilensky: for instance, anyone can easily publish a paper, a book, pictures, or a movie without worrying about the cost of materials or critical reviews. Yet the task of separating the wheat from the chaff falls squarely on the shoulders of the searcher. "Now, with dissemination virtually free, what's really expensive is attention," Wilensky says. Reviewers can help, but you need to be able to trust the reviewer's judgment. So Wilensky and his students created an algorithm to identify trend-spotting reviewers—reviewers who peg the overall numerical consensus often and early. "The algorithm picks out the 'good' reviewers automatically," says Wilensky. In one case study, the reviewers highlighted by the algorithm were the same reviewers the users on Epinions.com ranked highly.

Finding the words you want can be difficult, but refine your search enough and it's usually possible. Pictures, however, are a different story. Browsing the internet's offerings would be ideal, but it's tough even to know what to search for—let alone how to find it. "Think how hard it would be to shop in a bookstore that shelved its books by publication date," says Professor David Forsyth, "The problem with organizing pictures sensibly is that it's still very hard to tell what a picture is about." But the task is not insurmountable. Most pictures have associated words: captions, descriptive names, dates, or a dedicated web page nearby. Those words often describe aspects of the picture that are difficult to extract automatically. "The great thing is that the words and picture complement one another. The caption-writer will tell you that it's a rose, but usually doesn't say that its red," says Forsyth, "and computer vision can tell that it's red but may have difficulty telling that it's a rose."

Forsyth has teamed up with Michael Jordan to build systems that can use word and image information together to get a better take on what an item means. Once an image has been segmented into regions, each of whose color and texture is analyzed, these systems build descriptions of words and regions that occur together—for example, an orange, stripy region and the word "tiger." They have used these systems to build a clustered interface to the museum artifacts from the database of the Fine Arts Museum of San Francisco, in the hopes of enticing web surfers into visiting the museum's collection in person. The systems can also select potential pictures to illustrate a given bit of text. In one test, researchers selected some text from Moby Dick and asked the system to look for suitable illustrations from the Fine Arts Museum archive. The algorithm recovered a set of pictures of people in whaling boats, whaling.

These models can also predict words that go with pictures, and slap one-word labels on different regions within the image—tiger, for instance, or sky. "That's object recognition," says Forsyth. "These learning methods let us understand what's easy to recognize and how well we can recognize things. And what's really exciting is that these methods allow us to search for pictures of objects."

^ *Picture regions "translated" into words by a program built at Berkeley that analyzes captioned pictures automatically, using vision and learning methods. It's quite accurate, though it sees a "tree" on the facade of the building in the middle picture.*

TIMELINE

1988

DAVID PATTERSON AND RANDY KATZ, WITH SUPPORT FROM DAVID HODGES, *develop an innovative memory system, integrating multiprocessor snooping caches with virtual memory management in the SPUR multiprocessor chip set. They originate the term "snooping caches" widely used today to describe multiprocessor cache consistency mechanisms. Several project students become faculty at leading universities:* ***Susan Eggers*** *(Washington),* ***Mark Hill*** *(Wisconsin),* ***James Larus*** *(Wisconsin then Microsoft),* ***Corinna Lee*** *(Toronto and then Industry),* ***David Wood*** *(Wisconsin), and* ***Ben Zorn*** *(Colorado then Microsoft).*

MIKE STONEBRAKER AND EUGENE WONG *honored with the ACM System Software Award for INGRES.*

JEROME FELDMAN *joins the CS faculty from his position at the University of Rochester, to lead the newly formed International Computer Science Institute in Berkeley. ICSI provides an institutional venue for collaboration between the Berkeley CS community and the international research community.*

1989

WILLIAM KAHAN WINS THE ACM TURING AWARD *"For his fundamental contributions to numerical analysis. One of the foremost experts on floating-point computations. Kahan has dedicated himself to 'making the world safe for numerical computations.'" Most computers built since 1980 incorporate the IEEE 754 Floating Point standard championed by Kahan.*

ELWYN BERLEKAMP *wins the IEEE Kobayashi Award.*

LOFTI ZADEH *wins the Honda Prize.*

RANDY KATZ AND DAVID PATTERSON, *pioneer multi-disk server attachment in the RAID Project. They coin the term Redundant Arrays of Inexpensive Disks (RAID). The RAID industry is now a $25 billion per year market segment.*

MIKE STONEBRAKER FOUNDS ILLUSTRA *to commercialize the Postgres technology. Eventually acquired by Informix (founded by* ***Roger Sippl****, a Berkeley undergrad). IBM in turn acquires Informix.*

DURING THE DECADE OF THE 1980S, *twelve new faculty members joined the CS Division.*

1990

RICHARD KARP *wins the von Neumann Theory Prize.*

BERKELEY UNIX PROJECT *releases 4.3 Reno BSD, providing the reference implementation for well-behaved "correct" Internet protocols.*

METRICS OF EXCELLENCE

TURING AWARD WINNERS

FACULTY

- KARP, RICHARD (1985)
- KAHAN, WILLIAM (VELVEL) (1989)
- BLUM, MANUEL (1995)

GRADUATES

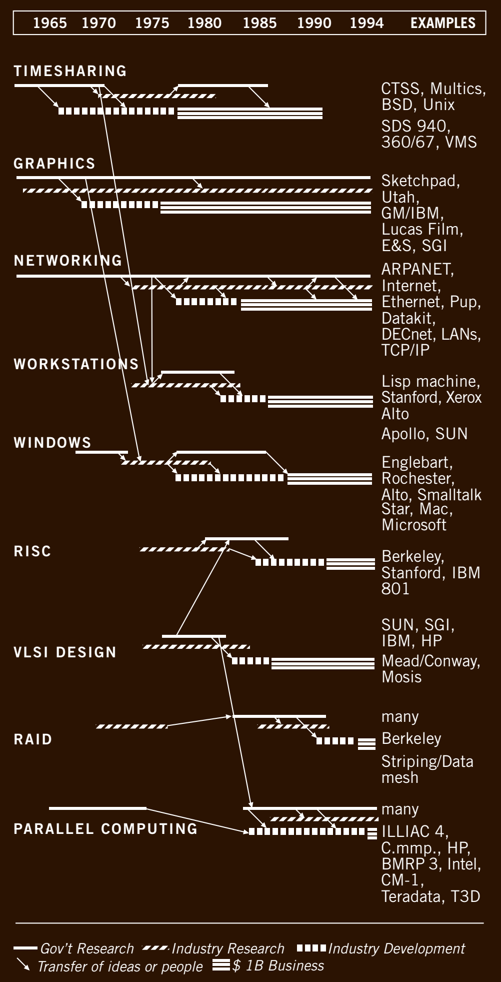
- THOMPSON, KEN (1983)
- WIRTH, NIKLAUS (1984)
- LAMPSON, BUTLER W. (1992)
- ENGELBART, DOUGLAS (1997)
- GRAY, JAMES (1998)
- ADLEMAN, LEONARD (2002)

ACM DISSERTATION WINNERS

GRADUATES

- KATEVENIS, MANOLIS G.H. (1984)
- BACH, ERIC (1984)
- UNGAR, DAVID (1986)
- NISAN, NOAM (1990)
- GIBSON, GARTH (1991)
- LUND, CARSTEN (1991)
- ROSENBLUM, MENDEL (1992)
- SUDAN, MADHU (1993)
- ARORA, SANJEEV (1995)
- MCCANNE, STEVEN R. (1997)
- BALAKRISHNAN, HARI (1998)

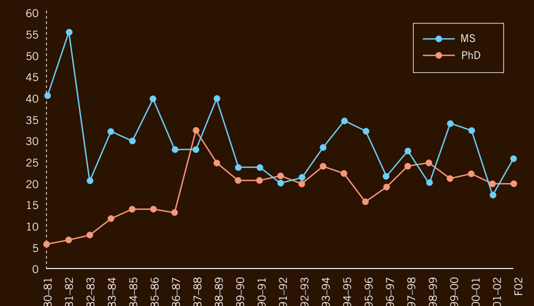
ROLE OF UNIVERSITY R & D IN CREATION OF INDUSTRIES



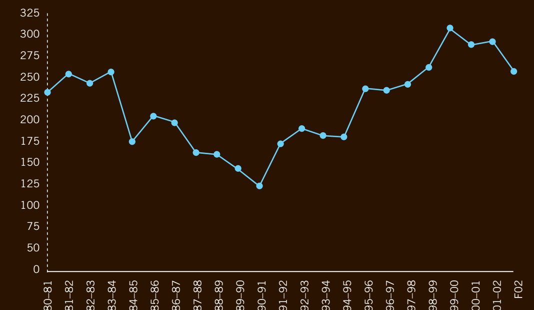
In the National Academy of Science's 1995 report *Funding a Revolution: Government Support for Computer Research*, CSTB highlighted the path from university research to industrial products in various areas of computer science. Research at Berkeley had a major impact on several of these, including BSD (*Berkeley Standard Distribution*), RISC (*Reduced Instruction Set Computers*) and RAID (*Redundant Arrays of Inexpensive Disks*).

BERKELEY COMPUTER SCIENCE

MASTERS AND PhDS GRANTED—COMPUTER SCIENCE



BS/BA DEGREES GRANTED—COMPUTER SCIENCE



TOTAL DEGREES AWARDED: MS: **701**; PhD: **441**; BA/BS: **5,011**

CHAIRS OF COMPUTER SCIENCE

- **1973–1975** RICHARD KARP
- **1975–1977** ELWYN BERLEKAMP
- **1977–1980** MANUEL BLUM
- **1980–1983** CARLO SEQUIN
- **1983–1987** DOMENICO FERRARI
- **1987–1990** RICHARD FATEMAN
- **1990–1993** DAVID PATTERSON
- **1993–1996** ROBERT WILENSKY
- **1996–1999** RANDY KATZ (CHAIR, EECS)
- **1999–2002** CHRISTOS PAPADIMITRIOU
- **2002–PRESENT** JITENDRA MALIK

BERKELEY PhDS CURRENTLY TEACHING AT TOP SCHOOLS

University of Washington
Borriello, Gaetano
Eggers, Susan
Gribble, Steven
Ivory, Melody
Ladner, Richard
Ruzzo, Walter

UC Berkeley
Demmel, James
Garcia, Daniel
Hearst, Marti
Katz, Randy
McMains,Sara
Vazirani, Umesh
Wagner, David

Stanford University
Fox, Armando
Gill, John
Jurafsky, Dan
Kozyrakis, Christos
Motwani, Rajeev
Ng, Andrew
Rosenblum, Mendel

Cal Tech
Perona, Pietro
Umans, Christopher

UCLA
Klinger, Allen
Majumdar, Rupak
Potkonjak, Miodrag
Tamir, Yuval

University of Michigan
Chen, Pete
Lafortune, Stephane
Mao, Morley

University of Wisconsin
Arpaci-Dusseau, Andrea
Arpaci-Dusseau, Remzi
Bach, Eric
Chenney, Stephen
Hill, Mark
Klein, Sheldon
Wood, David

University of Illinois at Urbana-Champaign
Erickson, Jeff
Hwu, Wen-mei
Lumetta, Steven
Wah, Ben
Yu, Yizhou

University of Texas, Austin
Dahlin, Mike
Dhillon, Inderjit
Warnow, Tandy
Zuckerman, David

MIT
Asanovic, Krste
Balakrishnan, Hari
Devadas, Srinivas
Goldwasser, Shafi
Lampson, Butler
Madden, Sam
Micali, Silvio
Rubinfeld, Ronitt
Sipser, Michael
Sudan, Madhu
Teller, Seth

Cornell
Birman, Ken
Schmoys, David

Harvard
Grosz, Barbara
Mitzenmacher, Michael
Seltzer, Margo
Welsh, Matt

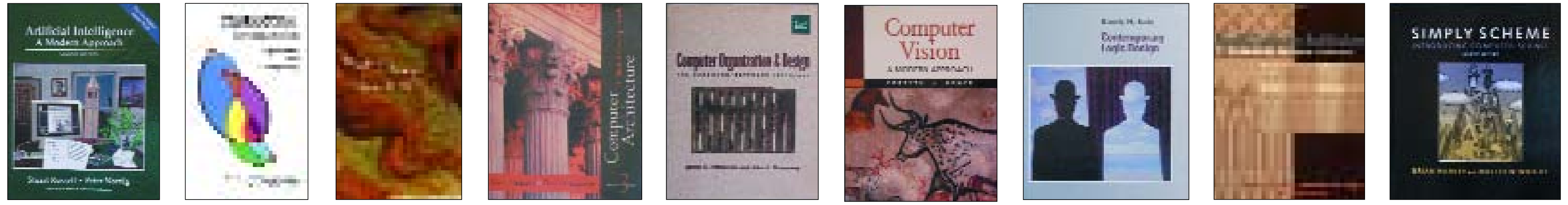
Yale
Angluin, Dana
Krishnamurthy, Arvind

Carnegie Mellon University
Cooper, Eric
Efros, Alexei
Fedder, Gary
Gibson, Garth
Goldstein, Seth
Gupta, Anupam
Harchol-Balter, Mor
Heckbert, Paul
Miller, Gary
Rudich, Steven
Seshan, Srini
Song, Dawn
Zhang, Hui

Princeton
Arora, Sanjeev
Funkhouser, Tom
Malik, Sharad
Wang, Randy

IMPACT ON EDUCATION: *INFLUENTIAL PUBLICATIONS BY BERKELEY FACULTY*

Berkeley Computer Science has always led in teaching, and not just in the classroom. The CS faculty have been authors of many standard textbooks in subdisciplines of Computer Science such as programming, hardware design, architecture, artificial intelligence, and computer vision.



ROC-SOLID SYSTEMS

Patterson has no plans to sit back and watch that grim future come to pass. For the last few years Patterson—a key player in earlier industry-shaping Berkeley projects such as RISC and RAID—has set his sights on making computers more reliable. He and former Berkeley Ph.D. student Armando Fox, now an assistant professor at Stanford, have set out to help redefine the industry's approach to dependability, focusing initially on the stability of large computer networks.

Until now, reliability efforts have been concentrated in a handful of specialized subfields—notably memory stability and the highly fault-tolerant computer systems that operate space shuttles. Designers of the shuttle systems have drawn on techniques such as hardware redundancy, error-spotting software, and the fine-tuning of individual components to achieve an amazing consistency. But attempts to map this success to the dynamic sprawl of networks have failed: Internet services suffer 8 hours or more of downtime each year. Since an hour of downtime on a large commercial network can cost millions of dollars in lost business, financial costs eclipse mere frustration.

What's wrong with the current fault-tolerant approach? Patterson argues that it is subject to one striking limitation. It focuses on error prevention: but to prevent errors you must know what errors to expect. For networks, which are subject both to constant fluctuations and to the fallibility of human administrators, this approach hits a wall. In the end, error will always occur, Patterson says. So to make a system more stable, he argues it must be built to bounce back from—not to prevent—the mistakes no one could have anticipated.

To lay out ground rules for stable systems, Patterson and Fox have joined forces on what they have named the Recovery-Oriented Computing (ROC) project. While Fox's group has been working on ways to restart isolated hardware and software components of a large network without having to shut down the whole system, Patterson's contingent is focusing on the impact of human operators on system stability.

Operator error is a surprisingly big deal: after studying three independent Internet servers and one public telephone switching network, Berkeley students found that more than half of the errors serious enough to be spotted by ordinary users were caused by operator mistakes. And on two of the Internet servers, operator error caused 75% of the total time the system was offline. "Traditional approaches to dependability largely ignore operator error, yet it is likely the biggest challenge," Patterson says. Since Patterson's team can't magically make human operators perfect, it is working on the next best thing: a systemwide undo function to erase operator mistakes.

JUST UNDO IT!

Word processor designers solved the problem of unknowable mistakes long ago, when they created an "undo" function to allow users to retract any action they immediately regret. Patterson's team argues that that is just what's needed for large networks—but on a grander scale. They have created a global undo for an email server that acts like a time-machine: it rolls the system state back to a point before the mistake occurred, allows the operator to repair the mistake, and then rolls the system forward to the present.

Berkeley Ph.D. student Aaron Brown refers to these three time-machine steps as the "three Rs": rewind, repair, and replay. Individually, none of the three Rs is novel—each has appeared in commercial software—but only the Berkeley group has integrated all three. Brown and Patterson have put together the first test case of the three Rs on an e-mail storage

^ Above and top of following pages: Equipment from the Berkeley Experimental Machine Room, where the ROC project is put to the test.

TIMELINE

1990 CONTINUED

BERKELEY COMPUTER SCIENTISTS DEVELOP VLSI-BAM: *a single chip architecture for Prolog.*

DURING THE EARLY 1990S, *Berkeley UNIX forms the foundation for several ports of UNIX to PCs: FreeBSD, NetBSD, BSDi.*

DAVID PATTERSON AND STANFORD PROFESSOR JOHN HENNESSY *publish their landmark textbook Computer Architecture: A Quantitative Approach, Morgan Kaufmann Publishers, Inc., 1990.*

PIETRO PERONA (BERKELEY Ph.D. 1990) AND JITENDRA MALIK INTRODUCE *the anisotropic diffusion partial differential equation for image processing.*

NOAM NISAN WINS ACM DISSERTATION AWARD *for "Using Hard Problems to Create Pseudorandom Generators."*

1991

LOFTI ZADEH LAUNCHES BISC *(Berkeley Initiative in Soft Computing): a consortium of fuzzy logic, neurocomputing, evolutionary computing, probabilistic computing and machine learning. Soft computing grows rapidly in visibility and importance.*

DAVID PATTERSON WINS THE ACM KARL V. KARLSTROM OUTSTANDING EDUCATOR AWARD *for his innovative project-oriented hardware courses and his widely-used textbooks.*

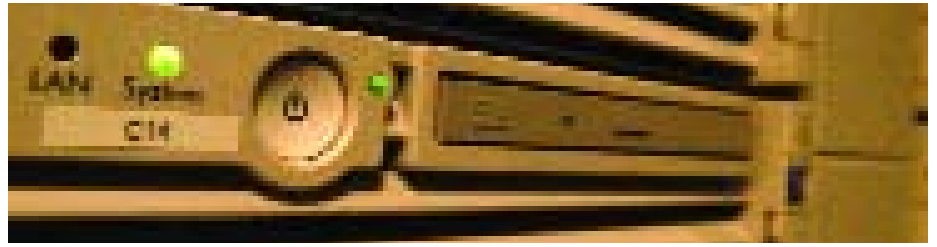
ELWYN BERLEKAMP *wins the IEEE Hamming Medal.*

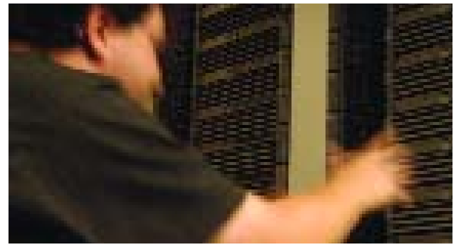
GARTH GIBSON WINS ACM DISSERTATION AWARD *for "Redundant Disk Arrays: Reliable, Parallel Secondary Storage." Currently at CMU.*

JAMES DEMMEL AND WILLIAM KAHAN *receive an "Outstanding Paper" prize from the SIAM Activity Group in Linear Algebra. Follow-on papers by Demmel and collaborators led to Linear Algebra software that is substantially more accurate with little sacrifice of performance.*

1992

MENDEL ROSENBLUM WINS ACM DISSERTATION AWARD *for "The Design and Implementation of a Log-structured File System." Currently at Stanford University.*



server with five thousand imaginary users programmed to send, receive, open and delete mail.

The undo allows the system operator to repair a wide range of problems, as diverse a misconfigured spam filter, a virus attack, or an accidentally deleted user mailbox. The function maintains system snapshots, together with a timeline that captures all the ordinary user actions; these allow the operator to rewind the computer to any instant in time in the last month. Once back at the point where the error first occurred, the operator's actions are unrestricted—she can use trial and error, error injection, or pinpointing tools to diagnose the problem. After the system has been repaired, the undo apparatus replays and restores all the harmless actions of ordinary users.

The snapshots and timeline incur storage costs—but since storage is plentiful and cheap, this shouldn't create undue financial burdens, Brown says. "A 120 gigabyte disk costs about $110 today, and even if you need several of them for redundancy, it's still a small price to pay compared to what you might have to pay in the cost of lost productivity."

BENCHMARKING THE COMPETITION

To justify the additional cost of an undo command, Patterson's team is looking for ways to prove that it increases availability and worker productivity. In contrast to the well established "industry benchmarks" that measure computer speed and performance, computer science lacks a universal method to measure reliability. Most companies instead compute the 'availability' of their systems—the assessment of whether the system can respond moment-by-moment to user requests.

But conditions during these test runs are more ideal than real—the computers are kept isolated from humans. Berkeley ROC members are pushing to create more meaningful benchmarks that include the operators themselves. "We want to keep people in the loop," Brown says, "The idea is to go in while the benchmark is running and actively break the system, then give it to people to fix." To put this goal into practice, the team is selecting realistic problems to inject into the e-mail storage server and developing ways to familiarize operators with the system so that they can cope with surprises. Eventually, the team hopes such benchmarking techniques should be applicable to a wider range of recovery situations.

COLLABORATIVE ENTERPRISE

While Patterson's group pursues the challenge of dealing with operators, Fox and students have focused on slashing the amount of time that restarts and shutdowns pull a system off line. All crashes are not created equal, they've discovered. For example, their analysis of a network that records satellite data showed that the impact of downtime is nonlinear: during a satellite fly-over, a two-minute shutdown may have few repercussions, but a ten-minute shutdown can cause a data-loss catastrophe. The team built tools to shrink the downtime from ten minutes to two, by shutting down and restarting only as much of the system as is needed to fix a problem.

The Berkeley and Stanford groups gather for semi-annual retreats with more than 40 industry representatives. "Berkeley has a tradition of tackling real-world problems and forging strong connections with industry," Brown says. So far, industry representatives have been encouraging, Brown says. "These are problems real people are having, and if we're able to solve them, we will have an impact."

Patterson hopes that defining recovery-oriented ideals and building the tools to put them into practice will inspire system designers in the wider community. "Just the idea that we should be resilient against unexpected errors by having a margin of safety, even if it makes things more expensive, is foreign to many scientists," he says. "But once they hear it, people find it thought-provoking and begin to think about what we should be doing differently." With time, the team hopes the ideas underlying ROC will ripple out into industry to make the lives of system operators and ordinary users a little less frustrating.





__Above:__ Berkeley computing systems researchers have pioneered the concept of the research retreat, where students, faculty, and research project sponsors from industry and government, spend several days together, reviewing project developments, discussing technical issues over meals, and having fun while building team spirit through group activities.

The group also holds 10 year reunions for past projects, where project teams get together to share how their professional and personal lives have developed since their graduate student days. This is the SPUR research team, reassembled in 1999.

__L-R, first row:__ Professor David Patterson, Garth Gibson (Professor, CMU), Professor Richard Fateman, Shin Kong (Silicon Image), Brent Welch (Electric Cloud);

__second row:__ Former Professor John Ousterhout (Electric Cloud), Mark Hill (Professor, UWisconsin), Susan Eggers (Professor, UWashington), Paul Hansen (SUN Microsystems), B. K. Bose, Mike Nelson, Mendel Rosenblum (Professor, Stanford), Ben Zorn (Microsoft)

__last row:__ David Wood (Professor, UWisconsin), James Larus (Microsoft), Luigi Semenzano, Professor Randy Katz, Scott Richie (Consultant), George Taylor, Andrew Cherenson (SGI), Corinna Lee (ATI), Ken Lutz (DARPA Gigascale Center), David Lee (Silicon Image), Professor David Hodges

TIMELINE

1992 CONTINUED

BUTLER LAMPSON *(Berkeley Ph.D., 1967; member of Berkeley Faculty in early 1970s) received the ACM Turing Award "For contributions to the development of distributed, personal computing environments and the technology for their implementation: workstations, networks, operating systems, programming systems, displays, security and document publishing."*

THE FOUNDATIONAL PUBLICATION OF APPROXIMATION THEORY IS PUBLISHED: *Arora, Lund, Motwani, Sudan, Szegedy, "Proof verification and hardness of approximation problems," Proceedings of the 33rd IEEE Symposium on Foundations on Computer Science, 1992, 14-23.*

DAVID CULLER AND HIS STUDENTS *develop Active Messages, becoming the de facto standard for structuring distributed computations in multicomputer systems.*

FIRST RELEASE OF LAPACK, *an essential linear algebra numerical software library now widely used by industry and commercially supported by many companies. Joint work between **James Demmel** and colleagues at the University of Tennessee, New York University, NAG, Argonne National Lab, and Rice University.*

RANDY KATZ *wins the UC Berkeley Distinguished Teaching Award.*

1993

MADHU SUDAN WINS ACM DISSERTATION AWARD *for "Efficient Checking of Polynomials and Proofs and the Hardness of Approximation Problems." Currently at MIT.*

JAMES DEMMEL WINS THE WILKINSON PRIZE, *recognizing his contributions to the field of numerical software and computational science.*

LED BY RANDY KATZ AND DAVID PATTERSON, *RAID-II pioneers Network-Attached Storage, via switched HIPPI fabric, predecessor of Fibre Channel. The project incorporates **John Ousterhout** and **Mendel Rosenblum's** Log Structured File System to achieve orders of magnitude greater I/O throughput than existing file server architectures.*

DAVID CULLER AND KATHERINE YELICK *lead a team that develops Split-C, one of the most influential parallel programming languages.*

SUSAN GRAHAM AND HER STUDENTS STEVE LUCCO AND ROBERT WAHBE *invent software fault isolation. Lucco and Wahbe's start-up is later acquired by Microsoft, and they transition software fault isolation into commercial realization in Microsoft's .Net.*

CS 160

IT'S NOT EVERY COMPUTER SCIENCE COURSE THAT SENDS STUDENTS OUT TO INTERVIEW FIREFIGHTERS OR ESCORT BLIND SHOPPERS THROUGH A SUPERMARKET. BUT IN COMPUTER SCIENCE 160—"USER INTERFACE DESIGN AND DEVELOPMENT"—BERKELEY UNDERGRADS COME FACE-TO-FACE WITH A PART OF THE COMPUTING WORLD THAT, TO MANY COMPUTER SCIENTISTS, IS A FACELESS ABSTRACTION: **REAL, LIVE USERS.**

^ *CS 160 students present their updated digital whiteboard, "Firewall," to El Cerrito Firefighters.*

CS 160 reflects the growth of Human-Computer Interaction as a field. And, as Scott Klemmer, a graduate Teaching Assistant for the course, explains, it signals a new mindset about how computer engineers should be educated. "Twenty-five years ago, just getting a computer to do what you wanted was so difficult that education was oriented towards making computers work," he says. Today anyone can buy a powerful computer for under $500, so "the fundamental questions of our discipline have shifted," Klemmer says, "from asking whether something is technologically possible to asking whether it makes sense to the people who use it."

Unlike in most computer science courses, where students are assigned several small projects, CS 160 students are challenged to devise entirely new user-interfaces, or to identify problems with existing ones that they wish to solve. Working together in small groups, the students take their proposals through all stages of design, prototype and evaluation. Prospective users are involved from the start: students visit intended users in the field, test prototypes with them after each design round, and study psychological issues that affect designs' usability. Recent projects have included a system for remotely checking the security of one's home while traveling, a handheld-organizer application that finds local restaurants, and a web interface to bring together would-be carpoolers.

Since its inception seven years ago, "User Interface Design and Development" has spawned an enthusiastic core of alumni, many of whom credit the course with helping them land a job. It's often called "the most useful class taken at Berkeley," in terms of the work its students are doing today in industry. And a number of CS 160 class projects have been visionary, with similar products cropping up on the commercial market within a few years. That's no coincidence, since a key guideline for CS 160 projects is that they be able to work with current technology, or technology that is expected shortly to be commonplace. "You're not writing science fiction about things that will never come to be," says Corey Chandler, who took CS 160 in 2000. "That's part of what makes the course so enticing."

USERS AND DESIGN

Chandler's team learned early on how crucial it is to involve users in the design process. Their initial idea was a visual display for firefighters' helmets showing instructions from their commanding officer or information about the crewmates' locations. But the proposal was energetically rebuffed by the firefighters the team consulted. "We already wear 40 pounds of gear when we enter a burning building," the firefighters explained. The last thing they wanted was some fancy, fragile gadget floating in front of their eyes. "They said that to experience what a firefighter goes through, you should put on a blindfold, then get inside a dryer and set it on spin," Chandler recalls. "That's how crazy it is."

Discouraged, the students considered scrapping the whole project. But further talks with the firefighters pointed to a new angle: design of a system supporting the commander of a firefighting team. The students' final design was a digital whiteboard named "Firewall" that would display the location of firefighters in a building and help their commander track such vital statistics firefighters' body temperatures and the amount of oxygen left in their tanks.

The intense focus on adjusting and readjusting a design's initial conception and execution in order to meet user needs is one of the most important lessons of CS 160, says former student Sunny Consolvo. In 1997, when Consolvo's team invented an electronic cookbook, she fully expected it to fly through user testing, despite admonitions that users always want to make changes. But Consolvo's users pounced on details her group had never thought of. "I remember a user looking for something in the upper-left corner of the screen, and it was there about an inch from where they were looking, but they never found it," she says. "It was driving us crazy, and we felt like saying, 'Look, it's right there, just move your finger down'. But you can't do that—you have to understand what it is about the design that made them unable to find it."

The experience was a revelation for Consolvo. Initially contemplating graduate study in graphics, Consolvo had long been frustrated by how badly many user interfaces work—but she'd never known there was an entire discipline dedicated to solving such problems. "I would often see people struggling just to create a Word document," she says. "They were perfectly smart people who were having trouble with computers." After completing CS 160, decided to devote herself to interface design for ubiquitous computing; she now works at Intel Research/Seattle, in Washington.

APPLICABILITY

At an end-of-semester course fair, CS 160 students present posters on their projects to industry representatives. Many students land jobs or internships straight out of the course. Corey Chandler, for instance, got a call from Microsoft, and ended up doing two internships there. "It was pretty much a result of the class," he says. Chandler now hopes to become a professional user-interface designer. And although he and his teammates are no longer working on the Firewall project, the professor who taught them, James Landay, is pursuing their idea as part of a long-term research project to develop better ways to support emergency workers, sponsored by the University of California's Center for Information Technology Research in the Interest of Society.

Helping real users take better advantage of computers is also the research focus of Computer Science's newest HCI faculty member, Jennifer Mankoff. She works on the ambiguous inputs that plague efforts to perfect recognition-based input systems. Such systems include speech-, pen-, and gesture-recognition, as well as computer vision, all of which play a growing role in the development of mobile devices, smart rooms, and applications for the handicapped (who can use neither keyboard nor mouse). Projects arising out of Mankoff's CS 160 courses are expected to be as innovative and forward-looking as those that have gone before. For Mankoff shares the goals of this unique course—to make engineering students more sensitive to the goals of interface design, and to lead them to an awareness that, as Landay has said, "designers are there to make things work for real people."

CS 160 students definitely carry a fresh perspective to industry, Scott Klemmer adds. "The techniques the course teaches are going to become the standards used in industry 20 years from now."

^ *Professor Jennifer Mankoff (above) specializes in assistive technology and ubiquitous computing.*

TIMELINE

1993 CONTINUED

RANDY KATZ PUBLISHES HIS LANDMARK TEXTBOOK, *CONTEMPORARY LOGIC DESIGN*, *Addison-Wesley, 1993. It becomes the standard introduction to hardware design, used at over 200 universities, and the first to integrate computer-aided design tools with logic design.*

JOHN HENNESSY AND DAVID PATTERSON *publish their second landmark textbook: Computer Organization and Design: The Hardware-Software Interface, Morgan Kaufman, 1993.*

AS THE END OF THE SECOND DECADE *of the Division's existence, the faculty stands at 29.75 FTE, continuing to grow despite a large number of retirements in the early 1990s.*

1994

MIKE STONEBRAKER HONORED AS FIRST WINNER OF THE ACM SIGMOD INNOVATIONS AWARD, *recognizing him as the premier innovator in database research.*

ETHAN BERNSTEIN AND UMESH VAZIRANI *deliver the paper laying the foundations of Quantum Computation, "Quantum Complexity Theory," SIAM J. Computing October 1997.*

STUART RUSSELL AND PETER NORVIG *(Berkeley Ph.D. 1985) publish* Artificial Intelligence: A Modern Approach *(Prentice Hall, 1995), which becomes the bible of AI. It has been adopted at over 720 universities in 80 countries, spurring a fundamental shift in the field of AI.*

MIKE STONEBRAKER BEGINS THE MARIPOSA PROJECT, *one of the first open-source distributed systems to be built using economic computing mechanisms.*

THE COMPUTER SCIENCE DIVISION *moves into its new home in Soda Hall, largely designed with the aid of Professor Sequin's architectural visualization software.*

1995

SANJEEV ARORA WINS ACM DISSERTATION AWARD *for "Probabilistic Checking of Proofs and Hardness of Approximation Problems." Currently at Princeton.*

MANUEL BLUM *(at Berkeley from 1968-2001, currently at CMU) wins the ACM Turing Award "In recognition of his contributions to the foundations of computational complexity theory and its application to cryptography and program checking."*

OVER THE PAST 30 YEARS, THE BERKELEY THEORY GROUP HAS PLAYED AN INSTRUMENTAL ROLE IN DEVELOPING IDEAS AT THE BACKBONE OF THEORETICAL COMPUTER SCIENCE. NOW, BERKELEY THEORETICIANS ARE TURNING THEIR ATTENTION OUTWARD. AS MORE AND MORE FIELDS OF SCIENCE ARE PROVING TO CENTER AROUND INFORMATION, FOUR BERKELEY RESEARCHERS ARE TURNING THE POWER OF THEORETICAL COMPUTER SCIENCE ON THE WIDER SCIENTIFIC WORLD. "COMPUTATION IS A NEW LENS FOR SCIENCE," SAYS PROFESSOR ALISTAIR SINCLAIR.

< Inside the Quantum computer lab, 2003

INTERDISCIPLINARY THEORY

QUANTUM COMPUTING

Ten or twenty years ago, says Professor Christos Papadimitriou, computer science was a "second-class citizen," regarded by the other sciences mainly as a source of useful computer programs. This has completely changed in the last decade, Papadimitriou says. "Computer science is being hailed as an essential discipline in its own right, one that can offer the other sciences not just computational power but new ideas," he says.

Sinclair, Papadimitriou, and Professors Richard Karp and Umesh Vazirani have jointly received grants from NSF and DARPA to export the ideas of theoretical computer science to fields as disparate as statistical physics, the economics of the Internet, biology and quantum physics. "We believe computer science can transform in a very profound way how other disciplines are going about their business," Papadimitriou says. "That is our rallying cry."

When Umesh Vazirani became interested in quantum computation just over a decade ago, the field—if it could be called a field—hovered on the fringe of computer science. In the intervening years, due in large part to critical input from Vazirani and other Berkeley researchers, it has emerged as a robust paradigm that radically extends the limits of computation.

The extended Church-Turing thesis, developed in the 1960s and 1970s, postulates that there is essentially only one model of computation: that no computer, no matter what its design, can solve any problem significantly faster than a simple model of computation known as the probabilistic Turing machine. For decades the thesis was treated as gospel, Vazirani says. "It was accepted as unquestionably true."

In 1992, Vazirani became intrigued by the question whether quantum physics—in which particles can exist in a superposition of many states, and distant particles can influence each other instantaneously—might offer a computational framework outside the limits of the Church-Turing thesis. The celebrated physicist Richard Feynman had pointed out in 1982 that classical computers appeared incapable of simulating a several-particle quantum system efficiently. Turned around, this suggests that a computer that exploited quantum mechanics could solve problems beyond the scope of a classical computer. "It seemed to me that this was the first and only time we saw a place where the extended Church-Turing thesis might be violated," Vazirani says. "It had the potential to challenge the foundations of computer science."

In 1993, Vazirani and Ethan Bernstein, then a Berkeley Ph.D. student, published a paper presenting a model for a digital quantum computer that is universal, in the sense that it could be programmed to simulate any other quantum computer. What's more, they established that this model violates the extended Church-Turing thesis. "It showed that from the viewpoint of the foundations of computer science, quantum computation is a very significant model," Vazirani says.

Vazirani and Bernstein's work introduced a new class of problems into computational complexity theory: BQP, the collection of problems that can be solved by a quantum computer in polynomial time. Since that time, researchers have shown that a number of important hard problems, including factoring, lie in that class. That means, in particular, that a quantum computer could break any cryptographic scheme that depends on the difficulty of factoring large numbers.

Vazirani and Bernstein proved that BQP contains all the problems a classical computer can solve in polynomial time. Vazirani has now set his sights, among other projects, on figuring out where BQP sits in the infinite hierarchy of complexity classes with problems of greater and greater difficulty. "That would give us a better sense of how powerful quantum computers are," he says. It would give scientists a handle on just how much work nature puts into each quantum mechanical computation, Vazirani says. "In a sense, it would be telling us how hard nature thinks."

TIMELINE

1995 CONTINUED

LOFTI ZADEH IS AWARDED THE IEEE MEDAL OF HONOR *"For pioneering development of fuzzy logic and its many diverse applications."*

STUART RUSSELL WINS IJCAI COMPUTERS AND THOUGHT AWARD *for his work on bounded rationality, which offers a new foundation for intelligent systems.*

ERIC BREWER AND HIS STUDENT PAUL GAUTIER *develop Inktomi, a scalable Internet search engine, as an application on top of NOW. Gautier and Brewer found Inktomi (acquired by Yahoo 2003).*

ERIC BREWER AND RANDY KATZ LEAD THE BARWAN PROJECT *to develop Client-Proxy-Server Architectures and Wireless Overlay Mobility and Wireless Transport Protocols. Grad students involved include:* ***Armando Fox*** *(Stanford),* ***Steve Gribble*** *(Washington),* ***Srini Seshan*** *(CMU) and* ***Hari Balakrishnan*** *(MIT).*

THE POSTGRES PROJECT *enters the open-source mainstream as PostgreSQL, now a leading open-source database engine developed and used worldwide. By 2001, Red Hat releases Red Hat Database on Linux, an "enhanced version" of PostgreSQL.*

JOE HELLERSTEIN, JEFFREY NAUGHTON AND AVI PFEFFER *(Berkeley Ph.D. '95, now at Harvard) publish first paper on Generalized Search Trees (GiST). GiST software contributed to PostgreSQL, subsequently underlies several open-source Geographic Information Systems. Informix later integrates GiST into their core database engine.*

1996

JOHN WAWRZYNEK AND HIS STUDENT KRSTE ASANOVIC *develop T0 (Torrent-0), the world's first single-chip fixed-point vector microprocessor designed for multimedia, human-interface, neural network, and other digital signal processing tasks.*

BRIAN HARVEY *wins the CS Division's sixth Campus Distinguished Teaching Award.*

RICHARD KARP *wins the National Medal of Science.*

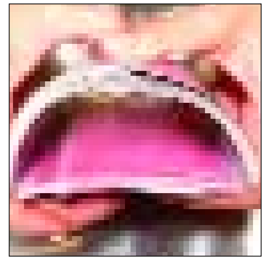
GÖDEL PRIZE FOR OUTSTANDING JOURNAL ARTICLES *in the area of theoretical CS awarded jointly to* ***Mark Jerrum*** *and* ***Alistair Sinclair*** *for "Approximate counting, uniform generation and rapidly mixing Markov chains," Information and Computation, 82 (1989), 93-133, and "Approximating the Permanent," SIAM Journal on Computing, 18 (1989), 1149-1178.*

FORMER CS Ph.D. STUDENT AND MIT FACULTY MEMBER SHAFRIRA GOLDWASSER *wins the ACM Grace Murray Hopper Award.*

PAUL DEBEVEC, CAMILLO TAYLOR AND JITENDRA MALIK DEVELOP THE FACADE SYSTEM *for photorealistic modeling of architecture from photographs. Helps launch the field of image based modeling and rendering in graphics.*

MIXING PROBLEMS




Cool water down to 0 degrees Celsius, and it converts to ice. Heat up a horseshoe magnet, and its magnetism disappears. Cool a normal metal, and it suddenly becomes a semiconductor. At first, such phase transitions may look like a problem for physicists. But Professor Alistair Sinclair belongs to a small band of computer scientists who see a connection between these physical phenomena and computation. Like matter, many computational problems appear to undergo abrupt phase transitions of their own, from 'solvable' to 'insolvable.'

Sinclair has been looking at mixing problems, such questions as how many times you have to shuffle a deck of cards before the order is random. But Sinclair doesn't work with cards—instead, he has focused on mixing problems concerning, for example, shuffling the orientation of atoms in the famous Ising model for ferromagnetism. This mixing problem appears to undergo a phase transition from exponential time to polynomial time at exactly the same temperature that the magnet undergoes a phase transition from order to chaos. "Our goal is to turn this apparent connection—and others like it—into a theorem," says Sinclair, who over the years had developed, with colleagues and students, powerful tools for analyzing mixing problems of many kinds.

The Ising model describes the behavior of a large array of individually magnetized atoms, each one pointing either up or down. Atoms like to take on the orientation of their neighbors, and at low temperatures this interaction is so strong that the magnet as a whole tends to be ordered, with nearly all atoms pointing the same way. At high temperatures, however, entropy wins out, and the individual atoms pick directions more or less randomly. Somewhere in the middle of the temperature scale is an abrupt phase transition.

The Ising model doesn't predict the configuration of ups and downs a given magnet will adopt, but rather gives a probability distribution over all the possible configurations. So the way to study thermodynamic properties of the model, such as entropy and specific heat, is to measure those properties on a smattering of 'typical' configurations sampled from the probability distribution.

But how do you get a typical configuration? To do so, researchers perform a mixing procedure analogous to shuffling a deck: start with a given configuration, possibly far from typical, then pick an atom randomly, and either flip it or leave it alone according to a probability that depends on the orientations of its neighbors and the temperature. This process, Sinclair notes, is not only a reasonable way to shuffle the configuration, but also a plausible model for the dynamics of the actual magnet. The question is, how many flips do you have to do before you reach a typical configuration?

For card shuffling, researchers showed about a decade ago that it takes approximately seven riffle shuffles to mix a deck. For the Ising model, it's more complicated: above the critical temperature, the speed of mixing is thought to be polynomially fast (with respect to the number of magnets), but below that temperature, it is thought to be exponentially slow.

The intuition behind this is fairly straightforward. At high temperatures, any configuration is about as likely as any other, since the orientation of an atom isn't influenced much by its neighbors. So if you start with a configuration where all atoms point down, say, it won't take too many flips to get to a typical state. But at low temperatures, the most typical configurations are the ones dominated either by ups or downs. Each atom exerts a strong influence on its neighbors, so if you start with the all-down configuration, it's extremely unlikely that random flips will allow significant pockets of ups to form. Thus, even though the configurations dominated by ups represent fully half of the likely configurations, it's almost impossible to get to them by mixing the all-down configuration.

So far, researchers have been able to make this heuristic argument rigorous only for two-dimensional Ising models. Sinclair is trying to put the connection between phase transitions and mixing times on a sound theoretical footing in a wide range of other settings.

The link between phase transitions and mixing times opens up an exciting spectrum of possibilities, Sinclair says. "A phase transition is not a priori a computational concept," he says. "But with the discovery that a phase transition has a precise computational manifestation, physicists and computer scientists have realized that they are investigating the same questions from different angles and with different techniques. This promises to have far-reaching consequences in both fields."

TIMELINE

1997

DOUGLAS ENGELBART, *Berkeley Ph.D. 1955, awarded the ACM Turing Award "For an inspiring vision of the future of interactive computing and the invention of key technologies to help realize this vision."*

JOHN OUSTERHOUT WINS THE ACM SOFTWARE SYSTEMS AWARD *for Tcl/Tk "For the Tcl scripting language which allows developers to create complex systems from pre-existing components. The embedded Tk provides a simple mechanism for creating graphical user interfaces. Together they make a powerful addition to the software repertoire."*

WILLIAM KAHAN *gives the SIAM von Neumann Lecture, honoring his contributions to mathematics and its applications.*

"THE CAMPANILE MOVIE," *directed by* ***Paul Debevec****, premieres at the SIGGRAPH electronic theatre marking a major milestone in the creation of photorealistic synthetic imagery of natural scenes.*

THE BRASS PROJECT CULMINATES WITH WAWRZYNEK AND HIS STUDENTS *releasing the design of GARP, an innovative hybrid processor with RISC core and reconfigurable array that emerges as the standard model for reconfigurable computing.*

JITENDRA MALIK'S GROUP DEMONSTRATES AUTOMATED DRIVING *under visual control in the I-5 corridor near San Diego. This is the result of a joint Honda-PATH project.*

MOSS, A SYSTEM FOR DETECTING PLAGIARISM IN SOFTWARE, *is released on the Internet by* ***Alex Aiken*** *and immediately becomes a standard tool for teachers of programming courses world-wide.*

STEVE MCCANNE DEVELOPS *influential multicast-based conferencing software for the Internet: VIC and VAT. His realtime protocol, RTP/RTCP, migrates to Microsoft's conferencing software. McCanne wins the ACM Dissertation Award for "Scalable Compression and Transmission of Internet Multicast Video".*

BARBARA SIMONS *Berkeley Ph.D. 1981, is elected President of the Association of Computing Machinery for the 1998–2000 term.*

DAVID CULLER'S NOW-2 PROJECT *constructs the first cluster processor consisting of more than 100 processors.*

DAVID PATTERSON LEADS THE TERTIARY DISK PROJECT, *which applies commodity disk and processor technology for highly available, large-scale web service. It is used by S.F. Museum of Art to host their on-line accessible art collections*

GAME THEORY AND THE INTERNET

TCP/IP

Christos Papadimitriou's work isn't the first time UC Berkeley researchers have tackled the Internet. Berkeley computer scientists were there right at the beginning, implementing the first open source versions of TCP/IP, the protocols for exchanging information on which the Internet is based.

Berkeley's share in the open source revolution started when Robert Fabry, then a Berkeley professor, brought the Unix operating system from Bell Labs to Berkeley in 1974. Berkeley computer scientists, led by graduate student Bill Joy—who went on to co-found Sun Microsystems—quickly developed many new features, including virtual memory, a Pascal compiler and a visual text editor, that transformed the capabilities of the Unix operating system.

In 1980, Berkeley got a DARPA contract to develop, among other things, a Unix implementation of TCP/IP for the ARPAnet, the Internet's forerunner. Until then, TCP/IP had been implemented only on special hardware. Berkeley's open source distribution allowed users to adapt the protocols to suit their needs. "It was an important part of the idea of sharing the code among a community," Katz says. "The fact that Berkeley Unix code was open and modifiable has been a lasting legacy."

In the late 1980s, Berkeley made a second vital contribution to the viability of the Internet. As the ARPAnet grew, it was getting so congested that it could barely function. Van Jacobson of Lawrence Berkeley Laboratory developed a control loop to slow down TCP/IP when the network got busy, so users wouldn't clog the system. He then collaborated with UC Berkeley researchers to implement his fix on Berkeley Unix. The implementation proved that the new protocols could handle huge traffic increases. "Fixing congestion in the 1980s was essential to enabling the Internet to scale to millions of nodes," Katz says.

To Christos Papadimitriou, the Internet stands apart from the other creations of computer science. No individual has designed it: instead, it is the outgrowth of the busy, competitive jostlings of thousands of individuals and companies. "The Internet is the first computational artifact that we must approach with humility, the same way that physicists approach the Universe, or biologists approach the cell," Papadimitriou says.

Because the Internet is the product of the interactions—sometimes competitive and sometimes cooperative—of many agents, Papadimitriou and Berkeley colleagues Karp and Scott Shenker are investigating it using the tools of game theory, which studies strategic behavior in competitive situations. One of game theory's main conceptual contributions is the idea of a strategic equilibrium, such as the Nash equilibrium, which is a choice of strategies for the players such that no player has an incentive to switch unilaterally to a different strategy. "The Internet is an equilibrium, but what is the game?" Papadimitriou says.

That question is too broad to tackle in its full enormity, but Papadimitriou and other researchers are making headway on some of its facets. One such problem is to understand the cost to society of the anarchic nature of the Internet. For instance, assume that users sending packets through the Internet choose routes to minimize the traffic delays to their own individual packets. How does the total delay incurred by their selfish choices compare to that of the socially optimal routing, the one a benevolent dictator would choose to minimize total delays? In 1998, Papadimitriou and University of Athens Professor Elias Koutsoupias showed that in a particular simple network, anarchy is about 1.5 times slower than socially conscious routing. More recently, researchers have shown that for a broad range of networks, the cost of anarchy is a factor of two. "We're getting results that seem to support the idea that an all-powerful coordinator might make things a little more efficient, but not enough to be worth the trouble," Papadimitriou says.

Other questions abound. Why, for instance, hasn't the Internet teetered out of control, getting more and more congested as it grows? And what is the right way to assign value to personal information, to protect privacy? Theoretical computer scientists can bring valuable insights to these questions, Papadimitriou says. "We've always been obsessed with scalability, and scale is the essence of the Internet," he says. "As the Internet is getting huger and huger, we have the mathematical tools to understand the nature of the beast."

HAPLOTYPING

Professors Richard Karp, Luca Trevisan and Umesh Vazirani (from right) "hold court" at Nefeli's coffee shop in Berkeley with CS students. There, students and faculty discuss and debate current theories, issues and problems encountered in their research.

With the much-heralded sequencing of the human genome, biologists are suddenly awash in data. The cellular blueprints encoded in the genome have revealed the rich repository of information stored in biological systems. But making sense of the data is anything but straightforward. To sift through the boatloads of information, biologists are turning increasingly to theoretical computer scientists intent on applying mathematical models to tease the secrets out of the genetic code.

Knowing the letters of the code doesn't tell you what it does. How the genome's instructions get translated into the complex responses that keep a cell alive in an unpredictable world is still unclear. To Richard Karp, the challenge of figuring out just what is hidden in the genetic code is exhilarating. "In the early 90s I became convinced that applications of theoretical computer science to biology were going to become extremely important," he says. "We have great power now to measure all kinds of events within living cells and make models of how they operate."

One of the questions Karp has tackled is the problem known as 'haplotyping'. Every human being contains two slightly different copies of each gene—one from the mother and one from the father—and every gene is a long string of letters drawn from an alphabet of four nucleotides usually abbreviated A, T, C and G. It is a fairly quick procedure to determine, for each position along the genetic sequence, which two nucleotides an individual has on his two copies of the gene (his genotype). But the two nucleotides don't come with nametags saying which one belongs to the paternal copy and which to the maternal copy. And it is this information—the individual's haplotype—that is most helpful for studying inherited diseases.

Karp, together with Berkeley postdoc Eran Halperin and Eleazar Eskin of Columbia University, wondered if there was a way to use computational tools to pull an individual's haplotype out of his genotype. If you're looking at just one individual, there's no basis for assigning nametags to the two nucleotides at each position. But if you compare the genotypes of a population of individuals, some ways of assigning nametags to the nucleotides become more probable than others.

Different haplotypes emerge as the result of mutations, and mutations tend to be rare. Therefore, the number of haplotypes in a population will tend to be small. This suggests that an assignment of nucleotide nametags that produces a small number of different haplotypes in a population is more likely to be correct than an assignment that produces a large number. Using this and similar principles, Karp's team came up with an algorithm for reconstructing haplotypes from genotypes. In an experimental trial, the team used the algorithm to guess the haplotypes of a collection of individuals, and compared the results with known genetic information about the individuals' parents. They found that the algorithm correctly identified individuals' haplotypes 99 percent of the time.

To Karp, who is attracted to problems with a combinatorial flavor, genomics offers a wide variety of other tantalizing problems. Computer science can help biologists get to the core of such questions quickly, Karp says. "Twenty years ago, people could only study these problems by painstaking biochemical experiments, but now that we have the genome, we can supplement that with combinatorial studies," he says. "The biochemistry can be more focused and efficient, so it will take six months to explain an important gene's regulation, not 30 years."

TIMELINE

1997 CONTINUED

MIKE STONEBRAKER, JOE HELLERSTEIN AND THEIR STUDENTS *form Cohera Corp to commercialize the Mariposa technology. It is eventually acquired by PeopleSoft.*

MICHAEL JORDAN, *an intellectual leader in the field of statistical learning theory from MIT, joins the Berkeley faculty with a joint appointment in CS and Statistics. In 1996 he and colleagues developed a variational approach to large-scale probabilistic inference, forging connections between machine learning, optimization theory, and statistical physics.*

1998

FORBES MAGAZINE NAMES MIKE STONEBRAKER *one of the eight innovators driving the Silicon Valley wealth explosion in its 80th anniversary edition.*

HARI BALAKRISHNAN WINS ACM DISSERTATION AWARD *for "Challenges To Reliable Data Transport Over Heterogeneous Wireless Networks." Currently at MIT.*

JIM GRAY (BERKELEY Ph.D. 1969) AWARDED THE ACM TURING AWARD *"For seminal contributions to database and transaction processing research and technical leadership in system implementation."*

RICHARD KARP *wins the Technion's Harvey Prize.*

DOUG TYGAR, *a leader in security and privacy, joins the Berkeley faculty from CMU. His appointment is joint with the School of Information Management Systems (SIMS), strengthening the ties between technology, policy, law, and economics on the Berkeley campus.*

1999

RON AVNUR (MS '00) AND JOE HELLERSTEIN *publish first paper on Eddies, an adaptive dataflow routing technique overturning 25 years of database query engine design. Eddies form cornerstone of subsequent Telegraph project at Berkeley.*

ERIC BREWER AND JOE HELLERSTEIN *selected as two of the 100 "most remarkable young innovators ever assembled" by Technology Review magazine in their inaugural "TR100" list.*

RANDY KATZ *wins the American Society for Engineering Education Frederick Emmons Terman Award for authoring a leading engineering textbook before his 45th birthday.*

GARTH GIBSON, RANDY KATZ, AND DAVID PATTERSON WIN THE IEEE REYNOLD B. JOHNSON INFORMATION STORAGE AWARD *"for the development of Redundant Arrays of Inexpensive Disks (RAID)."*

PROOF-CARRYING CODE

DOWNLOADING HAS BECOME A NATIONAL PASTIME. MUSIC, SOFTWARE UPDATES AND THE NEWEST FLASHY GAMES FLOW STEADILY FROM THE INTERNET TO OUR COMPUTERS. MOST PEOPLE SHRUG OFF THE RISK OF BOOBY-TRAPPED OR FAULTY CODE—BUT SHOULD THEY? EVEN TRUSTWORTHY SOURCES CAN ISSUE SOFTWARE UPDATES WITH BUGS THAT BRING DOWN A COMPUTER. WORSE YET, HACKERS MIGHT TAMPER WITH SOFTWARE, OR IT COULD GET CORRUPTED IN TRANSIT. AND WHAT ABOUT CODE FROM THE SHADIER REGIONS OF THE INTERNET—FROM PROGRAMMERS WHO ARE NOT SO RELIABLE?

George Necula is figuring out how to let people download and use untrusted code without fear. Necula envisions a system in which manufacturers prove to users that their code does what they claim—that it won't start digging into private files, for example, or mail copies of itself over the Internet.

At first, it might seem that any explanation of how the code works would be huge—many, many times the size of the code itself. This was indeed the case in early implementations of Necula's "Proof-carrying code" (PCC), which he developed as a graduate student with his advisor Peter Lee at Carnegie Mellon University. On average, the proofs were three times larger than the software they explained. "People were joking, you don't have proof-carrying code—you have code-carrying proof!" Necula says.

Since coming to Berkeley, Necula has revamped his approach to writing proofs and, amazingly, whittled down the resulting explanations to about 20% the size of the software. In the early versions of PCC, the proof wrote out, in gory detail, all the safety conditions that had to be tested; 'proof-checking' software on the user's end then verified the code satisfied these conditions. Necula has since hit upon a more concise approach, in which the proof helps the user's proof-checker over only the most challenging hurdles.

The shortened proof is like a guide through a labyrinth, Necula says. Finding a path through a maze can be hard even with a bird's eye view, he observes, but "if someone shows you the path to take, it's very easy to check that the path is valid and will get you through."

In a similar way, the proof helps the user's proof-checker navigate the code. The proof-checker runs through the software on its own steam, looking for instructions that might be problematic—for example, places where the software reads or writes to memory, or directs the operating system to create a file. Next to each of these potentially dangerous commands, the software writer has placed a simple description of what needs to be checked to establish the security of the command. For a command that writes to memory, for example, the proof would direct the proof-checker to check the addresses to which the software is writing and verify that they are within acceptable bounds. Once the checker is satisfied, it goes on until it hits the next problematic instruction in the code.

Proof-checking code is only the first step on the road to Necula's ultimate goal—an open virtual machine that sits on a user's hard-drive and can check, verify, and run code sent to it in any language. Currently, software sent to a virtual machine must be written in Java or Microsoft Common Launguage Infrastructure. On the open virtual machine Necula envisions, users would first upload a verifier tailored to a specific language. The machine would then be able to accept and verify code in that language.

The challenge, Necula says, is to figure out how to verify the verifier. "How can I have a guarantee that the verification done with the verifier you just sent me is correct?" he says. "Maybe you sent me a verifier that always says yes, yes, yes—a verifier that always says, 'This is all right.' How can I prevent that from happening? That's the next thing," Necula says. "That's what we're trying to develop right now."

YOUNG FACULTY

Like many of Berkeley's young faculty, George Necula came to Berkeley directly from graduate school. Originally leaning toward a career in industry, he was lured into academia by the high caliber of his future colleagues and the Berkeley atmosphere. "It's a very friendly place," he says. "The computer science division makes a point of hiring only as many people as it hopes to tenure," says Division Chair Jitendra Malik.

"This makes for a welcoming environment and encourages collaboration," Necula says. "Here, you don't feel as if you're competing against the others for limited resources or tenure spots," he says. "We're all competing with the outside world, not with each other."

TIMELINE

1999 CONTINUED

FORMER BERKELEY CS Ph.D. STUDENT *and Illinois faculty member Wen-mei Hwu wins the ACM Grace Murray Hopper Award.*

IN THE DECADE OF THE 1990S, *the CS Division rebounds from the large number of retirements in the early 1990s by hiring 21 new faculty members.*

2000

LUCA TREVISAN *wins the Oberwolfach Prize.*

DAVID PATTERSON WINS IEEE JOHN VON NEUMANN MEDAL *(with Hennessy) and the* ***IEEE Mulligan Education Medal****, becoming the first person to win two IEEE medals in a single year.*

RANDY KATZ WINS THE ACM KARL V. KARLSTROM OUTSTANDING EDUCATOR AWARD.

WILLIAM KAHAN WINS THE IEEE EMANUEL R. PIORE AWARD *for his outstanding contributions to the field of information processing.*

LOFTI ZADEH WINS THE ACM ALLEN NEWELL AWARD *for his contributions to artificial intelligence.*

JOHN KUBIATOWICZ LAUNCHES OCEANSTORE, *an ambitious reliable distributed peer-to-peer network storage system.*

FORMER CS Ph.D. STUDENT NARENDRA KARMARKAR *wins the ACM Paris Kanellakis Theory Award for his development of a highly efficient algorithm for linear programming.*

THE TELEGRAPH PROJECT, *directed by* ***Mike Franklin*** *and* ***Joe Hellerstein****, provides adaptive continuous queries integrating multiple streaming and networked data sources. Telegraph's Y2K Presidential Election web service performs live correlation of political donor data with publicly available financial and demographic sources.*

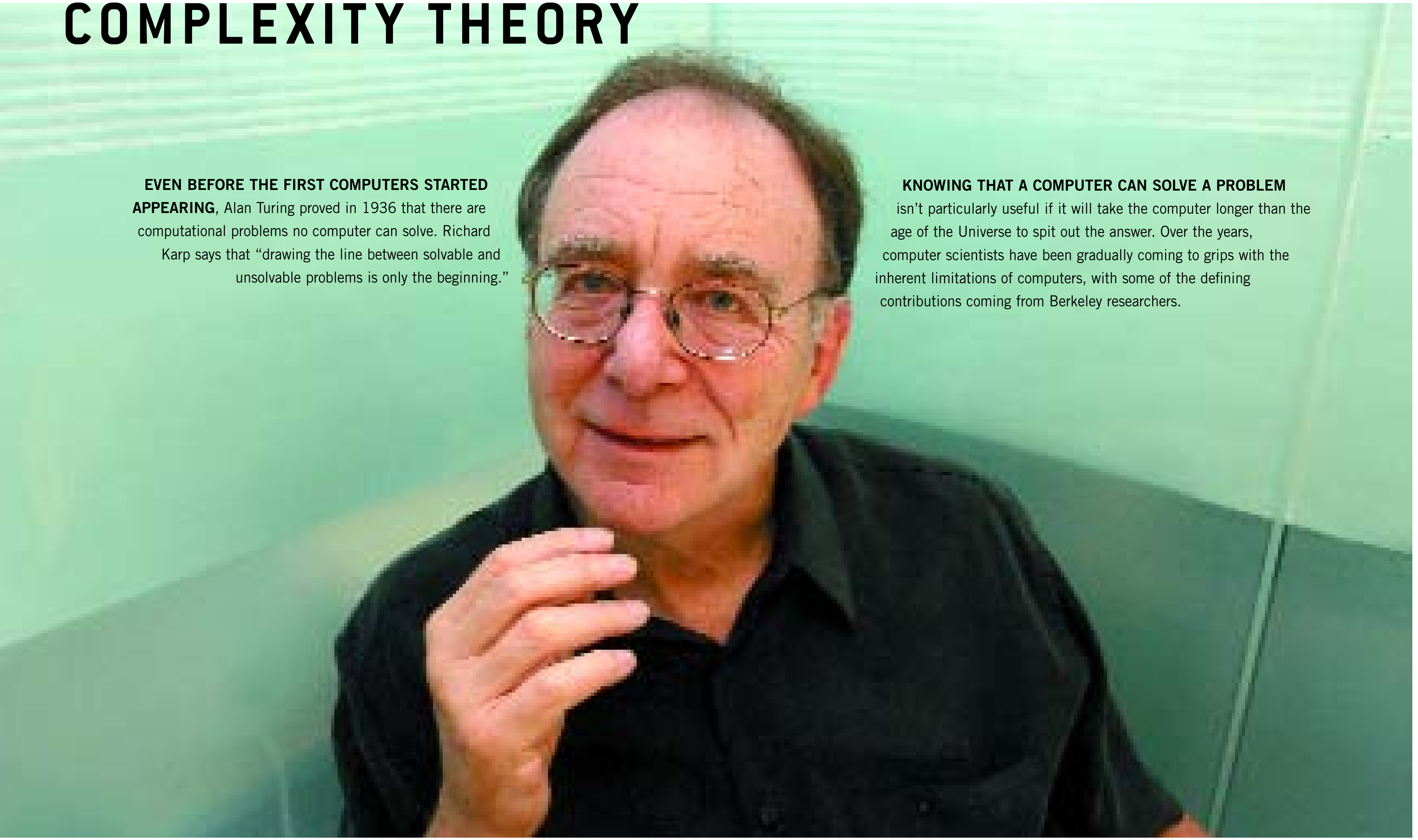
JOHN KUBIATOWICZ WINS AN NSF PECASE AWARD, *the highest award for young academic scientists and engineers recognizing their future career promise.*

2001

ION STOICA WINS ACM DISSERTATION AWARD *for his work at CMU on "Stateless Core: A Scalable Approach for Quality of Service."*

DAVID WAGNER WINS ACM DISSERTATION AWARD *for "Static Analysis and Computer Security."*

COMPLEXITY THEORY



EVEN BEFORE THE FIRST COMPUTERS STARTED APPEARING, Alan Turing proved in 1936 that there are computational problems no computer can solve. Richard Karp says that "drawing the line between solvable and unsolvable problems is only the beginning."

KNOWING THAT A COMPUTER CAN SOLVE A PROBLEM isn't particularly useful if it will take the computer longer than the age of the Universe to spit out the answer. Over the years, computer scientists have been gradually coming to grips with the inherent limitations of computers, with some of the defining contributions coming from Berkeley researchers.

BY THE MID 1960S, computer scientists were reaching a consensus about what makes a problem tractable: it should be solvable in an amount of time that grows only polynomially with the size of the input (such a problem is said to lie in the class P). Although many problems were shown to be in P, many other natural problems appeared to lie in a broader class called NP. A given problem in NP might, or might not, be solvable in polynomial time, but given a potential solution, it's possible to check in polynomial time whether that solution is valid. All problems in P are automatically in NP, but not vice versa, scientists believe: the hardest problems in NP are thought to be impossible to solve in polynomial time.

At first, it wasn't clear whether there was any property that linked the hard problems in NP. But in 1971, Stephen Cook of the University of Toronto proved that a problem called SAT was "NP-complete": if it could be solved in polynomial time, then so could every hard problem in NP (something computer scientists believe is very unlikely). At about the same time, Leonid Levin of Boston University (at the time based in the Soviet Union) proved the same result independently.

On reading Cook's paper, Berkeley Professor Richard Karp quickly realized that a host of important problems have the same universality property as SAT. Karp's work in operations research—on puzzles such as the traveling salesman problem and the design of digital circuits—had given him an instinct for what made particular problems hard. "My work had revolved around the design of algorithms, so I was very familiar with these combinatorial explosions, and I was pretty cognizant of the quintessential examples that people should look at," says Karp, who in 1985 was awarded the ACM's A.M. Turing Award, one of computer science's highest honors, chiefly for his work on NP-completeness. "I think it was important to have one foot in the theory camp and one foot in the applied camp."

In 1972, Karp proved that 21 fundamental problems were NP-complete. Other computer scientists quickly jumped on the bandwagon, using Karp's methods to show that thousands of problems are NP-complete. "Karp was in some sense the St. Paul who popularized the concept and gave it to the masses," Papadimitriou says. The notion of NP-completeness appears to capture a remarkable dichotomy between hard and easy problems: nearly every natural problem is either NP-complete or in P. "This was one of those lucky moments in science when a particular theoretical concept had very wide applicability and could help classify practically everything," Papadimitriou adds.

Fleshing out the concept of NP-completeness made computer scientists realize that for many questions, it is unrealistic to expect a computer to come up with an exact answer efficiently. Researchers wondered whether they would fare better trying to find approximate answers. Amazingly, in the early 1990s, Berkeley graduate students Madhu Sudan and Sanjeev Arora showed, together with several other theoretical computer scientists, that for some problems, finding even a reasonably good approximation is as hard as solving NP-complete problems exactly. The researchers' work landed in the theoretical computer science community like "a huge rock in the middle of the lake," recalls Papadimitriou. "There were waves everywhere." Sudan and Arora were each awarded the ACM Doctoral Dissertation Award, the most prestigious award for a finishing Ph.D. student.

It might seem that by uncovering so many problems computers can't handle, the results of Karp, Sudan, Arora and others have been depressingly negative. But the proof that a problem is intractable often points researchers away from fruitless endeavors towards more promising directions. "Without NP-completeness, maybe we'd have people wasting time trying to color graphs now, for instance," Papadimitriou says. "Showing something is NP-complete isn't a way to kill problems, but to create the right problems." If computer scientists have had to lower their sights about what computers can do, the resulting vision has been no less rich.

< *Professor Karp striking a characteristic pose. Karp was Berkeley's first chair of the CS Division, and he continues to innovate in the field of theoretical computer science thirty years later.*

TIMELINE

2001 CONTINUED

GEORGE NECULA WINS THE ACM GRACE MURRAY HOPPER AWARD *for his CMU thesis work on Proof Carrying Code.*

CHRISTOS PAPADIMITRIOU *is recognized as the Most Eminent Greek Computer Scientist by the Greek Information Technology and Computer Science Societies.*

THE CENTER FOR INFORMATION TECHNOLOGY RESEARCH IN THE INTEREST OF SOCIETY (CITRIS) *is established as a California Science and Technology Center, involving the Berkeley, Davis, and Santa Cruz campuses.* ***Ruzena Bajcsy****, from the University of Pennsylvania and the National Science Foundation, joins the faculty as Institute Director, with* ***James Demmel*** *serving as Chief Scientist.*

2002

CHRISTOS PAPADIMITRIOU WINS THE KNUTH PRIZE.

DAVID WAGNER *is selected by Popular Science Magazine as one of its "Brilliant 10".*

FORMER GRADUATE STUDENT MADHU SUDAN *wins the Nevanlinna Prize, honoring mathematicians under 40, for his work on the concept of a mathematical proof.*

DAVID PATTERSON'S VIRAM PROJECT *tapes out a 125M transistor microprocessor with vector multimedia support and 13 Megabytes of memory on a single chip.*

G. MORI AND JITENDRA MALIK DEMONSTRATE FOR THE FIRST TIME *a computer vision program that defeats the E-Z GIMPY Captcha. These are used at Yahoo and elsewhere as Turing tests to distinguish computers and humans.*

DOUG TYGAR AND ADRIAN PERRIG DEVELOP MICRO-TESLA—*the world's first system for strong authentication of wireless nodes. Tygar's student* ***Dawn Song*** *releases Athena, which offers orders of magnitude improvements on the best previous authentication methods. Together, these offer the first comprehensive method for generating mobile secure protocols.*

JOHN KUBIATOWICZ SELECTED *as one of the 50 Leading Young Scientists by Scientific American.*

WORKSHOP ON "THEORY OF COMPUTATION AND THE SCIENCES" AT BERKELEY.

QUANTUM COMPUTATION SEMESTER AT MATHEMATICAL SCIENCES RESEARCH INSTITUTE (MSRI)

UC BERKELEY COMPUTER SCIENCE FACULTY 2003–2004

1. Randy H. Katz
2. James Demmel
3. Brian A. Barsky
4. Katherine A. Yelick
5. John Wawrzynek
6. Jennifer Mankoff
7. Michael A. Harrison
8. David Patterson
9. James O'Brien
10. Joseph M. Hellerstein
11. Carlo H. Séquin
12. Dan Garcia
13. Jerome A. Feldman
14. Jitendra Malik
15. David E. Culler
16. Christos Papadimitriou
17. Scott Shenker
18. Michael H. Clancy
19. Ken Goldberg
20. George Necula
21. Luca Trevisan
22. Satish Rao
23. Richard J. Fateman
24. David Forsyth
25. Paul N. Hilfinger
26. Peter Bartlett
27. David Wagner
28. Ras Bodik
29. Ion Stoica
30. Robert Wilensky
31. Tom Henzinger
32. Ruzena Bajcsy
33. Eric Brewer
34. Umesh Vazirani
35. Alan J. Smith
36. William M. Kahan (center of picture)

Left to Right: John F. Canny, Michael Franklin, Susan L. Graham, Brian K. Harvey, Michael Jordan, Anthony D. Joseph, Richard M. Karp, John Kubiatowicz, Gene Myers, Jonathan Shewchuk, Alistair Sinclair, Doug Tygar.

TIMELINE

2003

LEN ADLEMAN, A STUDENT OF MANUEL BLUM, *shares the Turing Award with* ***Ronald Rivest*** *and* ***Adi Shamir*** *for their work on encryption.*

JONATHAN SHEWCHUK WINS THE WILKINSON PRIZE *for his outstanding contributions to the field of numerical software and computational science.*

EUGENE MYERS JOINS THE CS DIVISION *from his position as Chief Computer Scientist at Celera Genomics. He developed the algorithmic breakthroughs that enabled the compilation of the human genome. He won the ACM Paris Kanellakis Theory Award in 2001 for this work.*

CHRISTOS PAPADIMITRIOU *publishes* **Turing** *(a novel about Computation).*

PETER BARTLETT JOINS THE CS DIVISION *with a joint appointment with Statistics. In 1997 he developed the first satisfactory theoretical explanation of how neural networks learn.*

NINE NEW FACULTY MEMBERS HAVE JOINED THE CS DIVISION *thus far in the decade of the 00s. The size of the CS Division faculty reaches 43.49 FTE, the largest since the Division was founded in 1973.*

"HE WHO RECEIVES AN IDEA FROM ME, RECEIVES INSTRUCTION HIMSELF WITHOUT LESSENING MINE; AS HE WHO LIGHTS HIS TAPER AT MINE, RECEIVES LIGHT WITHOUT DARKENING ME."

–Thomas Jefferson, on the importance of sharing knowledge

THIS BOOK IS DEDICATED TO *the many individuals and their collective successes—too numerous to mention—that have been affiliated with the Computer Science Division at Berkeley over the last thirty years. We could not have built this outstanding educational and research computer science institution without their contributions.*

EDITOR-IN-CHIEF: Randy Katz

EDITORIAL BOARD: Alex Aiken, James Landay, Jitendra Malik, James O'Brien, David Patterson, Stuart Russell, Alistair Sinclair

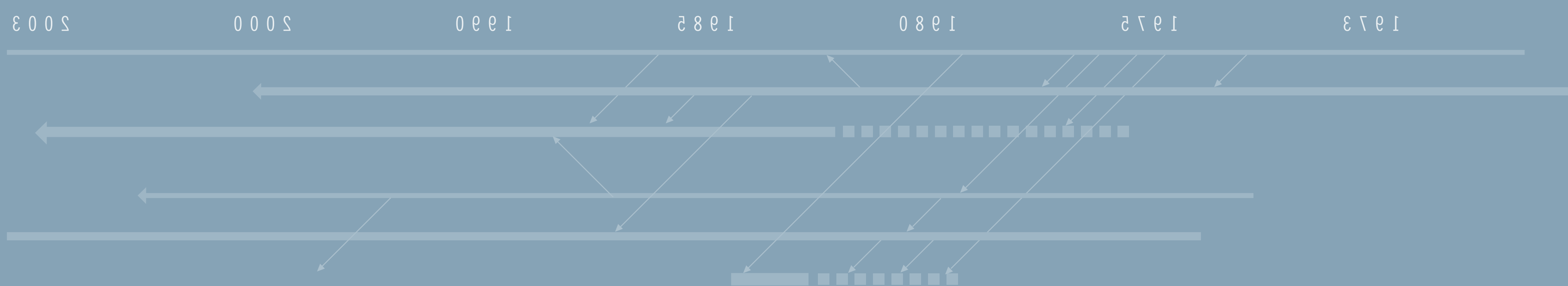
EDITORIAL ASSISTANT: Kate Riley

EDITORIAL ASSOCIATE: Debra Zaller

SCIENCE WRITERS: Katie Greene, Erica Klarreich

PHOTOGRAPHER: Peg Skorpinski

BOOK DESIGN: Julie Rozelle, Shelby Designs & Illustrates


2003
2000
1990
1985
1980
1975
1973