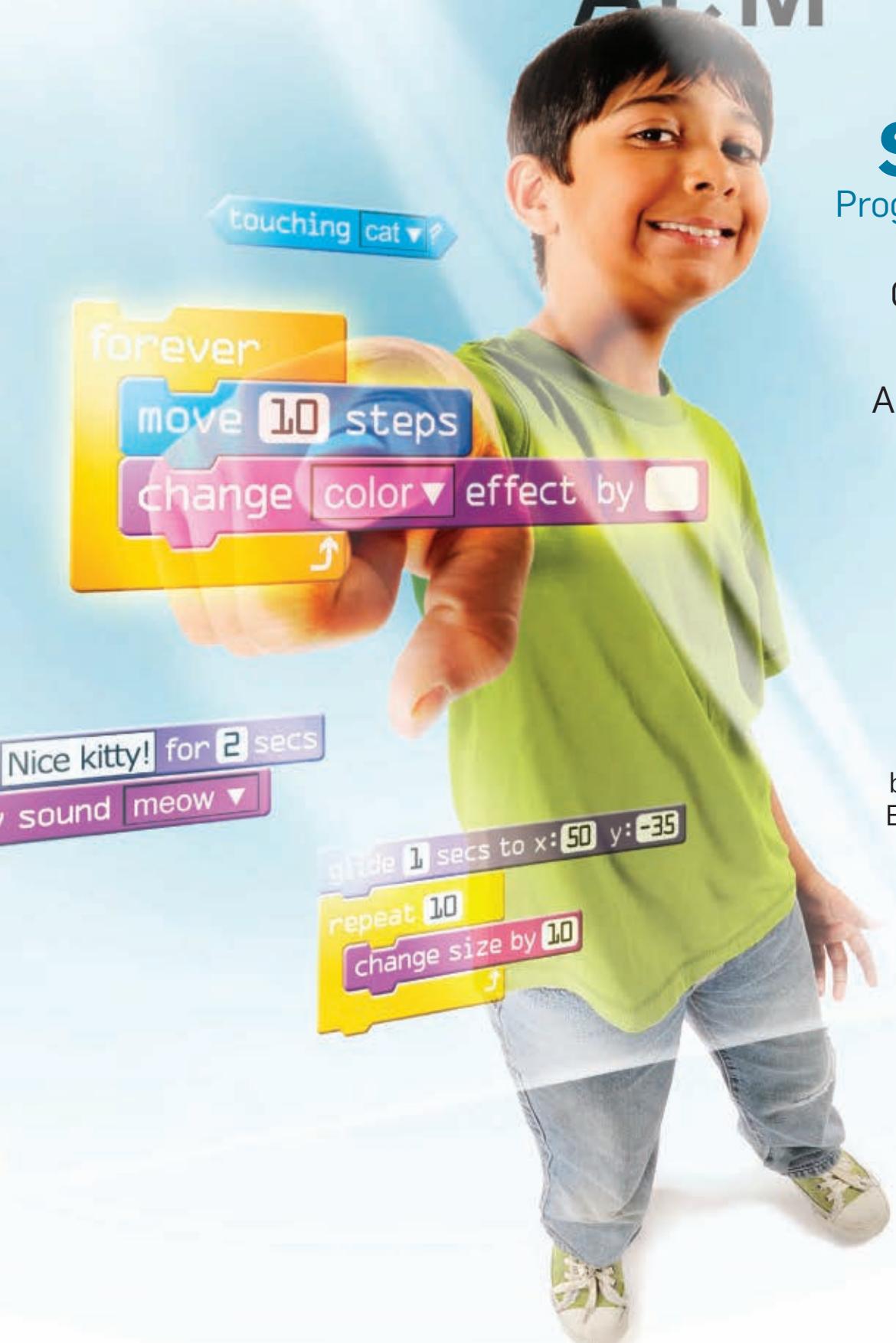


COMMUNICATIONS OF THE ACM

CACM.ACM.ORG

11/09 VOL.52 NO.11



Scratch Programming for All

Communications
Surveillance

An Interview with
Ping Fu

Usable Security:
How To Get It

E-Paper's
Next Chapter

Turing Lecture
by Edmund M. Clarke,
E. Allen Emerson, and
Joseph Sifakis

Association for
Computing Machinery





CALL FOR PAPERS

<http://fdg2010.org>

the International Conference on the
Foundations of Digital Games

June 19 - 21 2010

(cc) (by =) Veronica Vale
www.flickr.com/people/vbv/



(cc) (by =) Steve Schwartz
www.flickr.com/people/vasculata

Submission Deadlines

Papers & Posters: 5th Feb

Doctoral Consortium: 12th Feb

Demos: 2nd April

*Asilomar Conference Grounds
Monterey, California, USA.*

Learning in Games

*Infrastructure (Databases,
Networks, Security)*

Artificial Intelligence

Graphics & Interfaces

*Computer Science &
Games Education*

Game Design

Game Studies

Conference Chair:
Program Chair:
Doctoral Consortium Chair:
Workshops Chair:
Panels Chair:
Tutorials Chair:
Industrial Relations Chair:
Local Arrangements Chair:
Webmaster:

Ian Horswill, Northwestern University
Yusuf Pisan, University of Technology, Sydney
Zoran Popovic, University of Washington
Michael Mateas, University of California, Santa Cruz
Ian Bogost, Georgia Institute of Technology
Robin Hunnicke, That Game Company
Hiroko Osaka, Northwestern University
Marilyn Walker, University of California, Santa Cruz
Karl Cheng-Heng Fua, Northwestern University

An Official Conference of the
**Society for the Advancement of
the Science of Digital Games**

SASDG

FDG 2010 Supported by:

Microsoft®

Research

Noteworthy Titles



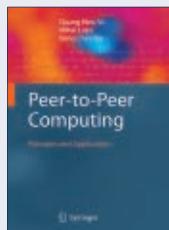
Security for Web Services and Service-Oriented Architectures

Bertino, E.; Martino, L.; Paci, F.; Squicciarini, A.

First book to cover research and existing or

upcoming standards as well as platform-dependent functionalities. A comprehensive guide to security for Web services and SOA. Covers all relevant standards such as XML Encryption, WS-Security, SAML, XACML, and related others, and puts them into a conceptual framework. Introduces a reference framework for future research and developments along security dimensions such as integrity, confidentiality, and availability.

2009. Approx. 250 p. Hardcover
ISBN 978-3-540-87741-7 ► \$69.95



Peer-to-Peer Computing

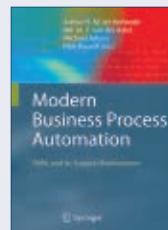
Principles and Applications

Vu, Quang Hieu; Lupu, Mihai; Ooi, Beng Chin

In this book, Vu, Lupu

and Ooi present the technical challenges offered by P2P systems, and the means that have been proposed to address them. They provide a thorough and comprehensive review of recent advances on routing and discovery methods; load balancing and replication techniques; security, accountability and anonymity, as well as trust and reputation schemes; programming models and P2P systems and projects.

2010. X, 300 p. Hardcover
ISBN 978-3-642-03513-5 ► \$89.85



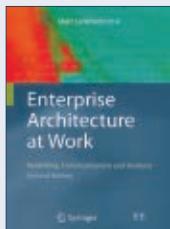
Modern Business Process Automation

YAWL and its Support Environment

ter Hofstede, A. H. M.; van der Aalst, W. M. P.; Adams, M.; Russell, N. (Eds.)

This book provides a comprehensive treatment of the field of Business Process Management (BPM) with a focus on Business Process Automation. It achieves this by covering a wide range of topics, both introductory and advanced, illustrated through and grounded in the YAWL (Yet Another Workflow Language) language and corresponding open-source support environment.

2010. IV, 664 p. 321 illus. Hardcover
ISBN 978-3-642-03120-5 ► \$99.00



Enterprise Architecture at Work

Modelling, Communication and Analysis

Lankhorst, Marc

Introduces the ArchiMate® modeling language for enterprise architecture, an Open Group standard. Describes quantitative analysis methods to assess the impact of architectural changes. Based on industry standards like IEEE 1471, Zachman, UML, and TOGAF and Designed by leading researchers and tested in large-scale industry projects with Extensive industry support.

2nd ed. 2009. XX, 352 p. 164 illus.
(The Enterprise Engineering Series) Hardcover
ISBN 978-3-642-01309-6 ► \$69.95



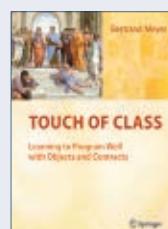
The Social Semantic Web

Breslin, John G.; Passant, Alexandre; Decker, Stefan

Intended for computer science professionals, researchers, and

graduates interested in understanding the technologies and research issues involved in applying Semantic Web technologies to social software. Practitioners and developers interested in applications such as blogs, social networks or wikis will also learn about methods for increasing the levels of automation in these forms of Web communication.

2009. Approx. 290 p. Hardcover
ISBN 978-3-642-01171-9 ► \$69.95



Touch of Class

Learning to Program Well with Objects and Contracts
Meyer, Bertrand

Unique to *Touch of Class* is a combination of a practical, hands-on approach to programming with the introduction of sound theoretical support focused on helping students learn the construction of high quality software. The use of full color brings exciting programming concepts to life. First introductory programming book to use concepts of Design by Contract for software safety and reliability.

2009. LXIV, 876 p. Hardcover
ISBN 978-3-540-92144-8 ► \$79.95



Easy Ways to Order for the Americas ▶ Write: Springer Order Department, PO Box 2485, Secaucus, NJ 07096-2485, USA ▶ **Call: (toll free)** 1-800-SPRINGER

▶ **Fax:** 1-201-348-4505 ▶ **Email:** orders-ny@springer.com or **for outside the Americas ▶ Write:** Springer Customer Service Center GmbH, Haberstrasse 7, 69126 Heidelberg, Germany ▶ **Call:** +49 (0) 6221-345-4301 ▶ **Fax:** +49 (0) 6221-345-4229 ▶ **Email:** orders-hd-individuals@springer.com

▶ Prices are subject to change without notice. All prices are net prices.

014434x

COMMUNICATIONS OF THE ACM

Departments

- 5 **Editor's Letter**
Is The Image Crisis Over?
By Moshe Y. Vardi

- 6 **Letters To The Editor**
Pay for Editorial Independence

- 9 **In the Virtual Extension**

- 10 **blog@ACM**
Computer Science Curriculum, Deceptive Advertising
Ramana Rao writes about the evolution of computer science curriculum and Greg Linden reflects on ethics and advertising.

- 12 **CACM Online**
Internet Addiction: It's Spreading, but is it Real?
By David Roman

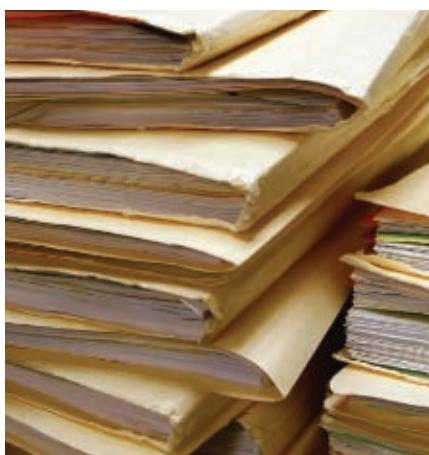
- 33 **Calendar**

- 106 **Careers**

Last Byte

- 112 **Puzzled**
Covering the Plane
By Peter Winkler

News



- 13 **Deep Data Dives**
Discover Natural Laws
Computer scientists have found a way to bootstrap science, using evolutionary computation to find fundamental meaning in massive amounts of raw data.
By Gary Anthes

- 15 **Electronic Paper's Next Chapter**
The technological challenge for researchers working on the next generation of electronic paper is to render color as brightly as traditional paper, without increasing power requirements or end-user costs.
By Kirk L. Kroeker

- 18 **Implementing Electronic Medical Records**
Despite a number of challenges, patients' medical records are slowly making the transition to the digital age.
By Leah Hoffmann

- 21 **Exploring New Frontiers**
The Expeditions in Computing program provides scientists with the funding to work on ambitious, often multi-disciplinary research.
By Gregory Goth

Viewpoints

- 25 **Privacy and Security**
Usable Security: How to Get It
Why does your computer bother you so much about security, but still isn't secure? It's because users don't have a model for security, or a simple way to keep important things safe.
By Butler Lampson

- 28 **Legally Speaking**
Are Business Methods Patentable?
How the U.S. Supreme Court's forthcoming decision in the *Bilski v. Doll* case is expected to affect existing and future software patents.
By Pamela Samuelson

- 31 **Economic and Business Dimensions**
The Broadband Price is Not Right
Developing an effective pricing index is essential to understanding the value of broadband connectivity.
By Shane Greenstein

- 34 **Viewpoint**
On Public Service and Computer Science
Members of the computer science community should become more involved in public service by becoming program managers at federal agencies, the opportunities and benefits of which are outlined here.
By Jonathan M. Smith

- 36 **Interview**
An Interview with Ping Fu
Ping Fu, CEO of the digital shape sampling and processing company Geomagic, discusses her background, achievements, and challenges managing a company during a period of dynamic growth.
By Bob Cramblitt



Association for Computing Machinery
Advancing Computing as a Science & Profession

Practice

- 42 **Communications Surveillance: Privacy and Security at Risk**
As the sophistication of wiretapping technology grows, so too do the risks it poses to our privacy and security.
By Whitfield Diffie and Susan Landau
-
- 48 **Four Billion Little Brothers? Privacy, mobile phones, and ubiquitous data collection**
Participatory sensing technologies could improve our lives and our communities, but at what cost to our privacy?
By Katie Shilton
-
- 54 **You Don't Know Jack about Software Maintenance**
Long considered an afterthought, software maintenance is easiest and most effective when built into a system from the ground up.
By Paul Stachour and David Collier-Brown

 Article development led by **ACM Queue**
queue.acm.org Review Articles

Contributed Articles

- 60 **Scratch: Programming for All**
“Digital fluency” should mean designing, creating, and remixing, not just browsing, chatting, and interacting.
By Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, and Yasmin Kafai
-
- 68 **Why IT Managers Don't Go for Cyber-Insurance Products**
Proposed contracts tend to be overpriced because insurers are unable to anticipate customers' secondary losses.
By Tridib Bandyopadhyay, Vijay S. Mookerjee, and Ram C. Rao

Review Articles

- 74 **Turing Lecture**
Turing Lecture from the winners of the 2007 ACM A.M. Turing Award:
Edward M. Clarke, E. Allen Emerson, and Joseph Sifakis.
-
- 86 **Technical Perspective**
Narrowing the Semantic Gap In Distributed Programming
By Peter Druschel
-
- 87 **Declarative Networking**
By Boon Thau Loo, Tyson Condie, Minos Garofalakis, David E. Gay, Joseph M. Hellerstein, Petros Maniatis, Raghu Ramakrishnan, Timothy Roscoe, and Ion Stoica
-
- 96 **Technical Perspective**
Machine Learning for Complex Predictions
By John Shawe-Taylor

- 97 **Predicting Structured Objects with Support Vector Machines**
By Thorsten Joachims, Thomas Hofmann, Yisong Yue, and Chun-Nam Yu



About the Cover:
As if they were assembling Lego bricks, children snap together Scratch graphical programming blocks—shaped to fit together only in ways that make syntactic sense—to create their own programs, playfully explored in the cover story beginning on page 60.

Virtual Extension

As with all magazines, page limitations often prevent the publication of articles that might otherwise be included in the print edition. To ensure timely publication, ACM created *Communications'* Virtual Extension (VE).

VE articles undergo the same rigorous review process as those in the print edition and are accepted for publication on their merit. These articles are now available to ACM members in the Digital Library.

Offshoring and the New World Order
Rudy Hirschheim

If Your Pearls of Wisdom
Fall in a Forest...
Ralph Westfall

Quantifying the Benefits of Investing in Information Security
Lara Khansa and Divakaran Liginla

iCare Home Portal: An Extended Model of Quality Aging E-Services
Wei-Lun Chang, Soe-Tsyer, and Eldon Y. Li

Computing Journals and their Emerging Roles in Knowledge Exchange
Aakash Taneja, Anil Singh, and M.K. Raja

And What Can Context Do For Data?
C. Bolchini, C.A. Curino, G. Orsi, E. Quintarelli, R. Rossato, F.A. Schrieber, and L. Tanca

Why Web Sites Are Lost (and How They're Sometimes Found)
Frank McCown, Catherine C. Marshall, and Michael L. Nelson

Technical Opinion
Steering Self-Learning Distance Algorithms
Frank Nielsen



COMMUNICATIONS OF THE ACM

Trusted insights for computing's leading professionals.

Communications of the ACM is the leading monthly print and online magazine for the computing and information technology fields. *Communications* is recognized as the most trusted and knowledgeable source of industry information for today's computing professional. *Communications* brings its readership in-depth coverage of emerging areas of computer science, new trends in information technology, and practical applications. Industry leaders use *Communications* as a platform to present and debate various technology implications, public policies, engineering challenges, and market trends. The prestige and unmatched reputation that *Communications of the ACM* enjoys today is built upon a 50-year commitment to high-quality editorial content and a steadfast dedication to advancing the arts, sciences, and applications of information technology.

ACM, the world's largest educational and scientific computing society, delivers resources that advance computing as a science and profession. ACM provides the computing field's premier Digital Library and serves its members and the computing profession with leading-edge publications, conferences, and career resources.

Executive Director and CEO
John White
Deputy Executive Director and COO
Patricia Ryan
Director, Office of Information Systems
Wayne Graves
Director, Office of Financial Services
Russell Harris
Director, Office of Membership
Lillian Israel
Director, Office of SIG Services
Donna Cappo

ACM COUNCIL

President
Wendy Hall
Vice-President
Alain Chesaïnais
Secretary/Treasurer
Barbara Ryder
Past President
Stuart I. Feldman
Chair, SGB Board
Alexander Wolf
Co-Chairs, Publications Board
Ronald Boisvert, Holly Rushmeier
Members-at-Large
Carlo Ghezzi;
Anthony Joseph;
Mathai Joseph;
Kelly Lyons;
Bruce Maggs;
Mary Lou Sofza;
Fei-Yue Wang
SGB Council Representatives
Joseph A. Konstan;
Robert A. Walker;
Jack Davidson

PUBLICATIONS BOARD

Co-Chairs
Ronald F. Boisvert and Holly Rushmeier
Board Members
Gul Agha; Michel Beaudouin-Lafon;
Jack Davidson; Nikil Dutt; Carol Hutchins;
Ee-Peng Lim; M. Tamer Ozsu; Vincent
Shen; Mary Lou Sofza; Ricardo Baeza-Yates

ACM U.S. Public Policy Office

Cameron Wilson, Director
1100 Seventeenth St., NW, Suite 50
Washington, DC 20036 USA
T (202) 659-9711; F (202) 667-1066

Computer Science Teachers Association

Chris Stephenson
Executive Director
2 Penn Plaza, Suite 701
New York, NY 10121-0701 USA
T (800) 401-1799; F (541) 687-1840

Association for Computing Machinery (ACM)

2 Penn Plaza, Suite 701
New York, NY 10121-0701 USA
T (212) 869-7440; F (212) 869-0481

STAFF

GROUP PUBLISHER

Scott E. Delman
publisher@acm.acm.org

Executive Editor

Diane Crawford

Managing Editor

Thomas E. Lambert

Senior Editor

Andrew Rosenblom

Senior Editor/News

Jack Rosenberger

Web Editor

David Roman

Editorial Assistant

Zarina Strakhan

Rights and Permissions

Deborah Cotton

Art Director

Andrij Borys

Associate Art Director

Alicia Kubista

Assistant Art Director

Mia Angelica Balaquit

Production Manager

Lynn D'Addesio

Director of Media Sales

Jennifer Ruzicka

Marketing & Communications Manager

Brian Hebert

Public Relations Coordinator

Virginia Gold

Publications Assistant

Emily Eng

Columnists

Alok Aggarwal; Phillip G. Armour;
Martin Campbell-Kelly;
Michael Cusumano; Peter J. Denning;
Shane Greenstein; Mark Guzdial;
Peter Harsha; Leah Hoffmann;
Mari Sako; Pamela Samuelson;
Gene Spafford; Cameron Wilson

CONTACT POINTS

Copyright permission

permissions@acm.acm.org

Calendar items

calendar@acm.acm.org

Change of address

acmcoa@acm.acm.org

Letters to the Editor

letters@acm.acm.org

WEB SITE

<http://acm.acm.org>

AUTHOR GUIDELINES

<http://acm.acm.org/guidelines>

ADVERTISING

ACM ADVERTISING DEPARTMENT

2 Penn Plaza, Suite 701, New York, NY
10121-0701
T (212) 869-7440
F (212) 869-0481

Director of Media Sales

Jennifer Ruzicka
jen.ruzicka@hq.acm.org

Media Kit

acmmediasales@acm.org

EDITORIAL BOARD

EDITOR-IN-CHIEF

Moshe Y. Vardi
eic@acm.acm.org

NEWS

Co-chairs

Marc Najork and Prabhakar Raghavan

Board Members

Brian Bershad; Hsiao-Wuen Hon;
Mei Kobayashi; Rajeev Rastogi;
Jeanette Wing

VIEWPOINTS

Co-chairs

Susanne E. Hambrusch; John Leslie King;
J. Strother Moore

Board Members

P. Anandan; William Aspray;
Stefan Bechtold; Judith Bishop;
Stuart I. Feldman; Peter Freeman;
Seymour Goodman; Shane Greenstein;
Mark Guzdial; Richard Heeks;
Rachelle Hollander; Richard Ladner;
Susan Landau; Carlos Jose Pereira de Lucena;
Helen Nissenbaum; Beng Chin Ooi;
Loren Terveen

PRACTICE

Chair

Stephen Bourne

Board Members

Eric Allman; Charles Beeler; David J. Brown;
Bryan Cantrill; Terry Coatta;
Mark Compton; Benjamin Fried;
Pat Hanrahan; Marshall Kirk McKusick;
George Neville-Neil

The Practice section of the ACM Editorial Board also serves as the Editorial Board of *ACM Queue*.

CONTRIBUTED ARTICLES

Co-chairs

Al Aho and Georg Gottlob

Board Members

Yannis Bakos; Gilles Brassard; Alan Bundy;
Peter Buneman; Ghezzi Carlo;
Andrew Chien; Anja Feldmann;
Blake Ives; James Larus; Igor Markov;
Gail C. Murphy; Shree Nayar; Lionel M. Ni;
Sriram Rajamani; Jennifer Rexford;
Marie-Christine Rousset; Avi Rubin;
Abigail Sellen; Ron Shamir; Marc Snir;
Larry Snyder; Veda Storey;
Manuela Veloso; Michael Vitale;
Wolfgang Wahlster;
Andy Chi-Chih Yao; Willy Zwaenepoel

RESEARCH HIGHLIGHTS

Co-chairs

David A. Patterson and

Board Members

Martin Abadi; Stuart K. Card;
Deborah Estrin; Shafi Goldwasser;
Monika Henzinger; Maurice Herlihy;
Norm Jouppi; Andrew B. Kahng;
Gregory Morrisett; Linda Petzold;
Michael Reiter; Mendel Rosenblum;
Ronitt Rubinfeld; David Salesin;
Lawrence K. Saul; Guy Steele, Jr.;
Gerhard Weikum; Alexander L. Wolf

WEB

Co-chairs

Marti Hearst and James Landay

Board Members

Jason I. Hong; Jeff Johnson;
Greg Linden; Wendy E. MacKay



ACM Copyright Notice

Copyright © 2009 by Association for Computing Machinery, Inc. (ACM). Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or fee. Request permission to publish from permissions@acm.org or fax (212) 869-0481.

For other copying of articles that carry a code at the bottom of the first or last page or screen display, copying is permitted provided that the per-copy fee indicated in the code is paid through the Copyright Clearance Center; www.copyright.com.

Subscriptions

An annual subscription cost is included in ACM member dues of \$99 (\$40 of which is allocated to a subscription to *Communications*); for students, cost is included in \$42 dues (\$20 of which is allocated to a *Communications* subscription). A nonmember annual subscription is \$100.

ACM Media Advertising Policy

Communications of the ACM and other ACM Media publications accept advertising in both print and electronic formats. All advertising in ACM Media publications is at the discretion of ACM and is intended to provide financial support for the various activities and services for ACM members. Current Advertising Rates can be found by visiting <http://www.acm-media.org> or by contacting ACM Media Sales at (212) 626-0654.

Single Copies

Single copies of *Communications of the ACM* are available for purchase. Please contact acmhelp@acm.org.

COMMUNICATIONS OF THE ACM

(ISSN 0001-0782) is published monthly by ACM Media, 2 Penn Plaza, Suite 701, New York, NY 10121-0701. Periodicals postage paid at New York, NY 10001, and other mailing offices.

POSTMASTER

Please send address changes to *Communications of the ACM*, 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA



Association for
Computing Machinery



Printed in the U.S.A.



DOI:10.1145/1592761.1592762

Moshe Y. Vardi

Is The Image Crisis Over?

When *Communications* relaunched in July 2008, the issue included a “Viewpoint” column by Rick Rashid, entitled “Image Crisis: Inspiring a New Generation of Computer Scientists.”

Has anything changed in that regard in the last 17 months?

Let us first recall what the “image crisis” was. The computing field went through a perfect storm in the early 2000s: the dot-com and telecom crashes, the offshoring scare, and a research-funding crisis. After its glamour phase in the late 1990s, the field seems to have lost its luster. This has resulted in a precipitous drop in North American enrollments in undergraduate computer science programs. The Computing Research Association’s Taulbee Survey traced how U.S. computer-science enrollments started dropping in 2001, reaching 50% of the 2000 level in 2005, and then staying flat through 2007. It seemed that high-school students made a collective decision that computing is not a promising career, and simply voted with their feet.

The enrollment plunge led to the establishment of the Image of Computing Task Force in 2005 as a U.S. response to “lead a national coordination effort to expose a realistic view of opportunities in computing.” In 2008, the U.S. National Science Foundation funded a joint project with the WGBH Educational Foundation and ACM to “research and design a new set of messages that will accurately portray the field of computing.” Then, in March 2009, the 2007–2008 Taulbee Survey came out with data indicating that freshmen CS enrollments grew by almost 10% between 2007 and 2008.

So, is the “image crisis” over?

Taking a longer-term view of CS en-

rollments, one notices that the enrollments drop of the early 2000s came after a huge rise in enrollments in the late 1990s. In fact, CS enrollments have always been cyclical. The latest boom-bust cycle was triggered by the “Internet revolution,” but an earlier boom-bust cycle, in much of the 1980s, was triggered by the “PC revolution.” It is not clear what a “normal” level of CS enrollments would be. Was the “crisis” a real crisis?

There is no doubt that CS enrollments are hugely affected by the economic environment. The most recent rise in enrollments is probably tied to the recent economic crisis. While finance looked like a promising career two years ago, that gleam is clearly tarnished. A career in computing suddenly seems much more promising than a career on Wall Street. The rise in CS enrollments will probably continue to rise for the next few years.

We should not, however, let a good “crisis” go to waste. From it we learned that the image of our field is important and that the image of our field is not necessarily a positive one. Trying to change that image is a noble goal, though I doubt nothing short of a massive marketing campaign, at the probable cost of tens of millions of dollars, can add more color to the prevailing picture.

What we did not seem to learn is that the image of our field may be related to the *reality* of our field. We are woefully ignorant about the reality of computing careers. According to the Bureau of Labor Statistics, there

are close to four million information technology workers in the U.S. that span across several job categories. (Analogous information about the rest of the world is very difficult to obtain.) What we do not know is how computing graduates fit in this picture. We do not know how to measure the career outcome or value of a computing degree. How are CS graduates distributed across various industry sectors? What are their career trajectories? Peter Freeman and William Aspray’s 1999 report *The Supply of Information Technology Workers in the United States*, studied the supply of and demand for IT workers in the U.S. at a macro rather than the individual scale in an effort to better understand these issues. Yet today we still do not know how to determine the quality of computing careers, say in terms of lifetime income, job stability, autonomy and self-direction, promotion opportunity, and the like, compares to other careers, say, in electrical or chemical engineering. Moreover, we clearly do not know how to change the image problem of our field as viewed by women and most minorities.

These, I believe, are huge gaps in our knowledge. I do not see how we can develop messages that will accurately portray the field of computing, if we ourselves do not have an accurate portrait of the field. Before the “image crisis” completely fades away, let us not let it go to waste.

Moshe Y. Vardi, EDITOR-IN-CHIEF

DOI:10.1145/1592761.1592764

Pay for Editorial Independence

THOUGH MOSHE Y. VARDI'S Editor's Letter "Open, Closed, or Closed-Open Access?" (July 2009) addressed the question of who pays the bills, we must also address the price of quality.

By 1424, Cambridge University library had only 122 books; the number today is more than seven million. At the beginning of the 15th century, any of those 122 books would have cost as much as a farm. Following the invention of the printing press by Johannes Gutenberg around 1440, the price of books decreased dramatically due to the reduced cost of manufacturing. With the arrival of the Internet in the late 20th century, the dissemination of information reached unprecedented low cost through the elimination of paper and shipping. So, despite accessing information online at almost no cost, why do we still pay for scientific articles? The answer is we pay for high-quality articles. Ensuring their publication is never free, and the price must account for copy editing, reviewing, and indexing, as well as for something more important, editorial independence.

What differentiates regular journals from excellent journals is their editors' ability to know when to reject mediocre submissions and accept only those that are very good or groundbreaking. Editors must be fully independent to make decisions based on quality, not on financial considerations alone. Readers paying for high-quality articles means editors are free to decide without prohibitive financial pressure.

Answering who pays the bills is important, but determining how much to pay for quality regardless of publishing model is the overriding issue. I hope the market struggle between open and closed publishing models will put prices in their rightful place.

Agustí Solanas, Catalonia, Spain

Abolish Conference Proceedings

As program chair of an ACM conference (Hypertext 2009 <http://informatics.indiana.edu/fil>), I agree with both

Lance Fortnow's Viewpoint "Time for Computer Science to Grow Up" (Aug. 2009) and Moshe Vardi's Editor's Letter "Conferences vs. Journals in Computing Research" (May 2009). Moreover, as an interdisciplinary researcher, I experience firsthand how conference-driven publication practices hurt CS in terms of potential interdisciplinary collaboration, reach, and visibility.

That's why I propose the abolition of conference proceedings altogether. Submissions should instead go to journals, which would receive more and better ones, with refereeing resources shifting naturally from conferences to journals. As a result, journals would improve their quality and speed up their processes. With the CS community's full attention, the review process would be more rigorous and timely. Deadlines would no longer be so concentrated, and scientists would submit better work, revise as needed, and profit immediately from reviewer feedback; the same referee would judge improvements to a particular submission.

In many cases where conferences and journals are aligned, presentations could be invited from among the best papers published in the previous year. For newer areas and groundbreaking work, a conference or workshop could still accept submissions but would not publish proceedings. Publishing would be the job of journals.

ACM should shepherd such a transition as publisher of both the proceedings of most top computing conferences and of many top computing journals.

Filippo Menczer, Bloomington, IN

As a young researcher, I was intrigued by Lance Fortnow's explanation (Aug. 2009) of why the CS community is dominated by conference proceedings. However, I was less excited by his proposed solution, that "...leaders of major conferences must make the first move, holding their conferences less frequently and accepting every reasonable paper for presentation without proceedings." I fear such a move would

not have the intended effect of a more journal-focused community.

Fortnow only touched on the reason he thinks it wouldn't work, that CS journals have a reputation for slow turnaround, with most taking at least a year to make a publish/reject decision and some taking much longer before publishing. These end-to-end times are unheard of in other fields where journal editors make decisions in weeks, sometimes days.

Combine this with the trend toward fewer post-doc positions to begin with and young researchers trying to launch their careers by proving their ability to publish their research. Conference publications provide the quick turnaround they need, whereas journals can sometimes represent too great a risk early in a career.

For CS to grow up, CS journals must grow up first.

Jano van Hemert, Edinburgh, U.K.

Inspire CS Passion in All

If someone were to ask me to name the top-four "formidable challenges" facing computer science, I would not have come up with the ones listed by Bob Violino in his news story "Time to Reboot" (Apr. 2009):

- Declining enrollment in degree programs since 2001;
- Underrepresentation of women and minorities;
- Negative perception of CS among K-12 students; and
- Reports saying the rate of innovation in the field has decreased.

With the possible exception of the last one, all are "soft" issues that have little to do with science. Encouraging students to pursue a career, or at least an interest, in CS is worthwhile and part of the ambassadorial role everyone in the field should play anyway. It also makes sense for some in the field to focus on these areas. However, it should in no way crowd out its core scientific pursuits. Emphasize instead challenges in parallel programming, image processing, language develop-

ment, data mining, social computing, robotics, and bioengineering.

I admit I find the second item largely irrelevant. What is the optimal percentage in CS of men, women, and racial/ethnic groups? Is it a problem to be overrepresented in certain countries by men, whites, Indian, or Chinese CS professionals? Why not inspire a passion for the profession among all people, and let whoever is interested join the ranks? Anything else, like somehow encouraging or targeting certain groups while downplaying others, is racial, ethnic, or gender discrimination.

Ron Pyke, Bellevue, WA

Who Speaks for Books?

By objecting to the proposed Google book-search settlement without offering alternatives, Pamela Samuelson in "The Dead Souls of the Google Book Search Settlement" (July 2009) seemed to be allowing the perfect to be the enemy of the good, addressing policy issues beyond the immediate legal issues in the lawsuit. Each of her concerns could, however, be ameliorated without rejecting the basic framework.

Among the issues before the court is whether the Authors Guild is an appropriate representative of the plaintiff class. If the court decides the Authors Guild is an inappropriate class representative, the details of the settlement are irrelevant. However, this would also mean that the litigation will be prolonged and the ultimate results uncertain. In the meantime, everyone, including many scholars, will be without the benefit of Google Book Search. It may thus be prudent to allow the Authors Guild to represent a class, regardless of its limited and perhaps specialized membership, and focus on the merits of the settlement.

Orphan works by definition have no rights holders to speak for them. Presumably, some rights holders would like to have their works made freely available, while others would require compensation for and/or limits to access, and still others would refuse any access at all. A judicious approach might be to exclude orphan works from the settlement. Ideally, the U.S. Congress would establish a policy toward orphan works that would reflect policy choices beyond

the interests of the parties to a lawsuit.

That two de facto monopolies would be created by a settlement is a concern but can be addressed while still maintaining the basic settlement. First, the Registry would be the only general source of rights to digitize non-orphan, non-public domain books published before January 2009, but there is the possibility that another rights group could be organized in competition with the Registry, as BMI was founded to compete with ASCAP in music performing rights. Second, unless others are likely to receive terms comparable to Google's under the settlement, they could hesitate to seek a license from the Registry and undertake a comparable project without a license. As a result, Google may be the only licensee with rights to scan the non-orphan, non-public domain books. But even if that happens, others could develop business models analogous to those of Lexis-Nexis and Westlaw.

Google will also have the authority to develop pricing algorithms for the books. For that to happen, the court can clarify that any approval of the settlement would not preclude future antitrust review, and Google would then be bound by the law, just like any other company.

As Samuelson wrote, the settlement would achieve a substantial restructuring of the landscape of access to books. While there may be understandable concern about such far-reaching results from the settlement of a single lawsuit, it alone is insufficient to require the parties to continue litigating. By resolving this case with relatively modest adjustments, we would have the substantial benefit of greater access than ever to books.

Yee Wah Chin, New York

Author's Response:

Orphan-works legislation would be desirable and, in my judgment, more likely if the settlement is not approved than if it is.

The idea of excluding orphan works from the settlement is interesting, but one that would cause the settlement agreement to fall apart, since getting a license to orphan works is one of Google's main objectives in the class-action settlement. The Authors Guild and AAP also care about this because their authors and publishers will

benefit if Google pays BRR royalties for its exploitations of the orphans that, under the settlement agreement, are to be paid out to registered publishers and authors, after BRR's costs are deducted.

I am familiar with the "perfect is the enemy of the good" argument and think it has some relevance here. But part of what concerns me is what will happen in 10 years, 20 years, and beyond and that monopolies tend to engage in exclusionary conduct and excessive pricing. That is a reason to be very careful about approval of this settlement.

Pamela Samuelson, Berkeley, CA

Meeting Maurice

I was fortunate to have Professor Maurice Wilkes (interviewed by David P. Anderson, Sept. 2009) as my academic advisor 1953–1954 when I programmed for the EDSAC while earning a post-graduate diploma in "Numerical Analysis and Automatic Computing" that included writing a thesis on programming for the EDSAC. To the best of my knowledge, that was the first year a post-graduate computer science degree was ever awarded.

I have since had the pleasure of meeting Maurice many times, during both his association with Digital Equipment Corporation in Maynard, MA, in the 1980s and more recently in Cambridge, U.K., where I visited him around the time of his 95th birthday. Despite his advanced age, he drove me from his office to his home to have tea with him and his wife Nina, and later from his home to the station to catch a train back to London.

I am 20 years younger than Maurice but hope I will match his sprightliness when I reach his age.

Peter Wegner, Providence, RI

The interview with Maurice Wilkes by David P. Anderson (Sept. 2009) was of great interest, in spite of the unkind remarks regarding Alan Turing. If not for a few greats of World War II like Turing and Robert Watson-Watt [radar pioneer], the interview would have been carried out in German.

George T. Jacobi, Milwaukee, WI

Communications welcomes your opinion. To submit a Letter to the Editor, please limit your comments to 500 words or less and send to letters@cacm.acm.org.

© 2009 ACM 0001-0782/09/1100 \$10.00



IEEE 6TH WORLD CONGRESS ON SERVICES (SERVICES 2010)

July 5-10, 2010, Miami, Florida, USA, <http://www.servicescongress.org/2010>
Modernization of all vertical services industries including finance, government, media, communication, healthcare, insurance, energy and ...

IEEE 7th International Conference on Services Computing (SCC 2010)



In the modern services and software industry, Services Computing has become a cross-discipline that covers the science and technology of bridging the gap between Business Services and IT Services. The scope of Services Computing covers the whole lifecycle of services innovation research that includes business componentization, services modeling, services creation, services realization, services annotation, services deployment, services discovery, services composition, services delivery, service-to-service collaboration, services monitoring, services optimization, as well as services management. The goal of Services Computing is to enable IT services and computing technology to perform business services more efficiently and effectively.

<http://conferences.computer.org/scc/2010>

IEEE 8th International Conference on Web Services (ICWS 2010)



As a major implementation technology for modernizing services industry, Web services are Internet-based application components published using standard interface description languages and universally available via uniform communication protocols. In the eighth year, the program of ICWS 2010 will continue to feature research papers with a wide range of topics focusing on various aspects of implementation and infrastructure of Web-based services. ICWS has been a prime international forum for both researchers and industry practitioners to exchange the latest fundamental advances in the state of the art on Web services.

<http://www.icws.org>



Submission Deadlines For Abstracts and Papers

ICWS 2010: Feb. 1, 2010
SCC 2010: Feb. 15, 2010
SERVICES 2010: March 6, 2010
CLOUD 2010: March 6, 2010

Contact: Liang-Jie Zhang (LJ) at
zhanglj@ieee.org
(Steering Committee Chair)

IEEE 3rd International Conference on Cloud Computing (CLOUD 2010)

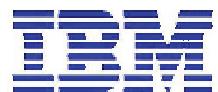


Cloud Computing is becoming a scalable services delivery and consumption platform in the field of Services Computing. The technical foundations of Cloud Computing include Service-Oriented Architecture (SOA) and Virtualizations of hardware and software. The goal of Cloud Computing is to share resources among the cloud service consumers, cloud partners, and cloud vendors in the cloud value chain. Major topics cover Infrastructure Cloud, Software Cloud, Application Cloud, and Business Cloud.

<http://www.thecloudcomputing.org/2010>



Celebrating 125 Years
of Engineering the Future



DOI:10.1145/1592761.1592765

In the Virtual Extension

Communications' Virtual Extension brings more quality articles to ACM members. These articles are now available in the ACM Digital Library.

Offshoring and the New World Order

Rudy Hirschheim

Outsourcing as a means of meeting organizational information technology (IT) demands is a commonly accepted and growing practice—one that is continually evolving to include a much wider set of business functions: logistics, accounting, human resources, legal, and risk assessment. Firms are rushing overseas to have their IT work performed by offshore vendors. Such change, many argue, is merely the natural progression of first moving blue-collar work overseas followed by white-collar work. IT jobs are most visible to us in the IT field, but the same is happening to other business functions/processes. This article analyzes some of the implications for the IT field from a U.S. perspective.

If Your Pearls of Wisdom Fall in a Forest...

Ralph Westfall

"Build a better mousetrap and the world will beat a path to your door." Will that really happen? Not if the world doesn't discover your concept! Few of us work with mousetrap technologies, but many of us do have good ideas that we would like to share with others. For academics, communicating research findings is very important for advancement. This article tells how to help more people find your good ideas among the hundreds of billions of Web pages in the maze that is the Internet.

Quantifying the Benefits of Investing in Information Security

Lara Khansa and Divakaran Liginla

Quantifying the benefits of investing in information security has been a challenge for researchers due to the lack of credible and available firm-level data. In this article, the authors use the revenue data from information security firms to quantify investment in information security products and services. They demonstrate that higher information security investments, especially identity and access management, are instrumental in reducing the severity of malicious attacks, which could be seriously detrimental to the stock price of breached firms. More importantly, they show that greater investment in

information security is associated with an increase in the stock price of information security firms.

iCare Home Portal: An Extended Model of Quality Aging E-Services

Wei-Lun Chang, Soe-Tsyer, and Eldon Y. Li

As the worldwide elderly population is expanding much faster than that of the younger generation, digital devices and applications that help care for the elderly are increasingly popular. This study proposes an electronic iCare model that utilizes collective decision-making to underscore the desired care quality elements of consumer participation and continuous quality improvement. This model goes beyond environmental, physical, and relationship aspects to envision possible forms of iCare e-services and the ontology required to empower agents to fulfill the collective decision process.

Computing Journals and their Emerging Roles in Knowledge Exchange

Aakash Taneja, Anil Singh, M.K. Raja

Scholarly articles in journals use citations to both provide a basis and context for the current work. These journals play three unique roles as sources, storers, and synthesizers of knowledge communication. While the journals' role as sources is recognized, their role as synthesizers is relatively unnoticed. This study investigates the interconnectedness of an expanded list of 50 computing journals through citation analysis to explore their roles as sources, storers and synthesizers in their communication network. It also draws on visual analysis along with dependence calculations to provide an insight about the domains and positions of computing journals.

And What Can Context Do For Data?

C. Bolchini, C. A. Curino, G. Orsi, E. Quintarelli, R. Rossato, F. A. Schrieber, and L. Tanca

Within an information system, the knowledge demands of users may depend on two different aspects: the application domain that represents the reality under examination, and the working environment, in other words, the context. This article shows how fitting data, possibly assembled

and integrated from many data sources, to applications needs is tantamount to fitting a dress—context is the tailoring scissors. Using a real-estate company as a backdrop, the authors define a tree-based context model and a context-guided methodology to support the designer in identifying the contexts for a given application scenario and to choose the correspondingly relevant subsets of data.

Why Web Sites Are Lost (and How They're Sometimes Found)

Frank McCown, Catherine C. Marshall, and Michael L. Nelson

One day a Web site is up; the next day it's all but disappeared. We probably know someone who has experienced the loss of a Web site, either through hard-drive crashes, ISP bankruptcies, and some such event. The authors survey individuals who have lost Web sites and examine what happened and how these individuals went about reconstructing their sites, including how they recovered data from search engine caches and Web archives. The findings suggest that digital data loss is likely to continue since backups are frequently neglected or performed incorrectly. Moreover, respondents perceive that loss is uncommon and that data safety is the responsibility of others. The authors suggest this benign neglect be countered by lazy preservation techniques.

Technical Opinion: Steering Self-Learning Distance Algorithms

Frank Nielsen

The concept of distance expresses the distortion measure between any pair of entities lying in a common space. Distances are ubiquitous in computational science. We concisely review the role and recent development of distance families in computer science. Today, the most appropriate distance functions of complex high-dimensional data sets can no longer be guessed manually and hard-coded, but rather must be fully automatically learned, or even better, partially user-steered for personalization. We envision a whole new generation of personalized information retrieval systems incorporating self-learning built-in distance modules, and providing user interfaces to better take into consideration the subjective tastes of users/groups.

The *Communications* Web site, <http://cacm.acm.org>, features more than a dozen bloggers in the BLOG@CACM community. In each issue of *Communications*, we'll publish excerpts from selected posts.

DOI:10.1145/1592761.1592766

<http://cacm.acm.org/blogs/blog-cacm>

Computer Science Curriculum, Deceptive Advertising

Ramana Rao writes about the evolution of computer science curriculum and Greg Linden reflects on ethics and advertising.



From Ramana Rao's "As I Once Thunk"

A blog posting by a long-ago associate has lingered in a Firefox tab for quite some time. The posting by Dan Weinreb is "Why Did MIT Switch from Scheme to Python?" with further discussion on Hacker News. Don't be fooled by the Hacker News title, because the thread is really about the evolution of a computer science (CS) curriculum.

Thirty years ago this August, I arrived at MIT, and registered in the course 6.031, *The Structure and Interpretation of Computer Programs*. The lecturer was Robert Fano, an electrical engineer famous for his work in information theory who moved into computer science in the 1960s, at least a decade before the first undergraduate CS curriculum emerged. My recitation instructor was Christos Papadimitriou, who was already distinguishing himself as a theoretical computer scientist. Within weeks, as I listened to the magic delivered by the Italian- and

the Greek-tinged voices, I decided to major in computer science.

In the next few years Hal Abelson and Gerald Jay Sussman would concentrate 6.031 into 6.001, the very renumbering punctuating an essential victory. They would pack Algol and Lisp into an elemental Scheme, unifying key concepts and intensifying workload in a curricular tour de force. Yet, my initial reaction was I quite liked seeing the two parts separately for their own individual elegances.

I can remember Fano, with a forceful enthusiasm, explaining call-by-name as he dashed out a diagram with contours and the magic of thunks as chalk hit chalkboard in the Green Building lecture hall. And in the second half of the course the same crystalline clarity emerged in Papadimitriou's recitations of Lisp's dynamic forms and list structures and meta-forms of representation and computation (that's data and control abstractions). As 6.001 came into force, my worry was that the sheer grind and the forging of axes into a volume would lose the po-

tency of the original. However, over the years, hearing wows and even avowals of love for 6.001, I accepted that perhaps not much was lost.

The present shift, with the retiring of 6.001 and the introduction of 6.01 this last year, is much more significant, belied by the simple loss of a zero. The course may still contain a good chunk of what was there, but it hits at perhaps one level of abstraction too high and with an abundance of grit and whirl of "real world" engineering and so much more stuff. All considered, I doubt the MIT CS curriculum will be damaged by this. Still, the transformation brings me to active discussions over the last few years in ACM venues on the declining enrollments in CS programs and on the shaping of computing courses for other fields and the primary education system. You see the same vector of replacing elemental formulations with ones that accommodate teaching students in other fields by contextualizing with relevant or engaging problem domains.

In all this discussion of pragmatic factors and educational theories, I wonder whether what may be lost in translation (for a while, anyway) is exactly the magic at the heart of the field that draws people into it. Thirty years ago the field's own core curriculum was still being formed for those in the field itself and now with the computing crescendo of the 1990s, we are in the next era of cultural diffusion. Even if it might make sense for MIT to retire 6.001, is it not time for an analogous course for a broader audience—say, simply Computing—to emerge? (Cer-

tainly both Peter J. Denning and Jeanette M. Wing have called along this direction for many years.) Just as in high-school courses in calculus and physics, distilled and pure forms may not just be a good idea as a foundation, prior to utility, but also for their innate beauties that inspire and subscribe. And though certainly the voices that I heard are to be thanked for my own arrival, by now it's clear that the forms themselves possess a voice of their own.



From Greg Linden's "Is Advertising Inherently Deceptive?"

Will Rogers once said, "Advertising is the art of convincing people to spend money they don't have for something they don't need." According to Will, advertising is inherently deceptive, and most profitable when it hoodwinks people into paying more for something than they should.

Another view is that consumers lack information and advertising can provide information. In this view, the opportunity for deception only exists because of missing information about reputations and alternatives. If advertisements are relevant enough to inform consumers, then opportunities for deception fade.

Ultimately, whether deception or relevance is more profitable to advertisements may depend on margins versus conversion rates. Deceptive advertising tends to have high profits on each sale, but usually very low conversions. Useful advertisements will yield much tighter margins, but have a much higher volume of conversions.

An extreme example is email spam, horribly deceptive advertisements with awful conversions. And the data there may give us some hope. One of the worst forms of deceptive advertising, email spam appears to be a barely profitable enterprise despite its ubiquity. This may suggest that deception does not inherently maximize profits.

Another example is search advertising. By targeting advertising closely to search keywords and intent, companies like Google have not only made search advertising very lucrative, but also relevant and useful to searchers. In search, ads are highly targeted, rare-

ly deceptive, only occasionally annoying, and often helpful.

But most other forms of advertising remain irrelevant and annoying. The most common technique we see still is broadly blasting ads across all eyeballs. It would be good to make advertising more helpful, relevant, and useful to people. Is it possible?

For a few years now, I have worked on personalized advertising. Personalized advertising tries to make advertising more useful and relevant to people by targeting ads to individual interests and needs.

Recently, I have been struggling with a moral question. Let's say we build more personalization techniques and tools that allow advertisers and publishers to understand people's interests and individually target ads. How will our tools be used? Will they be used to provide better information to people about useful products and services? Or will they be used for deeper and trickier forms of deception?

For me, it is an ethical issue that cuts deep. If personalized advertising will not be used to benefit people, to improve the usefulness of advertising, then I want no part of it. It seems clear that personalization can make ads more relevant, but I fear it also could be possible to use deep knowledge of individual interests to target deceptions. Which will advertisers do? Which will be more profitable?

I am hopeful that we can improve advertising and that advertising will be most profitable for most advertisers when it is useful and relevant. I am hopeful that any deceptions will be marginalized by a flood of more useful alternatives. I remain hopeful that advertising can move toward a helpful information stream and away from the art that Will Rogers deplored.

But, to be quite honest, I sometimes have doubts about the answer, which is why I bring it up for discussion. What do you think? Is advertising an industry fundamentally fueled by deception? Or is advertising better understood as a stream of information that, if well directed, can help people?

Readers' comments

I think in the perfect case of a recommender system, advertisements as such become irrelevant. You'd have a

direct mapping between customer desire and product awareness. The onus of sales would be on producing products that people wanted rather than on the ability to generate awareness for them.

Naturally, that's not how reality works, but it might be useful, seen as one of the extremes of a continuum. The other end of the spectrum is creating a market for things people don't actually want.

The latter sounds like something that's probably not good and the current state of affairs is somewhere between the two. It would seem that if you can admit the way that things currently work isn't evil, the case of targeted advertisement is increasingly less so.

The problem, of course, is that the function isn't entirely continuous. Targeting can potentially be used to trick people. It's not so much that tricking people is novel in advertising, but as you learn more about how a person ticks, your trickery could be that much more potent.

—Patrick Wheeler

Very few ads are intended to deceive. Most mass-media ads are designed to appeal to people's subconscious desires—to be attractive, secure, etc.—and to link a product to that desire. Online ads will dominate advertising only if they are effective at making that linkage. Personalized ads ought to be effective at selecting audiences for their subconscious desires by analyzing their conscious choices. See the BBC documentary "The Century of the Self" (available on Google video), especially segment three on the rise of lifestyle marketing, for an example of using conscious answers to infer subconscious desires.

If there is a continuum, it may be from informational ads with conscious appeal, to influential ads with subconscious appeal. Personalized targeting ought to improve the effectiveness of both kinds.

Note that we may still be squeamish about improving "influential" advertising. It doesn't make subconscious desires conscious; rather, it adds a subconscious link to a product—and thereby makes us willing to buy more of it and spend more on it than if we had not seen the ad.

—Ken Novak

Ramana Rao is the CEO of iCurrent Inc. **Greg Linden** is the founder of Geeky Ventures.

© 2009 ACM 0001-0782/09/1100 \$10.00



DOI:10.1145/1592761.1592767

David Roman

Internet Addiction: It's Spreading, but is it Real?

Communications' news stories cover a lot of ground and sometimes raise a provocative question. A recent case in point: "Is Internet Addiction Real?" I was sure the answer to this question was 'no' after reading a story posted on the site that told of a 15-year-old boy who was beaten to death at an Internet addiction treatment center in China (<http://cacm.acm.org/news/41829>) that sounded more like a re-education camp. That impression wasn't lessened by the Chinese government's estimate that 10% of its Internet users under the age of 18 are addicts.

But I wavered when I learned that Internet addiction centers are growing outside China as well, in South Korea, Taiwan, and the U.S. We published a story about ReSTART, an Internet detox center located a laser shot from Microsoft's headquarters (<http://cacm.acm.org/news/42675>). It treats behaviors worthy of a



12-step program, such as a monomaniacal desire for online time, an inability to disconnect, and lying about Web habits. But it's also true that many overworked software programmers would fail ReSTART's survey on Internet addiction (http://www.netaddiction.com/resources/internet_addiction_test.htm). As a corrective, neither the American Psychiatric Association's *Diagnostic and Statistical Manual of Mental Disorders* nor the World Health Organization's *International Classification of Diseases* recognize Internet addiction as a disorder, and the *Indian Journal of Psychiatry* reports that Internet Addiction Disorder (IAD) may have started as a satirical hoax (<http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2738353/>).

Addictions are destructive. Smoking kills 438,000 people in the U.S. each year, reports the Centers for Disease Control. And it's difficult to imagine how the problems that what IAD brings to individuals, relationships, and families could ever match alcoholism's rain of pain.

Addiction? Without stronger evidence, the jury is still out.

ACM Member News

INTRODUCING ACM'S MULTIMEDIA CENTER

Are you interested in watching a short video about social robots using Facebook? What about an instructional video on the uses of PenLight, which combines a mobile projector and a digital pen to create a dynamic visual overlay? How about a 23-minute discourse on articulated mesh animation from multiview silhouettes? These videos and others are currently available from ACM's Multimedia Center, <http://multimedia.myacm.org/>, which is offering free access to a collection of videos about a multitude of areas of computing.

The ACM Multimedia Center will feature a total of 10 videos, with a new video replacing an existing one each week. The current selection of videos range in length from nearly two minutes (video-based emergent storytelling) to an hour and two minutes (Frances E. Allen's 2006 A.M. Turing Award lecture), with most videos ranging in length from a few minutes to 20 minutes. Visitors can learn more about the context of each video by clicking on the link to its source in the ACM Digital Library. (Access to the source's full text requires an ACM membership and a Digital Library subscription.)

SC09

The premier international conference on high performance computing, networking, storage, and analysis, SC09 will take place Nov. 14–20 in Portland, OR. SC09's theme is "Computing for a Changing World," and will also include special focus discussions on bio-computing, sustainability, and 3D Internet.

The SC09 technical program will include featured speakers such as Intel Senior Fellow and CTO Justin Rattner, who will deliver an opening address; Leroy Hood, president and cofounder of the Institute for Systems Biology, who is an invited plenary speaker; and former U.S. Vice President Al Gore who is the conference keynote speaker.

For more information on SC09, please visit <http://sc09.supercomputing.org>.

Science | DOI:10.1145/1592761.1592768

Gary Anthes

Deep Data Dives Discover Natural Laws

Computer scientists have found a way to bootstrap science, using evolutionary computation to find fundamental meaning in massive amounts of raw data.

MINING SCIENTIFIC DATA for patterns and relationships has been a common practice for decades, and the use of self-mutating genetic algorithms is nothing new, either. But now a pair of computer scientists at Cornell University have pushed these techniques into an entirely new realm, one that could fundamentally transform the methods of science at the frontiers of research.

Writing in a recent issue of the journal *Science*, Hod Lipson and Michael Schmidt describe how they programmed a computer to take unstructured and imperfect lab measurements from swinging pendulums and mechanical oscillators and, with just the slightest initial direction and no knowledge of physics, mechanics, or geometry, derive equations representing fundamental laws of nature.

Conventional machine learning systems usually aim to generate predictive models that might, for example, calculate the future position of a pendulum given its current position. However, the equations unearthed by Lipson and Schmidt represented basic invariant relationships—such as the conservation of energy—of the kind that govern



Cornell University's Michael Schmidt, left, and Hod Lipson with one of the double pendulums used in their experiments.

the behavior of the universe.

The technique may come just in time as scientists are increasingly confronted with floods of data from the Internet, sensors, particle accelerators, and the like in quantities that defy conventional attempts at analysis. "The technology to collect all that data has far, far surpassed the technology to analyze it and understand it," says

Schmidt, a doctoral candidate and member of the Cornell Computational Synthesis Lab. "This is the first time a computer has been used to go directly from data to a free-form law."

The Lipson/Schmidt work features two key advancements. The first is their look for invariants, or "conservations," rather than for predictive models. "All laws of nature are essentially laws of conservation and symmetry," says Lipson, a professor of mechanical engineering. "So looking for invariants is fundamental."

Given crude initial conditions and some indication of what variables to consider, the genetic program churned through a large number of possible equations, keeping and building on the most promising ones at each iteration and eliminating the others. The project's second key advance was finding a way to identify the large number of trivial equations that, while true and invariant, are coincidental and not directly related to the behavior of the system being studied.

Lipson and Schmidt found that trivial equations could be weeded out by looking at ratios of rates of change in the variables under consideration. The program was written to favor those

equations that were able to use these ratios to predict connections between variables over time. "This was one of the biggest challenges we were able to overcome," Lipson says. "There are infinite trivial equations and just a few interesting ones."

Like human scientists, the software favors equations with the fewest terms. "We want to find the simplest equation powerful enough to predict the dynamics of the system," Schmidt says.

Applying Artificial Intelligence

Scientific data has become so voluminous and complex in many disciplines today that scientists often don't know what to look for or even how to start analyzing it. So they are applying artificial intelligence, via machine learning, to giant data sets without precisely specifying in advance a desired outcome. Unlike AI systems of the past, which were usually driven by hard-coded expert rules, the idea now is to have the software evolve its own rules primed with an arbitrary starting point and a few simple objectives.

Automating the discovery of natural laws marks a major step into a realm that was previously inhabited solely by humans.

Eric Horvitz, an AI specialist at Microsoft Research, says it's only the beginning. "Computers will grow to become scientists in their own right, with intuitions and computational variants of fascination and curiosity," says Horvitz. "They will have the ability to build and test competing explanatory models of phenomena, and to consider the likelihoods that each of the proposed models is correct. And they will understand how to progress through a space of inquiry, where they consider the best evidence to gather next and the best new experiments to perform."

A major challenge facing the European Organization for Nuclear Research (CERN) is how to use the 40 terabytes of data that its Large Hadron Collider is expected to produce every day. Processing that amount of data would be a challenge if scientists knew exactly what to look for, but in fact they can hardly imagine what truths might be revealed if only the right tests are performed. CERN researchers have turned to Lipson and Schmidt for help in finding a way to search for scientific

laws hidden in the data. "It could be a killer app for them," Schmidt says.

Indeed, Lipson and Schmidt have received so many requests to apply their techniques to other scientists' data that they plan to turn their methodology and software into a freely distributed tool.

Josh Bongard, a computer scientist and roboticist at the University of Vermont, says the Lipson/Schmidt approach is noteworthy for its ability to find equations with very little assumed in advance about their form. "That gives the algorithm more free rein to derive relationships that we might not know about," he says.

Bongard says earlier applications of machine learning to discovery have not scaled well, often working for simple systems, such as a single pendulum, but breaking down when applied to a chaotic system like a double pendulum. Further evidence of the scalability of the Lipson/Schmidt algorithm is its apparent ability to span different domains, from mechanical systems to very complex biological ones, he says.

The Lipson/Schmidt work takes search beyond "mining"—where a specific thing is sought—to "discovery," where "you are not sure what you are looking for, but you'll know it when you find it," Bongard says. A key to making that possible with large stores of complex data is having algorithms that are able to evolve building blocks from simple systems into successively more complex models.

Such methods aim to complement the efforts of scientists but not replace them, as some critics have suggested. "These algorithms help to bootstrap science, to help us better investigate the data and the models by acting like an intelligent filter," says Bongard.

Scientific research for decades has followed a well-known path from data collection (observation) to model formulation and prediction, to laws (expressed as equations), and finally to a higher-level theoretical framework or interpretation of the laws. "We have shown we can go directly from data to laws," says Schmidt, "so we are wondering if we can go from laws to the higher theory."

He and Lipson are now trying to automate that giant last step, but admit they have little idea how to do it. Their tentative first step uses a process of

analogy; a newly discovered but poorly understood equation is compared with similar equations in areas that are understood.

For example, they recently mined a large quantity of metabolic data provided by Gurol Suel, assistant professor at the University of Texas Southwestern Medical Center. The algorithm came up with two "very simple, very elegant" invariants—so far unpublished—that are able to accurately predict new data. But neither they nor Suel has any idea what the invariants mean, Lipson says. "So what we are doing now is trying to automate the interpretation stage, by saying, 'Here's what we know about the system, here's the textbook biology; can you explain the new equations in terms of the old equations?'"

Lipson says the ultimate challenge may lie in dealing with laws so complicated they defy human understanding. Then, automation of the interpretation phase would be extremely difficult. "What if it's like trying to explain Shakespeare to a dog?" he asks.

"The exciting thing to me is that we might be able to find the laws at all," says Lipson. "Then we may have to write a program that can take a very complicated concept and break it down so humans can understand it. That's a new challenge for AI." □

Further Reading

Lipson, H. and Schmidt, M. Distilling free-form laws from experimental data. Science 234 (Apr. 3, 2009), 81–85.

Waltz, D. and Buchanan, B. Automating science. Science 234 (Apr. 3, 2009), 43–44.

King, R., Whelan, K., Jones, F., Reiser, P., Bryant, C., Muggleton, S., Kell, D., Oliver, S. Functional genomic hypothesis generation and experimentation by a robot scientist. Nature 427, 6971 (Jan. 15, 2004), 247–252.

Bongard, J. and Lipson, H. Automated reverse engineering of nonlinear dynamical systems. In Proceedings of the National Academy of Sciences 104, 24 (June 6, 2007), 9943–9948.

Koza, J. Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA, 1992.

Gary Anthes is a technology writer and editor based in Arlington, VA.

© 2009 ACM 0001-0782/09/1100 \$10.00

Electronic Paper's Next Chapter

The technological challenge for researchers working on the next generation of electronic paper is to render color as brightly as traditional paper, without increasing power requirements or end-user costs.

ELCTRONIC PAPER, FIRST created in 1975 by Nick Sheridan at Xerox PARC, has begun to proliferate in consumer electronic devices in recent years. Amazon's Kindle and Sony's Reader, two notable applications of the technology, have transformed e-paper into a mass-market phenomenon. New uses for e-paper technology, such as in advertising, wristwatches, smart cards, and even enhancements for computer peripherals, are cropping up regularly. The presence of e-paper in consumer electronic devices is increasing not only because of its minimal energy requirements, making it ideal for low-power devices, but also because its display quality approaches that of the printed page.

Still, by most accounts, the biggest technological hurdle facing e-paper is the fact that current e-paper color displays are either of poor quality or too expensive to be commercially viable. "Color is the next big challenge for e-paper," says Sheridan, a physicist who cofounded Gyricon LLC, as a spinoff from Xerox PARC, to manufacture e-paper displays. "This is not easy to do, and most of the monochrome technologies cannot be modified to do good quality color. New invention is needed." In addition, current e-paper technology cannot render moving images as well as other display technologies. However, that may soon change as improving the color and rendering capabilities of e-paper is the focus of several research labs.

Even with mass-market e-readers being limited to monochromatic displays, much is happening in this area. Prime View International, a Taiwanese company that manufactures the Amazon Kindle and other electronic read-



An early prototype of a monochromatic Plastic Logic reading device featuring flexible display technology.

ers, has put up \$215 million to buy E Ink Corp., the company that develops the digital-ink technology for those readers. Also, brand-name companies are entering the e-reader market in droves, with Samsung being the most recent entrant with an e-reader that, at least for now, is only available in South Korea. According to the *Wall Street Journal*, Samsung plans to show prototypes of its e-reader for international markets in January 2010 and is negotiating with publishers for content.

Plastic Logic, another company making headlines, is positioning its forthcoming e-reader not as a competitor to the Amazon Kindle or the Sony Reader, but as a device designed for business users. The company says its e-reader, which sports an 8 x 11.5 inch screen, will have 3G and Wi-Fi connectivity and a gesture-based touch interface specifically designed for reading

and working with business documents. Also making headlines is Fujitsu, which earlier this year released the FLEPiia, a color-capable tablet featuring Windows CE 5.0 software and also designed for business documents. While Fujitsu claims the FLEPiia is the first color e-reader on the market, it can display only 260,000 colors (in contrast to the majority of desktop monitors, which can display 16.7 million colors) and is priced in the range of tablet PCs. Currently available only in Japan, the FLEPiia costs \$1,000. By comparison, Amazon's monochrome Kindle 2 costs \$299.

Sheridan, who calls the Kindle a "brilliantly executed document reader," says it and other e-readers are appearing at a fortuitous time, particularly as vast libraries are increasingly being digitized. "The Kindle can access a significant part of this, meaning that

Kindle and other e-readers have created a tipping point for e-paper, but consumers will also want video capabilities, bistable pixels, and thin or flexible displays, says Jason Heikenfeld.

shortly anyone on the globe can selectively download books from a library of millions," he says. However, even with the promise of so much content, potential buyers may put off purchasing e-readers until the displays can support desktop-quality color in devices that do not cost as much as a tablet PC.

Kars-Michiel Lenssen and his team at Philips Research in Eindhoven, Netherlands, are working to solve this problem. Lenssen, who is director and principal scientist at Philips Research, started the color e-paper project several years ago with the goal of making low-power, color e-paper brighter than is currently possible. "We believed that electronic paper would enable new applications, but we also realized that bright colors would be required for a really broad market acceptance in the future," Lenssen says. "That's why Philips decided to start a dedicated research project on this topic."

Philips' technology, called in-plane electrophoretics, is different from E Ink's electrophoresis technique. With the electrophoresis technique, used in the Kindle and other popular e-readers, an electric field controls titanium dioxide particles that are suspended in capsules. By applying an electric current, the particles can be forced to the top of the capsules. When the particles are near the display's surface, the display appears white because light is reflected or scattered. When the particles are farther away from the surface, the display appears dark because light is

absorbed. By selectively making certain areas light or dark, fonts and images can be rendered on the display.

Lenssen's in-plane electrophoretics technique, in contrast, relies on two particle-filled capsules for each pixel, one containing yellow and cyan, the other magenta and black. By controlling voltages, the colored particles either spread across the pixel or move out of sight altogether, making it possible to render different colors by controlling the number of colored particles shown. To create white, the particles simply shift to reveal the white substrate beneath the capsules. With in-plane electrophoretics research now maturing, Lenssen and his team are exploring several applications for the technology, with the next step being to bring the technology to production in real-world products.

Lenssen says he is looking beyond e-readers as the primary application. "We think there is more potential, particularly when bright color e-paper will be available," he says. "For example, replacing printed paper signs in retail with electronic paper could save a lot of money not only on printing costs, but also on distribution costs and labor costs for installing and replacing signs."

Besides display-type applications, such as e-readers and digital signage, Lenssen says there are many other opportunities for in-plane electrophoretics, such as digital surfaces on which color could be changed electronically.

Such surfaces, which Lenssen calls "digital paint," could be used, for example, as electronic skins for consumer devices. Instead of physically exchanging a device's skin with one of a different color, a user could electronically choose his or her preferred color for each occasion.

"Our e-paper technology could also enable patterns that appear on the electronic skin of a device like a kind of electronic tattoo," he says. "Chameleons and cuttlefish are inspiring examples from nature in this respect."

Electrofluidic Display

Another approach to the problem of low-power, high-quality color in e-paper comes from the Novel Devices Lab at the University of Cincinnati. The technology, called electrofluidic display, uses voltage to manipulate colored inks in much the same way that print heads operate in color printers. Jason Heikenfeld, a professor of electrical engineering at the University of Cincinnati and head of the Novel Devices Lab, formed Gamma Dynamics LLC earlier this year to create products based on his electrofluidic display technology. He and his colleagues are considering a wide range of applications, from e-readers to e-windows to tunable casings for electronic devices. "One challenge," he says, "is picking a first target application out of so many opportunities."

Heikenfeld says the Kindle and other e-readers have created a tip-



Jason Heikenfeld, head of the Novel Devices Lab at the University of Cincinnati, is working on electrofluidic display technology based on a process involving pigment dispersion.

ping point for e-paper, but maintains the technology hasn't yet come of age because consumers will eventually want video capabilities, bistable pixels (giving displays the ability to operate for long periods on very little or no power), thin or flexible designs, and, of course, vivid color. "No product or technology on the market is even close to offering this, including the FLEPIa," Heikenfeld says. "With our technology, we are aiming to provide the revolutionary increase in brightness that is not possible using the technologies currently available as a product."

Heikenfeld's electrofluidic display technology is based on a process called pigment dispersion. "The pigments look as good as they would on paper," he says. The technology, which Heikenfeld calls a "major step forward" in color e-paper research, consists of an insulator film situated between the pigment dispersion and an electrode film. When voltage is applied to the electrode film, it creates an electrical force that can stretch the pigment dispersion. "We don't mix the pigments," he says. "We display them in different areas, on demand." Obtaining red, for example, requires overlaying yellow and magenta. When the voltage is removed, the pigment dispersion bounces back to its favored geometry of a small droplet or bead shape.

"We have a lot of approaches under development that we have not published yet, so I can't go into all the details," Heikenfeld says.

Heikenfeld's electrofluidic display technology is one of almost a dozen different technologies being developed to create low-power e-paper that can render colors as brilliantly as traditional paper can. Judging by recent developments in terms of display size and power consumption in e-readers coming to market, the future for e-paper technology appears bright. In 10 or 20 years, Heikenfeld says, consumers might see large e-paper modules that are as thin and as flexible as magazines are today, with display brightness approaching that of conventional print.

In Heikenfeld's imagined future, these solar-powered devices will have touch interfaces, communication capabilities, and be so energy efficient that charging them will be an afterthought.

"You might click on an image in a story, and it will provide video or animation," he says. "There is nothing fundamental from an optics or electronics perspective that makes this impossible."

For his part, Sheridan believes e-paper eventually will make power-hungry desktop displays obsolete, and will help make heavy, back-breaking textbooks something school children might learn about in a history lesson on their lightweight e-readers, not lugged around with them in their backpacks. But when it comes to betting whether all paper books will become a thing of the past, the inventor of e-paper is cautious about predicting the obsolescence of the printed page. Sheridan simply suggests that in the future books might be printed on paper.

"E-paper will continue to find important applications, such as in fabric, large displays, and home and building decoration, to mention a few," says Sheridan. "The surest way to predict the future is to invent it. E-paper is rich in potential." □

Further Reading

- Feenstra, B. J. Electrowetting Technology Aims To Improve on the Performance of LCDs for Mobile Applications, *Information Display* 22, 11, 2006, 10–13.*
- Green, A.M., Montbach, E., Miller, N., Davis, D., Khan, A., Schneider, T., Doance, J.W. Energy Efficient Flexible Reflex Displays, *Proc. Int'l Display Research Conf.*, Society for Information Display, 2008, 55–58.*
- Heikenfeld, J., Zhou, K., Kreit, E., Raj, B., Yang, S., Sun, B., Milarcik, A., Clapp, L., Schwartz, R. Electrofluidic Displays Using Young-Laplace Transposition of Brilliant Pigment Dispersions, *Nature Photonics* 3, 5, 2009, 292–296.*
- Lenssen, K.-M. H., Baesjou, P.J., Budzelaar, F.P.M., van Delden, M.H.W.M., Roosendaal, S.J., Stofmeel, L.W.G., Verschueren, A.R.M., van Glabbeek, J.J., Osenga, J.T.M., Schuurmans, R.M. Novel Concept for Full-Color Electronic Paper, *J. Society for Information Display* 17, 4, 2009, 383–388.*
- Sheridan, N. K. and Berkovitz, M. A. A Twisting Ball Display, *Proc. Society for Information Display* 18, 3/4, 1977, 289–293.*

Based in Los Angeles, **Kirk L. Kroeker** is a freelance editor and writer specializing in science and technology.

© 2009 ACM 0001-0782/09/1100 \$10.00

Belated Apology For Turing

British Prime Minister Gordon Brown apologized for the British government's "horrifying" treatment 50 years ago of Alan Turing, the mathematical genius and a founder of modern computing, who was criminally prosecuted and convicted of "gross indecency" in 1952 after admitting to a homosexual experience. To avoid imprisonment, he underwent chemical castration. Two years later Turing committed suicide at the age of 41.

Earlier this year, British computer scientist and blogger John Graham-Cumming launched an online petition campaign urging the British government to apologize. The petition was supported by scientist Richard Dawkins, writer Ian McEwan, and gay-rights activist Peter Tatchell, and it received 31,612 signatures from British citizens and residents before Brown issued an apology.

Graham-Cumming has also written to Queen Elizabeth II, asking that Turing be awarded a posthumous knighthood.

In 1936, Turing wrote his seminal paper, "On Computable Numbers," which established the conceptual and philosophical basis for modern-day computers, and consequently developed the Turing Test, an important measure of success in the field of artificial intelligence. During World War II, Turing developed the Bombe, an electromechanical code-breaking device that enabled Britain to read secret messages encoded by Germany's Enigma cipher machines, complex typewriter-like devices that generated a constantly changing code for its military communications.

ACM President Dame Wendy Hall issued a statement applauding Brown's apology, recognizing his computer science and wartime contributions, and noting that "ACM looks forward to joining with other organizations to celebrate the centenary of Turing's birth in 2012."

Implementing Electronic Medical Records

Despite a number of challenges, patients' medical records are slowly making the transition to the digital age.

IN 2004, PRESIDENT George W. Bush set an ambitious goal for the American health-care industry: by 2014, he wanted all citizens to have access to an electronic medical record (EMR). Earlier this year, President Obama reinforced the federal government's commitment to that target, and announced that nearly \$20 billion in stimulus money would be available during the next five years to help health-care providers implement digital record systems.

EMRs are widespread in Europe, Australia, and elsewhere, but only 4% of American doctors have a fully functional system, according to the *New England Journal of Medicine*. Another 13% use a basic one. For patients, the benefits are obvious—convenience, portability, and efficiency. EMRs also offer benefits to health-care providers, including the reduction of clerical errors and computerized decision support. And scientists are excited by the technology's potential for furthering medical research.

For years, however, the economic incentives simply weren't powerful enough. Although studies suggest EMRs could save billions of dollars each year, the complexity and cost of developing, implementing, and managing the technology meant that American health-care providers—most of whom work in small practices with fewer than five physicians—found little reason to adopt it. "When a physician invests in EMR, 89% of the benefit goes to someone else," says Blackford Middleton, director of Clinical Informatics Research & Development at Partners HealthCare in Boston. Insurance companies, Middleton notes, are often the primary financial beneficiaries, saving money by streamlining the claims process and reducing duplicate tests and procedures. It's an imbalance



The Spanish Peaks Regional Health Center, based in Walsenburg, CO, plans to make the transition to an electronic health records system before the end of 2010, which will eliminate the need for office assistants to wade through piles of paper-based patient records.

that the federal stimulus, which makes physicians eligible for up to \$64,000 in incentives (and hospitals for up to \$11 million) beginning in 2011, may help overcome. But EMRs also present a number of other challenges involving protocols and standards, privacy, and how physicians practice medicine.

Protocols and Standards

One of the main difficulties is the lack of protocols and standards. Several groups across the country have developed their own EMR systems, including Partners HealthCare, the Cleveland Clinic, and the Regenstrief Institute in Indianapolis. Hundreds of EMR vendors offer products, too. But standards for the collection, exchange, and retrieval of electronic medical information vary widely from system to

system. The Healthcare Information Technology Standards Panel (HITSP), a private-public partnership, has worked since 2005 to harmonize protocols and standards, and the Obama administration has already convened two federal advisory committees to help tackle the problem. The trick, according to industry watchers, will be finding the right balance between standardization (which helps ensure interoperability and information sharing) and flexibility (which accommodates the various systems and architectures that different health-care providers need). Britain's National Health Service (NHS), for example, has fielded fierce criticism from clinicians about its plans to digitize the country's health system, which are derided as rigid and inadequate.

Privacy presents another formidable challenge in terms of how to keep patient data from falling into the wrong hands. In the U.S., health-care providers are subject to a set of stringent laws and regulations intended to keep patient data safe, most notably the Health Insurance Portability and Accountability Act (HIPAA). Other regulations vary from state to state, however, and it can be difficult to design EMR systems that comply with all the required policies. "Lawyers write these policies, and no one ever looks at them," says Annie Antón, a professor of computer science at North Carolina State University. "It's a huge problem." Formalizing rules that were intended to be interpreted by courts, rather than by programmers, is also challenging. Whereas European Union law grants its citizens ownership of their medical data—and thus a great deal of control about what happens to it—U.S. laws are less clear-cut. HIPAA doesn't address data ownership, and though it gives patients access to their records in most situations, it also grants doctors plenty of authority such as the right to share data with insurance companies, for example, and with other medical specialists.

A more basic privacy-related challenge stems from the complexity of health-care delivery. How is it determined which providers can access patient data? In general, only individuals with a direct medical need should be able to access files. Yet in an emergency-room setting, dozens of people may need to examine a single patient's record, from doctors to dieticians to interns—and speed is often of the essence. At Partners HealthCare's network of hospitals, emergency-room doctors can only access the records of patients who have already visited that hospital, and only view (but not modify) them. "It's an imperfect system," admits Cynthia Bero, CIO at Partners Community Healthcare, "but we have to balance providing the best care with preserving patient privacy." Other EMR systems enable hospitals to grant and revoke patient data access on a daily, and sometimes hourly, basis.

From a technological perspective, there are three different architectures: centralized models, where data is stored and held in a single database; federated models, where it is distributed across a

network; and a hybrid of the two. Each model has advantages and disadvantages. Centralized systems are generally regarded as more secure and easy to manage—and hence better for research purposes. Federated systems, however, may be easier for small practices to deploy, and to facilitate the aggregation of data across different networks. But they also force healthcare providers to rely on a third party to guard against data loss and breaches of privacy. It's not yet clear, experts say, exactly how architectures will evolve in the future.

Clinicians' Daily Workflow

To be truly useful, of course, EMRs must fit seamlessly into clinicians' daily workflow. Yet this, too, presents a challenge. From a physician's perspective, it's faster to scribble a prescription on a piece of paper than it is to log onto an EMR system and electronically enter the data. (Renewing a prescription, on the other hand, is much quicker with an EMR system.) HCI specialists like Jason Saleem, an investigator at the Regenstrief Institute, are therefore working to develop systems that are easier and more efficient to use. "By understanding doctors' workflow, we can design better EMRs," he explains. One lesson is that it may not be possible, or even desirable, for EMRs to fully replace paper. In a study published in the *International Journal of Medical Informatics*, Saleem observed that index cards and Post-It notes often serve as important extensions of EMRs when it comes to tasks such as remembering an impor-

Standards for the collection, exchange, and retrieval of electronic medical information vary widely from system to system.

Milestones

Computer Science Awards

The John D. and Catherine T. MacArthur Foundation and other organizations recently honored members of the computer science community for their innovative research.

MACARTHUR FOUNDATION "GENIUS" AWARD

Maneesh Agrawala, associate professor in the department of electrical engineering and computer sciences at the University of California, Berkeley, was among the 24 fellows for 2009 selected by the MacArthur Foundation. Agrawala will receive \$500,000 over the next five years. In announcing his selection, the foundation noted that Agrawala is working "at the intersection of visualization, human-computer interaction, and computers graphics" and is "designing visual interfaces that enhance our ability to understand large quantities of complex information."

NSF CAREER AWARD

National Science Foundation awarded Fengyan Li, assistant professor of mathematical sciences at Rensselaer Polytechnic Institute, a Faculty Early Career Development Award. Li will use the five-year, \$582,112 award to design, analyze, and implement computer algorithms for solving complex mathematical problems arising in sciences and engineering.

COMPUTER-AIDED VERIFICATION AWARD

The 2009 Computer-Aided Verification (CAV) award, which includes a \$10,000 prize and citation, was presented at the 21st annual CAV conference in Grenoble, France, to seven individuals who have made major advances in creating high-performance Boolean satisfiability solvers. They are Conor F. Madigan, Kateeva, Inc.; Sharad Malik, Princeton University; Joao P. Marques-Silva, University College Dublin, Ireland; Matthew W. Moskewicz, University of California, Berkeley; Karem A. Sakallah, University of Michigan; Lintao Zhang, Microsoft Research; and Ying Zhao, Wuxi Capital Group.

EMRs hold great potential for clinical-decision support by translating practice guidelines into automated reminders and actionable recommendations.

tant piece of information in the midst of juggling several other urgent tasks.

Saleem's work is complicated by the fact that the practice of medicine is highly individualized. Indeed, physicians in the same specialty often work differently even when performing the same medical procedure. Here standardization and flexibility must also be balanced. Too much standardization, and clinicians chafe under rules that don't match their own work habits. Too little, and workflow becomes less efficient. EMRs hold great potential for clinical-decision support, for example, by translating practice guidelines into automated reminders and actionable recommendations. Yet as Partners HealthCare CIO John Glaser explains, "You want to guide [clinicians] rather

than getting in the way." And striking the appropriate balance isn't always a straightforward task.

Technology, however, is only part of that equation. "It's really a much larger question that gets back to how we train our doctors," says Bero. Is medicine more art or science? Is good judgment more important than a rigid adherence to medical consensus? Though most people would agree that some variation in care is appropriate, the flexibility of EMR-encoded workflow depends on how you answer those questions. Economic incentives may also make a difference. Pay clinicians per procedure, as the U.S. health-care system typically does, and you give them little incentive to make workflow more streamlined or efficient.

As the U.S. debates these issues, the rest of the world is moving forward with its own initiatives. In Europe, EMR adoption rates are at 50% or higher in most countries, though as Middleton points out, that may be about more than just technology. "Where there are nationalized, centralized healthcare systems, there have typically been large investments in health IT," he says. The European Union is now trying to implement EMR standards that would enable country-to-country data exchange.

Meanwhile, thanks to a combination of frugal entrepreneurship and a more liberal approach to regulations, countries like India and Thailand have in many ways surpassed their rich-world counterparts when it comes

to health IT. Several Indian hospital chains use locally built EMRs, as does Bumrungrad hospital in Bangkok. And the rich world is beginning to take note. Apollo Health Street, an offshoot of India's Apollo Hospitals Group, sells HIT software to American hospitals, while Microsoft purchased software, intellectual property, and other assets from the Bangkok-based company that developed Bumrungrad's systems in 2007. □

Further Reading

Bates, D. and Gawande, A. Improving safety with information technology. *N Engl J Med* 348, 25, June 19, 2003, 2526–34.

Blumenthal, D. and Glaser, J. Information technology comes to medicine. *N Engl J Med* 356, 24, June 14, 2007, 2527–34.

Hillestad, R., Bigelow, J., Bower, A., Girosi, F., Meili, R., Scoville, R., Taylor, R. Can Electronic Medical Record Systems Transform Health Care? Potential Health Benefits, Savings, and Costs. *Health Affairs* 24, 5, 2005, 1103–1117.

Sittig, D.F., Wright, A., Osheroff, J.A., Middleton, B., Teich, J.M., Ash, J.S., Campbell, E., Bates, D.W. Grand challenges in clinical decision support. *J Biomed Inform* 41, 2, April 2008, 387–92.

Stead, W. and Lin, H. *Computational Technology for Effective Health Care: Immediate Steps and Strategic Directions*. The National Academies Press, Washington, D.C., 2009.

Leah Hoffmann is a Brooklyn-based science and technology writer.

© 2009 ACM 0001-0782/09/1100 \$10.00

Biology

Treating Human Disease

Peer Bork, a bioinformatician from the European Molecular Biology Laboratory in Heidelberg, Germany, has won the Royal Society and Académie des sciences Microsoft Award, one of the largest international prizes in science.

Bork won the €250,000 award, funded by Microsoft Research, for his work that focuses on discovering the important relations between the nature of the human microbiome—the union of all microorganisms that live in and around the human

body—and various parameters such as age, ethnic background, nutrition habits, and individual genetic components.

Due to improvements in technology, researchers are able to capture genomic information from microbes in tiny samples, such as pieces of skin. This has translated into an enormous amount of digital data stored in different databases, and Bork will use computational analysis to make sense of this vast amount of information and begin to draw relationships

between the different sets.

For example, when examining samples from humans with diarrhea, which causes one-fifth of child deaths worldwide, researchers might be able to find the species of microbe, which causes this disease. They hope to then develop an understanding of how to prevent or quickly treat it, perhaps with a yogurt containing other bacteria that can selectively reduce the harmful microbes.

Bork's earlier research uses computational analysis to mine

lists of unwanted side effects of any given drug for information that could help with possible new uses for the medication.

The Royal Society and Académie des sciences Microsoft Award was established to recognize outstanding contributions to science made by scientists working at the intersection of science and computing. The 2009 award was open to scientists working in Europe at the interface of the physical or biological sciences and computing.

Exploring New Frontiers

The Expeditions in Computing program provides scientists with the funding to work on ambitious, often multidisciplinary research.

EXPEDITIONS IN COMPUTING, the National Science Foundation's two-year-old program encouraging bold experimentation in computer science research, wasn't necessarily designed to showcase multidisciplinary projects. However, the seven winners of the \$10-million, five-year grants are in the vanguard of research in which profound advances in computer science often involve fundamental advances in other disciplines.

"A question that always arises when you give a big award like this is, 'Would it have been better to have given it to individuals instead of a big award?'" says Edmund M. Clarke, a professor of computer science and electrical engineering at Carnegie Mellon University and a recipient of a 2009 Expeditions grant. The answer, Clarke says, is that natural and social sciences are increasingly intertwined with computer science, and "no one of us is a master of all this material."

Jeannette M. Wing, assistant director of the Computer and Information Science and Engineering (CISE) directorate at the National Science Foundation, says Expeditions was designed in response to concerns from the computer science community that opportunities for expansive and visionary research has lagged in recent years. She says the program provides sufficient funding for a five-year period so the teams can focus on their research and not on writing grant proposals.

"It was really, really important for the CISE community, I felt," Wing says. "The expectation was that there would be multiple investigators working together in some kind of collaboration. It did not have to be interdisciplinary—that wasn't a requirement—but because we expected that a proposal would have multiple principal inves-



A concept drawing of a RoboBee, which belongs to an artificial colony of small-scale robotic bees, under development at Harvard's School of Engineering and Applied Sciences.

tigators, we wanted the whole to be greater than the sum of the parts."

The seven Expeditions projects—four were awarded in 2008 and three in 2009—bear out Wing's vision, with 27 universities and partner research organizations represented. Some projects tackle challenges within computer science itself, others bridge multiple disciplines, but all address a daunting challenge that, like an adventurous expedition, can stimulate one's imagination.

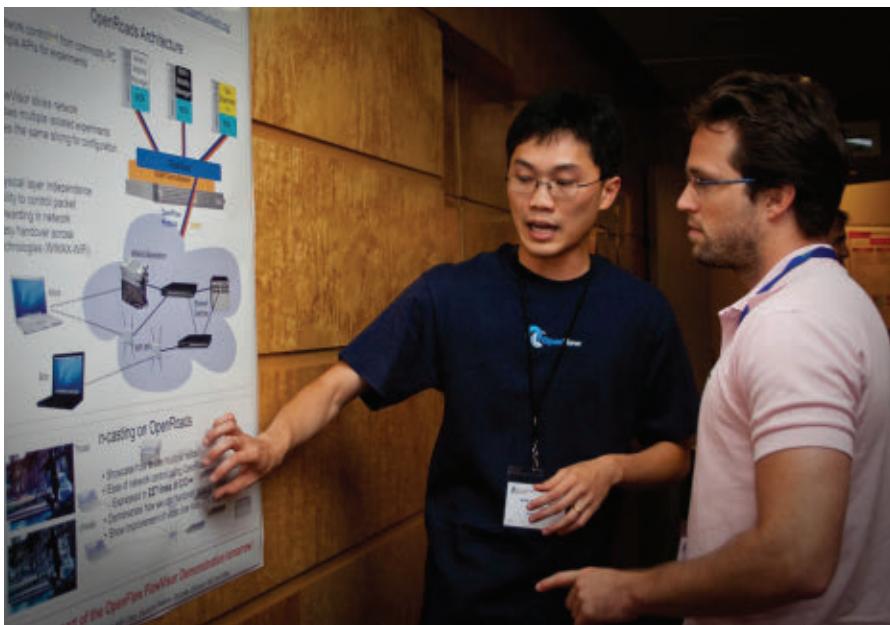
2009 AWARDEES

Combining Model Checking and Abstract Interpretation **Carnegie Mellon University**

The Carnegie Mellon project illustrates the symbiosis between critical issues in

computer science and other disciplines. Clarke and his colleagues will take principles in model checking and abstract interpretation and apply them to discovering new approaches for treatments of pancreatic cancer and atrial fibrillation, and also to better assure reliability in large-scale embedded systems such as those controlling aircraft functions.

Model checking considers every possible system design state against a designer's blueprint, and can warn of possible inconsistencies. Its granularity, however, limits the size of the systems it can analyze. Abstract interpretation, in contrast, develops an approximation of a system and preserves properties that need to be assessed. This makes it possible to analyze very large systems, but with less precision



Kok-Kiong Yap, left, a Ph.D. candidate in electrical engineering at Stanford University, explaining the award-winning poster "OpenRoads: Empowering Research in Mobile Networks" to an attendee at SIGCOMM 2009.

than model checking. In their Expeditions project, the researchers will take advantage of the strengths of both methods by tightly integrating the two into what they call MCAI 2.0.

"These domain challenges are at opposite ends of the spectrum of size," says Clarke, co-winner of the ACM A.M. Turing Award in 2007. (The 2007 Turing Award lecture, "Model Checking: Algorithmic Verification and Debugging," begins on p. 74.) "One is at the cellular level and the other is at the size of a 747 or Airbus A380—yet we believe that many of the same problems you need for handling complexity at one level will apply to the other as well."

Domain-Specific Computing University of California, Los Angeles

Jason Cong, professor of computer science and engineering at UCLA's Center for Domain-Specific Computing, likens the computing platform his team is devising to a human brain.

"If you look to the evolution of humans in the last 5,000 to 10,000 years, I don't think the number of neurons in the brain has changed that much, nor has the firing speed of neurons; we're all wired in a very similar way," Cong says. "So, I believe a lot of progress is done through specialization."

Just as training in a specific discipline leads one person to perform a giv-

en task much more easily than someone who is not similarly trained, Cong's team will explore creating a computing platform with what he calls a "very flexible" processor core with customizable elements such as operating frequency, voltage, and cache sizes, as well as a domain-specific programming fabric and a radio frequency-based communications fabric that can be tuned to multiple applications. All of the customizable computing and communications elements will be managed by a stack of intelligent software.

The UCLA-led project will focus on adapting these platforms for medical imaging and hemodynamic simulation. A successful imaging domain-

The seven projects address a daunting challenge that, like an adventurous expedition, can stimulate one's imagination.

specific platform, for instance, could significantly reduce the time a patient must spend in a CT scan machine. Other computational approaches to making scanning and image processing more powerful, such as massive parallel processing by general-purpose computers, are highly inefficient, and cloud computing poses possible data access and privacy issues, Cong says.

RoboBees: Body, Brain, and Colony Harvard University

The RoboBee researchers will explore complementary elements of creating robotic bees using three vectors modeled on live insects—body, brain, and colony. Topics within the body research includes all aspects of flight apparatus, propulsion, and power systems. The brain experiments involve research on the electronic nervous system equivalent of a bee's brain, including circuits for sensing and decision making. Colony research entails communication and control algorithms that will enable performance beyond the capabilities of an individual. The research team includes experts in biology, computer science, electrical and mechanical engineering, and materials science.

"We are trying to develop micro-mechanical devices, power electronics, and low-power computing fabrics we need to instrument the brain, and to wrap it all together, this notion of developing algorithms and simulating a whole colony of individual agents," says co-principal investigator Gu-Yeon Wei, associate professor of electrical engineering at Harvard's School of Engineering and Applied Sciences.

The researchers will also create an exhibit at the Museum of Science in Boston, which will explore the life of bees and the technologies required to create RoboBees. "Bees elicit a lot of excitement and imagery from young to old," Wei says. "We felt that a good way of tying it together is to share our research in an easily accessible manner."

2008 AWARDEES

Computational Sustainability:

**Computational Methods for a Sustainable Environment, Economy, and Society
Cornell University**

This Expedition applies techniques from computer science and related

fields, such as information science, operations research, applied mathematics, and statistics, to help manage the balance between environmental, economic, and societal needs for a sustainable future, says Carla Gomes, associate professor of computer science at Cornell University.

"The focus of computational sustainability is on developing computational and mathematical models, methods, and tools for decision making and policy making concerning the management and allocation of resources for sustainable development," Gomes says.

Gomes is also the director of Cornell's Institute for Computational Sustainability, which has launched a variety of interdisciplinary projects, such as wildlife conservation through nature reserve selection and corridor design, exploring the impact of renewable energy sources and climate change, policies for fishery management and fish preservation, and equilibrium models for bio-fuel policies.

The institute has fostered the establishment of a research community around this new field by organizing the First International Conference on Computational Sustainability, held in June 2009. It brought together 220 scientists, policymakers, and students in more than 15 disciplines from around the world. (For more about the Cornell conference, see "Computer Science Meets Environmental Science" on p. 23 of the Sept. 2009 issue of *Communications*.)

Programmable Open Mobile Internet (POMI) Stanford University

The Stanford Expedition aims to create an open Internet for ubiquitous mobile computing and communication networks, thereby increasing users' choices in the way their data and information is computed, communicated, and stored.

Nick McKeown, associate professor of computer science and electrical engineering at Stanford, says POMI is tackling what he calls the three main barriers to ubiquitous open networks and access. First, "private data is increasingly held by portals that capture our data and restrict the applications we can run against it; in response we

The Stanford project aims to create an open Internet for mobile computing and communications networks.

are developing the PRPL (PRivate-PubLIC) data system," says McKeown. Second, "there is abundant wireless capacity around us, but most is closed because of closed contracts; in response, we are developing OpenRoads as an experimental platform to allow network service to be separated from the underlying infrastructure." Lastly, "the current wireline and wireless network is ossified, and innovation in the infrastructure moves at a glacial pace. In response, we are deploying OpenFlow networks on college campuses to facilitate more rapid innovation."

As mobile standards and infrastructure evolve, POMI is an opportunity to develop an open architecture that will create an environment well suited for innovation, competition, and user experience.

Molecular Programming Project California Institute of Technology

The biomolecular programs of life, from the low-level operating system controlling cell metabolism to the high-level code for development—the process by which a single cell becomes an entire organism—serve as the key inspiration for this Caltech Expedition.

"The team aims to create analogous molecular programs using non-living chemistry in which computing and decision making will carried out by chemical processes themselves," says Erik Winfree, associate professor of computer science, computation and neural systems, and bioengineering at Caltech. The Expeditions work can help create a new subdiscipline of computer science, says Winfree, "that

will enable a yet-to-be imagined array of applications from chemical circuitry for interacting with biological molecules to molecular robotics and nanoscale computing."

The Expedition's first year has witnessed significant advances in all these areas, says Winfree. "As an example, we have developed a compiler for translating abstract formal chemical reaction equations into systems of DNA molecules that react with equivalent dynamics, and we have begun experimental investigations of small circuits generated this way," he notes.

Understanding, Coping With, and Benefiting From Intractability

Princeton University

Sanjeev Arora, professor of computer science at Princeton, says the Center for Computational Intractability, which was created out of this Expeditions grant, "seeks to bridge fundamental gaps in our understanding about the power and limits of efficient algorithms, which has the potential to revolutionize our understanding of algorithmic processes in a host of disciplines, and cast new light on fields such as quantum computing, secure cryptography, and pseudorandomness."

A key area of interest for the Princeton researchers is the amorphous boundary between problems considered tractable and those considered intractable; an example of this is reliance on cryptographic programs considered intractable, but which may be tractable and are possibly vulnerable to attacks.

In its first year, the Princeton team has carried out monthly day-long meetings focusing on finding new approaches to open problems of intractability, and via workshops, "which have been wildly popular," and visiting positions, has sought to actively engage other researchers, says Arora. It has also performed outreach activities such as popular talks and a one-month mini-course on computational complexity for high school students in the New Jersey Governor's School program. □

Gregory Goth is an Oakville, CT-based writer who specializes in science and technology.

CALL FOR PARTICIPATION

CTS 2010

Chicago, Illinois, USA



The 2010 International Symposium on Collaborative Technologies and Systems

May 17 – 21, 2010
The Westin Lombard Yorktown Center Hotel
Chicago, Illinois, USA

Important Dates:

Paper Submission Deadline -----	December 20, 2009
Workshop/Special Session Proposal Deadline -----	November 15, 2009
Tutorial/Demo/Panel Proposal Deadline -----	January 8, 2010
Notification of Acceptance -----	February 1, 2010
Final Papers Due -----	March 1, 2010

For more information, visit the CTS 2010 web site at:
<http://cisedu.us/cis/cts/10/main/callForPapers.jsp>



In cooperation with the ACM, IEEE, IFIP

DOI:10.1145/1592761.1592773

Butler Lampson

Privacy and Security Usable Security: How to Get It

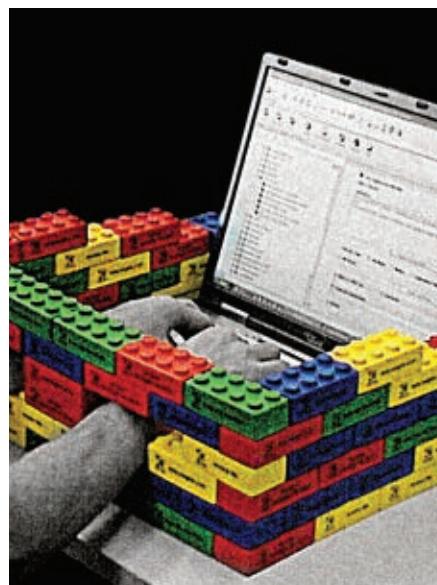
Why does your computer bother you so much about security, but still isn't secure? It's because users don't have a model for security, or a simple way to keep important things safe.

COMPUTER SECURITY TODAY is in bad shape: people worry about it a lot and spend a good deal of money on it, but most systems are insecure.

Security is not about perfection. In principle we can make secure software and set it up correctly, but in practice we can't, for two reasons:

- **Bugs:** Secure systems are complicated, hence imperfect. Of course software always has bugs, but even worse, security must be set up: user accounts and passwords, access control on resources, and trust relationships between organizations. In a world of legacy systems, networked computers, mobile code, and changing relationships between organizations, setup is error-prone.

- **Conflicts:** Even more important, security gets in the way of other things you want. In the words of General B.W. Chidlaw, "If you want security, you must be prepared for inconvenience."^a For users and administrators, security



adds hassle and blocks progress. For software developers, it interferes with features and with time to market.

To make things worse, security is fractal: Each part is as complex as the whole, and there are always more things to worry about. Security experts always have a plausible scenario that demands a new option, and a plausible threat that

demands a new defense. There's no resting place on the road to perfection.

Security is really about risk management: balancing the loss from breaches against the costs of security. Unfortunately, both are difficult to measure. Loss is the chance of security breaches times the expense of dealing with them. Cost is partly in dollars budgeted for firewalls, software, and help desks but mostly in the time users spend typing and resetting passwords, responding to warnings, finding workarounds so they can do their jobs, and so forth. Usually all of these factors are unknown, and people seldom even try to estimate them.

More broadly, security is about economics.² Users, administrators, organizations, and vendors respond to the incentives they perceive. Users just want to get their work done; they don't have good reasons to value security, and view it as a burden. If it's hard or opaque, they will ignore it or work around it; given today's poor usability they are probably doing the right thing. If you force them, less useful work will get done.¹ Tight security

^a Chidlaw, B. Dec. 12, 1954. Quoted by the International Spy Museum, Washington D.C.

usually leads first to paralysis and then to weak security, which no one complains about until there is a crisis.

Administrators want to prevent obvious security breaches, and avoid blame if something does go wrong. Organizations want to manage their risk sensibly, but because they don't know the important parameters they can't make good decisions or explain their policies to users, and tend to oscillate between too much security and too little. They don't measure the cost of the time users spend on security and therefore don't demand usable security. Vendors thus have no incentive to supply it; a vendor's main goal is to avoid bad publicity.

Operationally, security is about *policy* and *isolation*. Policy is the statement of what behavior is allowed: for example, only particular users can approve expense reports for their direct reports or only certain programs should run. Isolation ensures the policy is always applied. Usability is pretty bad for both.

Policy

Policy is what users and administrators see and set. The main reason we don't have usable security is that users don't have a model of security they can understand. Without such a model, the users' view of security is just a matter of learning which buttons to push in some annoying dialog boxes, and it's not surprising they don't take it seriously and can't remember what to do. The most common user model today is "Say OK to any question about security."

What do we want from a user model?

- ▶ It has to be simple (with room for elaboration on demand).
- ▶ It has to minimize hassle for the user, at least most of the time.
- ▶ It has to be true (given some assumptions). It is just as real as the system's code; terms like "user illusion" make as much sense as saying that bytes in RAM are an illusion over the reality of electrons in silicon.
- ▶ It does *not* have to reflect the implementation directly, although it does have to map to things the code can deal with.

An example of a successful user model is the desktop, folders, and files of today's client operating systems. Although there is no formal standard for this model, it is clear enough that users can easily move among PC, Macintosh, and Unix systems.

The standard *technical* model for security is the access control model shown in the figure, in which isolation ensures there is no way to get to objects except through channels guarded by policy, which decides what things agents (principals) are allowed to do with objects (resources). Authentication identifies the principal, authorization protects the resource, and auditing records what happens; these are the gold standard for security.³ Recovery is not shown; it fixes damaged data by some kind of undo, such as restoring an old version.

In most systems the implementation follows this model closely, but it is not very useful for ordinary people: they take isolation for granted, and they don't think in terms of objects or resources. We need models that are good for users, and that can be com-

piled into access control policy on the underlying objects.

A user model for security deals with policy and history. It has a vocabulary of objects and actions (nouns and verbs) for talking about what happens. History is what *did* happen; it's needed for recovering from past problems and learning how to prevent future ones. Policy is what *should* happen, in the form of some general rules plus a few exceptions. The policy must be small enough that you can easily look at all of it.

Today, we have no adequate user models for security and no clear idea of how to get them. There's not even agreement on whether we can elicit models from what users already know, or need to invent and promote new ones. It will take the combined efforts of security experts, economists, and cognitive scientists to make progress. Here are a few tentative examples of what might work.

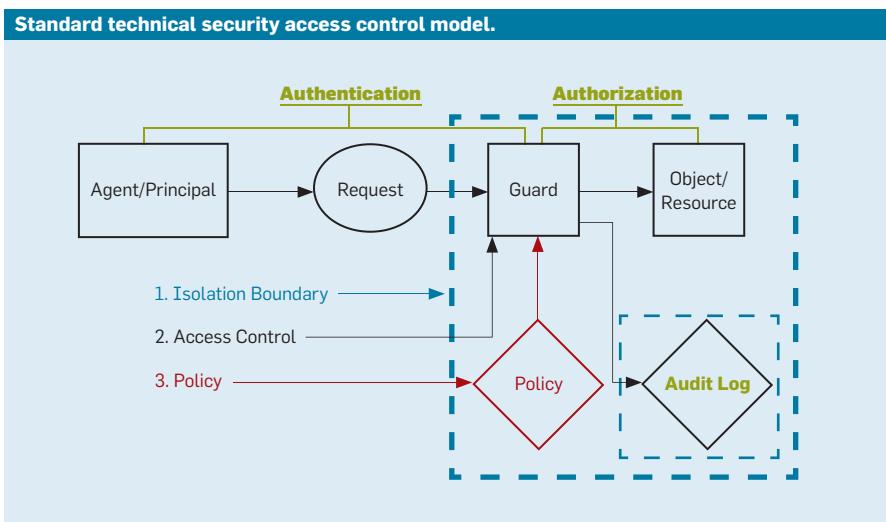
You need to know **who** can do **what** to **which** things. "Who" is a particular person, a group of people like your Facebook friends, anyone with some attribute like "over 13," or any program with some attribute like "approved by Microsoft IT." "What" is an action like read or write. "Which" is everything in a particular place like your public folder, or everything labeled medical stuff (implying that data can be labeled). An administrator also needs declarative policy like, "Any account's owner can transfer cash out."

A time machine lets you recover from damage to your data: you can see what the state was at midnight on any previous day. You can't change the past, but you can copy things from there to the current state just as you can copy things from a backup disk.

Isolation

Perfect isolation ensures that the only way for an input to reach an object is through a channel controlled by policy. Isolation fails when an input has an effect that is not controlled by policy; this is a bug. Some common bugs today are buffer overruns, cross-site scripting, and SQL code injection. Executable inputs like machine instructions or JavaScript are obviously dangerous, but modern HTML is so complex and expressive that there are many ways

Standard technical security access control model.



to trick a browser into running code, and widely used programs with simple inputs like JPEG have had buffer overruns. A modern client OS, together with the many applications that run on it, is bound to have security bugs.

Users can't evaluate these dangers. The only sure way to avoid the effects of dangerous inputs is to reject them. A computer that is not connected to any network rejects all inputs, and is probably secure enough for most purposes. Unfortunately, it isn't very useful. A more plausible approach has two components:

- Divide inputs into safe ones, handled by software that you trust to be bug-free (that is, to enforce security policy), and dangerous ones, for which you lack such confidence. Vanilla ANSI text files are probably safe and unfiltered HTML is dangerous; cases in between require judgments that balance risk against inconvenience.

- Accept dangerous inputs only from sources that are accountable enough, that is, that can be punished if they misbehave. Then if the input turns out to be harmful, you can take appropriate revenge on its source.

Accountability

People think that security in the real world is based on locks. In fact, real-world security depends mainly on deterrence, and hence on the possibility of punishment. The reason your house is not burgled is not that the burglar can't get through the lock on the front door; rather, it's that the chance of getting caught and sent to jail, while small, is large enough to make burglary uneconomic.

It is difficult to deter attacks on a computer connected to the Internet because it is difficult to find the bad guys. The way to fix this is to communicate only with parties that are accountable, that you can punish. There are many different punishments: money fines, ostracism from some community, firing, jail, and other options. Often it is enough if you can undo an action; this is the financial system's main tool for security.

Some punishments require identifying the responsible party in the physical world, but others do not. For example, to deter spam, reject email unless it is signed by someone you know or comes with "optional postage" in the form

The most common user model today is "Say OK to any question about security."

of a link certified by a third party you trust, such as Amazon or the U.S. Postal Service; if you click the link, the sender contributes a dollar to a charity.

The choice of safe inputs and the choice of accountable sources are both made by *your* system, not by any centralized authority. These choices will often depend on information from third parties about identity, reputation, and so forth, but which parties to trust is also your choice. *All trust is local.*

To be practical, accountability needs an ecosystem that makes it easy for senders to become accountable and for receivers to demand it. If there are just two parties they can get to know each other in person and exchange signing keys. Because this doesn't scale, we also need third parties that can certify identities or attributes, as they do today for cryptographic keys. This need not hurt anonymity unduly, since the third parties can preserve it except when there is trouble, or accept bonds posted in anonymous cash.

This scheme is a form of access control: you accept input from me only if I am accountable. There is a big practical difference, though, because accountability is for punishment or undo. Auditing is crucial, to establish a chain of evidence, but very permissive access control is OK because you can deal with misbehavior after the fact rather than preventing it up front.

Freedom

The obvious problem with accountability is that you often want to communicate with parties you don't know much about, such as unknown vendors or gambling sites. To reconcile accountability with the freedom to go anywhere on the Internet, you need two (or more) separate machines: a

green machine that demands accountability, and a red one that does not.

On the green machine you keep important things, such as personal, family and work data, backup files, and so forth. It needs automated management to handle the details of accountability for software and Web sites, but you choose the manager and decide how high to set the bar: like your house, or like a bank vault. Of course the green machine is not perfectly secure—no practical machine can be—but it is far more secure than what you have today.

On the red machine you live wild and free. You don't put anything there that you really care about keeping secret or really don't want to lose. If anything goes wrong, you reset the red machine to some known state.

This scheme has significant unsolved problems. Virtual machines can keep green isolated from red, though there are details to work out. However, we don't know how to give the user some control over the flow of information between green and red without losing too much security.

Conclusion

Things are so bad for usable security that we need to give up on perfection and focus on essentials. The root cause of the problem is economics: we don't know the costs either of getting security or of not having it, so users quite rationally don't care much about it. Therefore, vendors have no incentive to make security usable.

To fix this we need to measure the cost of security, and especially the time users spend on it. We need simple models of security that users can understand. To make systems trustworthy we need accountability, and to preserve freedom we need separate green and red machines that protect things you really care about from the wild Internet. □

References

1. Adams, A. and Sasse, A. Users are not the enemy. *Commun. ACM* 42, 12 (Dec. 1999), 41–46.
2. Anderson, R. Economics and Security Resource Page; <http://www.cl.cam.ac.uk/~rja14/econsec.html>
3. Lampson, B. Practical principles for computer security. In *Software System Reliability and Security*, Broy et al., Eds., IOS Press, 2007, 151–195.

Butler Lampson (ButlerLampson@microsoft.com) is a Technical Fellow at Microsoft Research and is an ACM Fellow.

Copyright held by author.



DOI:10.1145/1592761.1592772

Pamela Samuelson

Legally Speaking

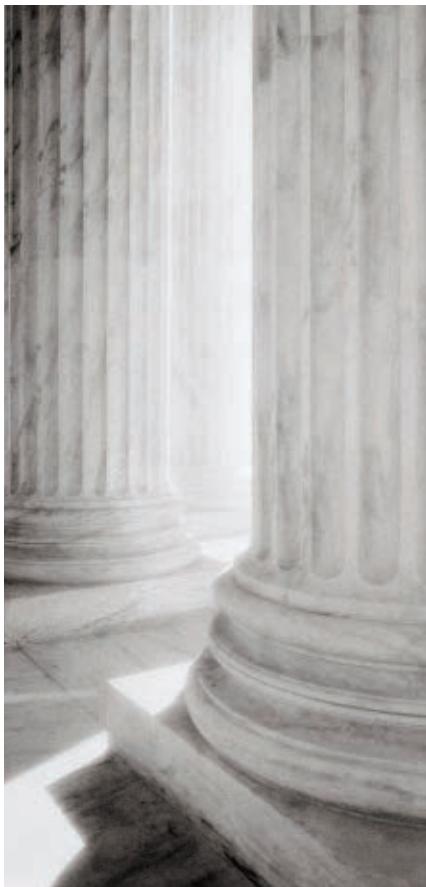
Are Business Methods Patentable?

*How the U.S. Supreme Court's forthcoming decision in the *Bilski v. Doll* case is expected to affect existing and future software patents.*

THE PATENTABILITY OF software and business methods has been a contentious topic for more than 50 years. Because the U.S. Supreme Court decided to review the patentability of business methods in *Bilski v. Doll* in the fall of 2009, this controversy will once again—and soon—reach a boiling point. *Bilski* is asking the Supreme Court to reverse decisions denying him a patent on a three-step method of hedging the risk of price fluctuations in commodities.

Although the *Bilski* case, strictly speaking, does not involve software patents, the Court is likely to say some things in *Bilski* that will have implications for the patentability of software innovations. This is partly due to the fact that *Bilski*'s method could be carried out with the aid of a programmed computer, but also because most of the Court's prior rulings on patent subject matter involve computer program innovations and many software patents are for software-implemented business methods.

The Court's last pronouncement on patent subject matter was its 1981 *Diamond v. Diehr* decision in which the Court ruled by a 5–4 majority that a rubber curing process that included computer program calculations was patentable subject matter. Many software companies, industry associations, and programmers will be weighing in with briefs in *Bilski* arguing for and against software patents.



This column will first briefly review the *State Street Bank & Trust Co. v. Signature Financial Services, Inc.* decision rendered by the U.S. Court of Appeals for the Federal Circuit (CAFC) in 1998 that opened the door to business method patents.

After the Supreme Court in 2006 began to signal its dissatisfaction with

the *State Street* test and with business method patents, the CAFC decided to reconsider the broad conception of patentable subject matter articulated in *State Street* and in October 2008, it rejected *Bilski*'s business method claim.

Rather than giving the CAFC leeway to develop its own post-*Bilski* jurisprudence, the Court has selected *Bilski* as its preferred vehicle for making a new pronouncement about patentable subject matter.

I predict the Court will rule that *Bilski*'s method is unpatentable, although for somewhat different reasons than most CAFC judges gave. The Court's pronouncement in *Bilski* will almost inevitably have direct implications for existing and future software patents. This may affect the behavior of both start-up and established software companies, venture capitalists, investors, and many others.

***State Street* on Patent Subject Matter**

Signature is a financial services company that in 1993 obtained a patent on an automated data processing system that used a hub-and-spoke structure to organize financial services. After negotiations over a license to use this patent broke down, State Street sought a court declaration that Signature's patent was invalid because it claimed a business method, which many prior decisions had deemed to be unpatentable subject matter.

In upholding Signature's patent,

the CAFC characterized the so-called business method exception to patentable subject matter as “ill-conceived.” The CAFC said that “anything under the sun made by man” was patentable subject matter, and any process could be patented as long as it yielded a “useful, concrete and tangible result,” as Signature’s did.

The *State Street* ruling has been controversial for years, in part because it contradicted numerous previous decisions, which critics of *State Street* thought had properly rejected business method patents. Yet, because the Supreme Court decided not to review the *State Street* ruling, some inferred that the Court had acquiesced in the CAFC’s expansive interpretation of patent subject matter.

Tens of thousands of business method and other nontechnological patent applications were filed in reliance on *State Street*, and many were granted.

Signals from the Supreme Court

Starting in 2006, the Supreme Court began signaling its dissatisfaction with the CAFC’s conception of patentable subject matter. One opinion criticized business method patents as vague and poor in quality. A second pointed out that the Court had never endorsed the *State Street* test for patent subject matter and questioned the patentability of methods that could be infringed by thinking (i.e., mental processes). During oral argument in a third case that involved a software patent, several Justices asked questions about the patentability of software, even though that issue was not really before the Court.

The U.S. Patent and Trademark Office (PTO) inferred from these signals that the Court did not agree with the *State Street* test. The PTO then began denying applications for business methods and other nontechnological patents, including Bilski’s application for a patent for a three-step method of hedging risks of price fluctuations as to commodities.

The *In Re Bilski* Case

The CAFC decided to hear Bilski’s appeal en banc (that is, before the full court, not just the usual three-judge panel). It also invited interested parties to submit briefs about whether

Tens of thousands of business methods and other nontechnological patent applications were filed in reliance on *State Street*, and many were granted.

the court should reconsider the *State Street* test.

In October 2008, nine of the 12 CAFC judges agreed that Bilski’s method was unpatentable because it was neither “tied to a machine,” nor did it “transform” anything from one state to another, as they thought Supreme Court precedents required.

Yet, two of the nine judges, along with another colleague, wrote separately to say that business methods are ineligible for patents. Two other judges dissented, objecting to the majority’s restrictive interpretation of patent subject matter.

Business Method Patents Critics

The CAFC critics of business method patents asserted that the historical record showed that business methods such as Bilski’s were not patentable subject matter. Prior to 1793, a well-established English practice restricted patents to manufacturing processes. This “reflects the understanding that only processes related to manufacturing or ‘manufactures’ were within the statute.”

This rule was carried over into U.S. law in the 1793 patent act. During the 19th and all but the last two years of the 20th century, U.S. patents were consistently limited to technological processes. There being no evidence that Congress ever decided to expand patent subject matter to include business methods, there was no basis on which to award Bilski or anyone else a patent on a business method, no matter how innovative it might be.

There was, moreover, an unbroken—until *State Street*—line of pre-

cedents going back to the 19th century that considered business methods to be unpatentable.

Judge Mayer’s opinion in *Bilski* took the strongest position against business method patents. In his view, “[t]he patent system is intended to protect and promote advances in science and technology, not ideas about how to structure commercial transactions.” Patents on business methods such as Bilski’s “lack[] constitutional and statutory support, serve[] to hinder rather than promote innovation, and usurp[] that which rightfully belongs in the public domain.”

Nor are patents necessary to bring about innovative business methods, for “by their very nature, [such new methods] provide a competitive advantage and thus generate their own incentives.” The PTO was being distracted from examining technology patents because *State Street* had led to a flood of applications for nontechnological processes. In the heyday of the Internet bubble, many entrepreneurs sought patents on methods of conducting business on the Internet.

Proponents of Broad Subject Matter

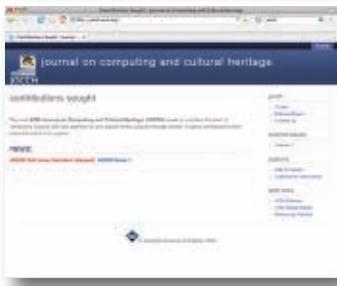
The dissenting CAFC judges thought it was a mistake to freeze the concept of patentable subject matter to that which was appropriate to the “age of iron and steel,” as they thought the majority’s “machine-or-transformation” test would do. This test might exclude innovative processes involving “subatomic particles and terabytes.”

The dissenters also worried that the majority’s restrictive view of patentable processes would undermine incentives to invest in innovation in new fields and would put at risk the position that the U.S. has enjoyed as “the world’s innovation leader.” Many investments had, moreover, been made in reliance on *State Street*’s test, so it was troubling to “disrupt the settled expectations of those who relied on the law as it existed” under *State Street*. Only one of the dissenters, however, stood up for the *State Street* “useful, concrete, and tangible result” test, saying that it had proven workable and should be retained.

What Will the Supreme Court Do?

Chances are high that the Supreme

ACM Journal on Computing and Cultural Heritage



JOCCH publishes papers of significant and lasting value in all areas relating to the use of ICT in support of Cultural Heritage, seeking to combine the best of computing science with real attention to any aspect of the cultural heritage sector.



www.acm.org/jocch
www.acm.org/subscribe



Association for
Computing Machinery

In the heyday of the Internet bubble, many entrepreneurs sought patents on methods of conducting business on the Internet.

Court will rule in *Bilski* that business methods are unpatentable. The Court typically finds historical evidence and decades of precedents persuasive. Both of these factors cut against the patentability of business methods.

The structure and purpose of the patent act also suggest that patents should only protect technological processes. Section 101 allows patents to issue for "new and useful machines, manufactures, compositions of matter, and processes." The constitutional purpose underlying this provision is "to promote the progress of...useful Arts," which in today's parlance is understood to mean the technological arts. The Court will probably also discuss some policy reasons for limiting patents to technologies.

The Court will probably be unanimous in striking down business method patents. Although the opinion written by one of the Justices for the Court will likely focus only on the unpatentability of business methods, there may well be at least one concurring opinion that takes a stronger and broader stance against nontechnological patents.

It is highly unlikely that the Court will endorse the *State Street* standard in part because it requires reading four words into the statute that aren't there (the "useful, concrete, and tangible result" part of the test). This test is, moreover, normatively unappealing and has caused the PTO to waste resources examining nontechnological claims.

Nor is the Court likely to endorse the CAFC's "machine-or-transformation" test because that test seems to conflict with the Court's 1972 decision in *Gottschalk v. Benson*. Benson sought a patent for a novel method of

transforming binary coded decimals to pure binary form. One claim contemplated carrying out this method in a programmed computer; another claim omitted references to computer technology. The Court ruled that this algorithm was ineligible for patent protection because it was an abstract mathematical idea. The Court did not distinguish between the claim that mentioned re-entrant shift registers and the one that didn't.

Unless the Court is ready to overturn *Benson*, it may say that Bilski's business method should not become patentable if Bilski simply mentions that the method could be carried out with the aid of a programmed computer or that when embedded in software, the method would transform data.

Implications of *Bilski* for Software Patents

In explaining why Bilski's method is unpatentable, the Court will almost certainly have to construe *Benson* and *Diehr*. Thus, its *Bilski* decision will almost inevitably reopen a host of questions about the patentability of software innovations.

Software innovations have been troublesome for patent law in part because they often, like Benson's algorithm, are mathematical ideas that can be carried out in a human mind or with the aid of a pen and paper, as well as by computer, and in part because they often automate business processes, which the Court is likely to rule are unpatentable.

Because the code that actually implements innovative software processes is protected by copyright law, some confusion exists about the respective roles of copyright and patent in protecting computer program innovations.

Although the Court may find it easy to reject Bilski's patent as nontechnological, it is a difficult task to develop a definitive test for judging which processes are and are not "technological." Are Web services or XML schemas, for example, technological processes?

Bilski will not definitely answer these questions, but it will give new life to this decades-old debate. C

Pamela Samuelson (pam@law.berkeley.edu) is the Richard M. Sherman Distinguished Professor of Law and Information at the University of California, Berkeley.

Copyright held by author.

Economic and Business Dimensions The Broadband Price is Not Right

Developing an effective pricing index is essential to understanding the value of broadband connectivity.

HOW MUCH SHOULD subscribers pay for broadband? Without a sound economic benchmark, it is difficult to tell if the old expression "whatever the market will bear" is too high. We need a good benchmark, but we do not have one.

Why do we need one? Consider recent experience. In September 2001, approximately 45 million U.S. households accessed the Internet through a dial-up connection, while only 10 million used a broadband connection. By March 2006, the situation was moving to the opposite: Approximately 47 million households (and growing) had broadband connections, while 34 million (and declining) used dial-up connections. According to the latest survey of the Pew Internet and American Life Project, in April, 2009, less than 10% of U.S. households had dial-up Internet connections, and 63% of U.S. households had broadband.

What happened to the price of broadband during and after that deployment? In our study, "The Broadband Bonus: Accounting for Broadband Internet's Impact on U.S. GDP," <http://www.nber.org/papers/w14758>, Ryan McDevitt and I inquired into broadband's value. We approached this question from a novel angle. We asked: "How fast would prices have to come down to reflect the



value created from the replacement of dial-up access with broadband?"

What was so novel about that question? While answering that question we strictly followed the standard procedures used by the U.S. Bureau of Labor Statistics (BLS) to estimate the Consumer Price Index (CPI).

We had several reasons for doing this. First, the consumer price index is the primary measure of inflation in the U.S.—for example, the Social Security Administration uses it to adjust its checks. However, economists have long suspected the CPI index contains biases. Many years ago those suspi-

cions were confirmed in integrated circuits and personal computers, as well as in cellular telephony. Accordingly, measurement procedures in those markets changed.

Should broadband change too? It is a big and open question because nobody has yet estimated the size of the bias (if any). Governments in the developed world use procedures similar to those used in the U.S. If U.S. statistics contain a bias, it is also likely in other countries.

A big policy issue also motivated us. Price indices can measure improvements (or not) in competitive perfor-

mance in markets with few suppliers, as in broadband. Yet, a policymaker cannot address questions about pricing without understanding whether the CPI measures prices accurately.

Measuring Technical Improvement

As it does elsewhere, BLS uses very cautious procedures for recording benefits from upgrading to broadband. There is a rationale behind that caution: Not all users experience the benefits of broadband in the same way. For example, capacity/bandwidth differs between households and the location in the national grid matters. Accordingly, rather than merely assuming households benefit from the availability of a new service, such as broadband, the BLS waits for clear evidence that buyers like the improvements.

This approach leads to what is sometimes called a “transactional” index. Standard procedures do not measure technical progress at the frontier unless users transact for a better good or service. Moreover, the CPI focuses on the average experience, getting an average price by taking a weighted average over all transactions, including those not at the frontier.

That is why, for example, wireless broadband tended not to play a big role in the official price index until after 2006. Wireless broadband revenue was small in comparison to the revenue for wireline broadband. Incidentally, that is also why McDevitt and I focused our study on wireline broadband, the bulk of transactions for which there is public data up until 2006.

The CPI for Internet access is officially called Internet services and electronic information providers, and the

BLS began compiling data in December 1997. BLS deserves some credit for starting this index not long after the diffusion of the Internet began. At that point approximately 20% of U.S. households had adopted dial-up commercial Internet service, and less than 1% had adopted broadband. The second and third rows of the table illustrate this point.

In the fourth row is a monthly quote from the official price index, taken from December of each year, and normalized to 100 for the year in which the index began. It indicates that the CPI for Internet access in the U.S. went mildly down and up and down for eight years. Then, in late 2006, it broke with all prior patterns and declined more than 18% from its base (i.e., $(94.5 - 77.2)/94.2 = 0.183$). That drop continued and settled at a 23% from its base in January 2007 (i.e., $(94.5 - 73.4)/94.5$), staying near that level ever since.

Did that last drop reflect increasing competitiveness of broadband markets? In fact, it does not. The dramatic drop in the official price index for the U.S. primarily reflects the pricing of the declining good, dial-up service. An informed reading of the business news explains why. The largest U.S. dial-up provider, America Online (AOL), altered its pricing policies in the fall of 2006. AOL announced it was moving to advertising-supported service. On top of that, BLS used its standard cautious procedures. That gave AOL's prices close to a quarter of the weight in the index (even though AOL's market share was dropping).

Diffusion and Prices

The official index is less informative about broadband prices for an addi-

tional but rather subtle reason. Standard price index survey procedures measure the price at which the new good transacted but not at the price that previously deterred the user from adoption. For example, in many neighborhoods broadband was not available in any form for some time until after 2000. Even when it became available, it may not have been reliable enough (initially) to spur many households to switch immediately from dial-up. These users waited until vendors improved the infrastructure or service arm of the organization. In short, users did not adopt until quality improved.

That common occurrence creates a problem for BLS. Household eventually converted from dial-up to broadband, as rows 2 and 3 of the table indicate. However, there was (seemingly) no measured price change or qualitative improvement affiliated with the upgrade.

What would have happened to the broadband price index if it reflected unmeasured qualitative improvement? That is a succinct way to understand one aim of our study. McDevitt and I followed the standard recommendation for this problem. It goes back to work by Sir John Hicks, a Nobel Prize Laureate in Economics, and has been used many times in economic analysis. The price index should employ what a user would have been willing to expend to get the new service before it was available.

Many surveys tell us what that willingness was. Even the most cautious surveys from the time period suggest the average convert was willing to pay approximately \$51.35 per month (on average), but actually had to pay less. If the actual price was \$36, for example,

Broadband benchmarks.

Year	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006
Internet HH (1)	19.1	27.2	35.5	44	53.8	56.7	59.5	66	73.3	81.8
Broadband HH (1)	n.a.(4)	n.a.(4)	0.9	3.2	9.6	13	18.5	27.5	41.1	47
Official Index (2)	100	103.3	96	95.7	100.3	99.6	97.2	94.5	77.2	73.1
Corrected Index (3)	100	103.3	95.4	94.1	95.9	93.8	89.5	84.1	65.6	61.1

(1) Millions of U.S. households.

(2) Quote from December of each year.

(3) Applies estimates from “The Broadband Bonus: Accounting for Broadband Internet’s Impact on U.S. GDP”, Table 7, row 2.

(4) Not available.

the upgrade provided a benefit “equivalent” to a decline in price of \$15.35, or a 30% decline.

The last row in the table illustrates our study’s estimate of a price index adjusted for upgrades. We follow the norms for a transactional index. Because only a small percentage of households a year upgraded to broadband, a price index that emphasizes transactions cannot give a huge weight to the upgrades in any given year. Upon initial assessment the gains appear modest even in our largest estimate, displayed in the last row of the table, around 2.2% decline per a year, each year prior by 2006.

Yet, it adds up over time. By 2006 the price index should have declined 16.4% more than it did ($(73.1 - 63.1)/73.1$). Our study experimented with many other different ways to count it. Depending on how it is counted, we estimate that somewhere between 4.8 and 6.7 billion dollars of benefit went uncounted in 2006 alone.

Notice one other feature. Our new estimate displays a big difference in the timing of the recorded price decline. Accounting for the upgrade when users upgraded would have realized the benefits several years sooner than BLS recorded in its official index.

Overall, our study suggests there are many unresolved issues in the price indices for a wide range of improving electronic goods where users realize discrete gains from upgrading to a new good. After all, broadband is far from the only improving service a user can upgrade. What was true of broadband is likely to be true for smartphones, Netbooks, digital camcorders, and even digital advertising. BLS has its hands full.

Policy

Our study also illustrates why a price index constructed for the CPI does not necessarily meet the needs of a policymaker interested in broadband policy, such as the Federal Communications Commission (FCC). We considered the gains from retirement of second phone lines at households. Once again, we experimented with numerous ways to estimate the gains from such retirement. We found that the savings on a second phone line accounts for anywhere from 30%–40% of the total savings in

How fast would prices have to come down to reflect the value created from the replacement of dial-up access with broadband?

2002–2006, or 21%–28% of the savings for 1999–2006. However, BLS price indices do not normally count the savings of expenditure in one category (on a second telephone line) as an input into calculating the price index for another (Internet access). While this procedural norm is fine for the CPI, it is misleading for broadband policy.

That would not have to be a problem if the FCC kept its own price index for its own purposes. However, the FCC has steadfastly refused to keep a broadband price index for many years, even while it tracks many other aspects of broadband. How can the FCC make informed competitive policy when it has to rely on the BLS for a price index that does not reflect the FCC’s policy priorities? Our study directs attention at this problem.

Indeed, the FCC has much to do. Its index would have to account for qualitative improvements, as well as the bundling of broadband pricing with other information services, such as cable television and telephony. The FCC also probably would want to measure changes to the frontier more quickly than BLS, watching prices of new access modes (smartphones), as well as the gains from Skype and other VoIP (using other phone services). It might be challenging to figure out the gains from Twitter over other text messaging, but that must be a component of the mix too. ■

Shane Greenstein (greenstein@kellogg.northwestern.edu) is The Elinor and Wendell Hobbs Professor in the Kellogg School of Management at Northwestern University, Evanston, IL.

Copyright held by author.

Calendar of Events

November 16–20
IEEE/ACM International Conference on Automated Software Engineering, Auckland, New Zealand, Sponsored: SIGART, SIGSOFT, Contact: John Grundy, Phone: 64-9-3737-599, Email: john-g@cs.auckland.ac.nz

November 17–19
Kuwait First e-Conference, Kuwait, Contact: Saleh Kassem, Email: saleh.kassem@yahoo.com

November 18–20
Asian Internet Engineering Conference, Bangkok, Thailand, Contact: Kanchana Kanchanasut, Email: kk@cs.ait.ac.th

November 22–25
8th International Conference on Mobile and Ubiquitous Multimedia, Cambridge, UK, Contact: Jonna Hakkila, Email: jonna.hakkila@nokia.com

November 23–24
NetGames ’09: Network and Systems Support for Games, Paris, France, Contact: Maha Abdallah, Email: maha.abdallah@lip6.fr

November 23–25
The ACM International Conference on Interactive Tabletops and Surfaces, Calgary, AB Canada, Contact: Gerald Morrison, Email: geraldm@smarttech.com

November 30–December 4
10th International Middleware Conference, Champaign, IL, Contact: Roy H. Campbell, Email: rhc@uiuc.edu

December 1–4
Conference on Emerging Network Experiments and Technologies, Rome, Italy, Sponsored: SIGCOMM, Contact: George Ventre, Phone: 39-081-7683908, Email: Giorgio@upina.it

Viewpoint

On Public Service and Computer Science

Members of the computer science community should become more involved in public service by becoming program managers at federal agencies, the opportunities and benefits of which are outlined here.

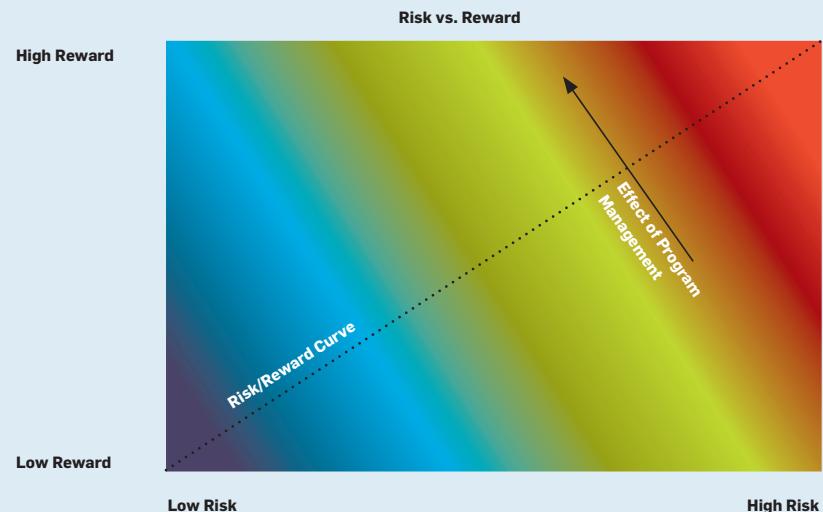
HERE HAS BEEN increasing concern about the availability of support for basic research in computer science.² Although there are always factors such as budget priorities and federal funds availability, based on my experience as a program manager at the U.S. Defense Advanced Projects Research Agency (DARPA) I believe there is an additional factor that is often missed. Neal Lane^a has spoken and written on related issues for science and technology in general; here, I focus on computer science.

The issue is the strength of the computer science representation “at the table” when priorities are decided and resource allocations are determined. The only way to change the resources allocated to computer science as a discipline is to ensure the best new ideas are represented passionately, effectively, and repeatedly. This means going beyond the advocacy role of representative organizations such as the Computing Research Association and voluntarily becoming public servants: to become agents for change.

What Can You Do?

Federal agencies such as DARPA and the U.S. National Science Foundation (NSF) are constantly looking for new ideas and people to make them hap-

Program management reduces risk, increases reward.



pen. Making ideas happen is the role of public servants such as DARPA program managers and NSF program directors, and these ideas are often far beyond what the risk/reward threshold of the private sector can support.

The figure here depicts a conceptual model of the effect that good program management can have on research risk and reward. The line labeled “Risk/Reward Curve” is meant to suggest there is a natural relationship between risk and reward, where low risk leads to low rewards and high risk is required to achieve the highest rewards. It is the goal of program

management to create an environment in which the most rewarding research is pursued with the least risk. The rewards are typically clear in the program manager’s vision, so it is risk that must be managed most aggressively.

Risks to manage include problem difficulty, researchers, time, and money. To manage problem difficulty, intermediate goals can be targeted, or multiple competitive approaches can be supported to reduce risk through a process akin to “portfolio management.” Researchers must be encouraged to be bold, but at the same time

^a See <http://www.nsf.gov/news/speeches/lane/nl91096.htm>

to be objective in deciding whether an approach may not be yielding results. Time and money are closely related and difficult to manage because breakthroughs are impossible to schedule, but competition for resources brings the estimates of other experts to bear on the questions of what results can be expected, how long it will take, and how much it will cost.

Therefore, in addition to vision and new ideas, effective program managers will bring deep expertise and credibility in their scientific discipline to their role in government. Depth is necessary to understand the real risks in a problem domain, to select the most promising researchers and plans, and to distinguish true technical innovation from rhetorical skill. Deep expertise and credibility require that program managers be drawn from the scientific community, while public service entails standing somewhat apart from the community, for fairness and to prevent real or perceived conflicts of interest. The perspective of a public servant drawn from the scientific community can be both challenging (since it requires a view at "field" level or even "national" level, rather than "specialty" level), and a tremendous opportunity for community and personal growth.

What Do You Get Out Of It?

One unfortunate, destructive, and derogatory myth about program management is that it is a purely administrative job, giving management briefings, rating proposals, and dunning people for annual reports. It is not. There are huge technical, professional, and personal rewards.

On the technical front, you are exposed to real problems, new technologies, and alternative solutions that you might never have imagined without the experience. A sample suggestion I got when I was looking for applications for my approaches to extreme networking environments was "read *Black Hawk Down*," which focused my attention on wireless communications and the problem-rich urban communications environment, leading to, for example, efforts in cognitive networking (SAPIENT, for Situation-Aware Protocols in Edge Network Technolo-

The perspective of a public servant drawn from the scientific community can be both challenging and a tremendous opportunity for community and personal growth.

gies⁴) and distributed radio (ACERT, for Adaptive Cognition-Enhanced Radio Teams^{1,5}). The broad exposure to other problems (for example, the traditional sciences, engineering disciplines such as aerospace, and emerging areas such as programmable matter) allows you to inject computer science thinking where it is necessary and shape new opportunities for advanced research.

The professional rewards accrue from doing a great job in public—you are never more visible than when you are in Washington, D.C., and your impact may persist for many years as a result of the programs you create and the people and teams you support. It is no surprise that your peers take notice and give credit where it is due. Further, the new skills you have gained catapult you into a new realm of mentoring and advisory roles, since they have direct bearing on the ability of any group to accomplish an agenda.

On the personal front, your service is the chance to set an agenda based on your new ideas, and support intellectual leadership for this agenda with significant resources. As an example, program managers at DARPA are hired based on their vision for change. Subtle changes can have surprising impact, for example, encouraging open source deliverables to stimulate software radio research.^{1,5} As an added benefit, you have access to a new form of publication, the research solicitation, which will be read far more closely

by proposers from your community than any conference or archival writing you do before or after your service. The solicitation itself can serve as an important tool in sparking new ways to approach problems.

Do You Have to Give Up Your Job?

Opportunities exist for academic researchers to perform public service while retaining their institutional affiliations, which of course demands extra care to avoid conflicts of interest. This can be accomplished, for example, under the Intergovernmental Personnel Act, which allows an agency to "borrow" staff members from the academic institution. Conflicts of interest, for example, would arise when the home institution is involved in a solicitation, and you must remove yourself from the decision-making path. Scholarly activities such as mentoring students that are near graduation can continue, albeit on a voluntary rather than a compensated basis.

Conclusion

Research really is an endless frontier,³ and the impact of our "general-purpose machines" on life and society has only just begun. I encourage you to consider public service as a way to keep the frontier spirit alive in the computer science community. □

References

1. Lau, R. et al. Cognitive adaptive radio teams. In *Proceedings of the 2nd International Workshop on Wireless Adhoc and Sensor Networks*, (IWWAN 2006) (New York, NY, 2006).
2. Lazowska, E.D. and Patterson, D.A. An endless frontier postponed. *Science* 308, 5723 (May 6, 2005).
3. *Science: The Endless Frontier: A Report to the President by Vannevar Bush, Director of the Office of Scientific Research and Development, July 1945*; <http://www.nsf.gov/od/lpa/nsf50/vbush1945.htm>
4. Smith, J.M. Cognitive techniques—Three types of network awareness. In B. Fette, *Cognitive Radio Technology*, 2nd ed., 2009.
5. Troxel, G.D. et al. Adaptive dynamic radio open-source intelligent team (ADROIT): Cognitively-controlled collaboration among SDR nodes. In *Proceedings of the First IEEE Workshop on Networking Technologies for Software Defined Radio (SDR) Networks* (Reston, VA, Sept. 25, 2006) (Held in conjunction with IEEE SECON 2006: Third Annual IEEE Communications Society Conference on Sensor, Mesh and Ad-Hoc Communications and Networks; invited paper).

Jonathan M. Smith (jms@cis.upenn.edu) is a professor of computer and information science at the University of Pennsylvania in Philadelphia. He was awarded the Office of the Secretary of Defense Medal for Exceptional Public Service in August 2006.

I wish to thank Ed Lazowska, Craig Partridge, and an anonymous reviewer for their wise suggestions.

Copyright held by author.

Interview

An Interview with Ping Fu

Ping Fu, CEO of the digital shape sampling and processing company Geomagic, discusses her background, achievements, and challenges managing a company during a period of dynamic growth.

PING FU, CHAIRMAN and CEO of Geomagic, envisions a future populated by small- and medium-sized companies empowered by digital design and distribution—where even a two-person company or an individual in a remote location can share their innovations with the rest of the world.

Geomagic (see <http://www.geomagic.com>) is a leading company in the industry category called digital shape sampling and processing (DSSP). DSSP describes technologies that close the loop between physical products and their digital representations. It is fundamentally changing the way products are made, driving a shift from mass manufacturing to mass customization.

In selecting her as Entrepreneur of the Year in 2005, *Inc.* magazine praised Fu as a “visionary” and said “she’s leading a modern industrial revolution that will make customization cheap and outsourcing obsolete, and forever change the way things get made—from turbines to artificial hearts.”

Vision by Necessity

Growing up in China during the Cultural Revolution, vision for Fu was not merely an asset, but a tool for survival. She was born in Nanjing, but raised by an aunt and uncle in a suburb of Shanghai. Her life changed dramatically at age seven, when she was taken away from her home and family by Red Guards. She spent more than a decade in captivity with her sister, who was three years old at the time



of the abduction. It was a life of pain and torture, sustained only by a strong will and an imagination that could conjure up beauty in the darkest of circumstances.

After Mao’s death and the end of the Cultural Revolution, Fu began her first formal education at age 18. She earned a literature degree and pursued further studies in journalism. Unfortunately, her inquisitive mind was not welcomed in her native country: Her reporting about the killing of baby girls in China was published in a leading Shanghai newspaper, leading to international outrage, imprisonment, and deportation.

Dream Becomes Reality

Fu came to the U.S. in 1981, enrolling at the University of New Mexico to study comparative literature. A year later, her life took a turn when she decided to pursue a computer science degree at the University of California-San Diego. After graduation, she worked in a series of jobs with increasing responsibilities. Although shy in demeanor, a restless intellect kept her thinking of new ways to apply technology.

While at Bell Labs, she led the development of data mining and ISDN digital switch software, the technology components that make digital telephony possible. As Director of Visualiza-

tion at the National Center for Supercomputing Applications, she initiated and managed the development of the NCSA Mosaic software project, which subsequently led to Netscape Navigator and Internet Explorer browsers.

At Geomagic, she leads a 110-person company that is applying computer geometry algorithms to a wide range of applications. Geomagic software is being used to enable mass customization, speed time to market for consumer and industrial products, make it possible for NASA to conduct in-flight inspections on the outside of the space shuttle, and optimize design for everything from racing cars to blimps.

In the following interview, journalist Bob Cramblitt talks to Fu about her challenges as a woman, an immigrant, and CEO of a company at an awkward stage of growth.

What made you decide to study computer science after coming to the U.S.?

I decided to study computer science a year after I arrived in the U.S. I was 25 at the time. I had studied comparative literature and realized I couldn't make a living with that major. I looked for a field where my literature skills could be put to good use. Computer science involves language and seemed like a lesser evil than other sciences.

I didn't have the normal background to study math or science, since I had no grade school or high school education. Fortunately in the early 1980s computer science wasn't a field taught in high school. It wasn't like today when everybody knows how to use a computer. Then, everyone in college had to start from scratch, so I wasn't at much of a disadvantage. I was naive and it was definitely a nontraditional way of picking a major. But it turned out to be a good choice.

What obstacles did you face as a Chinese immigrant pursuing computer science first academically, then in business?

The biggest obstacle in my case was the language barrier. When I came to the U.S., I only knew a few words of English. Business is more challenging for a Chinese immigrant because if you don't grow up in the U.S. you don't know American culture.

I don't believe in life/work balance, because I never think of home and work as separate things.

What did you do to acclimate?

The first thing was to try to understand the differences between Chinese and American culture. Business views are remarkably similar: They are both large countries, and people from larger countries tend to think bigger rather than provincially. Both countries also value individualism.

The huge difference is in communication. The Chinese language is symbolic and imprecise. I needed to learn how to communicate in a way people in the U.S. could understand. I took classes immigrants don't normally take in university—American history, culture, sociology. I wanted to understand the society in which I'd live and do business. Lots of immigrants don't do that when living in foreign country; they tend to hold on strongly to their native culture.

What obstacles did you face as a woman in computer science?

I was lucky to grow up in China during a time when men and women were considered equal. I didn't grow up with a view that a woman is inferior to a man. I also learned later through my Myers-Briggs profile that my thinking pattern and personal preferences are more typical of a man than a woman. I'm more rational and less emotional. I realize that women going into a male-dominated field will always have fewer colleagues, and you have to work harder to be viewed as capable, but I've never been too sensitive to those things.

What attributes from your background in China do you think helped you succeed in technology and business?

I grew up in a large country with a long history, so I had a wired-in ability to

think big and long term. Being able to live in two different societies for a significant amount of time has also been an asset. I lived in China for 24 years. I had the benefit of a cultural background where I learned more people skills than I probably would have learned if I'd grown up in the U.S.

I was exposed to a lot of atrocities growing up in China that I had to overcome. It gave me more tenacity and the ability to deal with pressure. I developed confidence from bad experiences that everything will work out—you can make it work out. That helps a lot in the business world.

What attributes as a woman have helped you in your career?

In general, women think broader and are more capable of multitasking. Even with a [Myers-Briggs] profile similar to a male, I was still brought up as a woman with different expectations of how to behave. You can't escape that.

Being a woman, I think I had more choices. Women are generally less success-driven and more opportunity-driven than men, who are raised to be successful in their careers, build a family, and be the breadwinners. Many women work because they like to, not because they have to. Being raised as a woman gave me more opportunity to explore cutting-edge technology without worrying whether I would be successful. I just followed what interested me.

In business I think my mothering instinct has been valuable. The mother is always the fallback; no job is too small and whatever someone else doesn't do is her responsibility. Mothers have a stronger sense of duty—they won't allow anything to fall apart. In general, the mother not only takes care of and loves her children, but disciplines them as well. A lot of principles I follow to raise my child I use in business.

You're the mother of a teenage daughter with a lot of interests outside of work. What do you think about the philosophy of balancing life and work?

I don't believe in life/work balance, because I never think of home and work as separate things. I've always believed in weaving life and work together. Work is a component of your life. I see it like the ying-yang concept, where work

ACM Digital Library

www.acm.org/dl

The screenshot shows the ACM Digital Library homepage. At the top, there's a navigation bar with links like "Using the ACM Digital Library" and "Frequently Asked Questions (FAQ'S)". Below this is a section titled "Recently loaded issues and proceedings" featuring "Journal of the ACM (JACM) Volume 52 Issue 5", "Journal on Educational Resources in Computing (ERIC) Volume 5 Issue 1", "ACM Transactions on Applied Perception (TAP) Volume 2 Issue 1", and "ACM Transactions on Software Engineering and Methodology (TOSEM) Volume 15 Issue 1". To the right, there are sections for "Advanced Search", "Browse the Digital Library" (listing Journals, Magazines, Transactions, Proceedings, Newsletters, Publications by Affiliated Organizations, and Special Interest Groups (SIGs)), "Personalized Services" (with "My Binder" and "TOC Service"), "Feedback" (with links to Report a problem and Take our Satisfaction survey), and "Computing Reviews" (with a link to Access critical reviews of computing literature). At the bottom, there are buttons for "Join ACM", "Join SIGs", and "Join SIGs & Sections".

The Ultimate Online INFORMATION TECHNOLOGY Resource!

- Over 40 ACM publications, plus conference proceedings
- 50+ years of archives
- Advanced searching capabilities
- Over 2 million pages of downloadable text

Plus over one million bibliographic citations are available in the ACM Guide to Computing Literature

To join ACM and/or subscribe to the Digital Library, contact ACM:

Phone: 1.800.342.6626 (U.S. and Canada)
+1.212.626.0500 (Global)
Fax: +1.212.944.1318
Hours: 8:30 a.m.-4:30 p.m., Eastern Time
Email: acmhelp@acm.org
Join URL: www.acm.org/joinacm
Mail: ACM Member Services
General Post Office
PO Box 30777
New York, NY 10087-0777 USA

and home fit together for a complete whole.

I've never understood the practice of coming home from work and shutting down. Work is not the enemy of your life. If you're at work and something happens at home you've got to take care of it, and vice versa.

Geomagic software is based on patented computational geometry technologies. What is it about computational geometry that appeals to you?

I was raised in China working with my hands in factories and in the countryside. I didn't sit behind computers or books when growing up. I always liked to create tangible things.

I think of computer geometry as a digital version of real-world objects. Mathematically, that appeals to me. It's a technology you can use to make all kinds of things. I realized from the beginning that this technology can revolutionize how we make products, especially personalized products, and that really excites me.

You have a rare skill: The ability to understand technology at the programming level, while seeing the bigger picture of its potential in the business world.

How did that come about?

It probably comes from my childhood and not being able to go to school. The ability to imagine things—pretty things, beautiful things, useful things—became overly developed.

Your mind develops rapidly from eight to 18, but if you're not able to go

I realized from the beginning that this technology can revolutionize how we make products, especially personalized products.

to school what do you do? You imagine things. At the same time, my daily life was practical. I made things. I was living in both imaginary and practical worlds. Almost anytime I could imagine something nice, I could make it happen. I'm not happy thinking just about a concept. I'm constantly asking: Where will this lead? How can I make this useful? Why would people want this? How could it improve people's lives?

I have a long attention span, and can visualize things from start to end. It could be my literary background. If you're trained in literature, you think about designing a complete piece—you think about the reader, the components to put in place, the point you want to make, and the flow you want to achieve. It's not discrete, single-problem solving. I had the ability to combine conceptual with practical problem-solving from early on, and was able to put it to good use in computer science.

At the end of 2000, four years after you founded Geomagic, you faced a financial crisis that you overcame within a year. What challenges do you now face as a growing company with more than 100 employees worldwide?

The challenges are quite different now. In 2000, the challenge was to survive and build a viable company. We survived and have thrived. Now the challenge is to build a company that will not run out of momentum. It seems simple, but the things you think about when your goal is not to run out of money are very different than when your goal is not to run out of momentum.

This stage is much harder. Despite recent events, the concept of not spending more money than you make is not difficult to understand. But to not run out of momentum you have to continue growing at a rapid pace when the risk is higher. That's a daunting job. Each stage of growth for a company is like climbing a mountain that is 10 times higher than the one you've scaled. And, you have to go down before you go up. When you go down, you're not sure if you're going to go up again.

Then, there's the people aspect. How can you educate people in good times to change? It's easier to moti-

I believe the 21st century is the century for customization, where we'll see the end of mass manufacturing.

vate people in bad times. Nothing I've done in the past 10 years is useful for the next 10. I need to continue to learn new things and face the unknown again and again. I also need to bring the whole team along. It is a difficult task to be a leader in unknown territories and to provide clear directions for others to follow.

Many entrepreneurs sell their companies at this stage, typically too early. When you reach this point, the easiest thing to do is to sell and go back to a situation where you know what to do. That's why there are so many serial entrepreneurs. If you start another company, you know how to do it better than before; you have experience and more money. It's a lot harder to take a company to the next level: one bad thing happens and you can fail because the monthly burn rate is so high. You can't put a second mortgage on your house or use your own savings to save the company anymore. It's a different game. The challenge for Geomagic is to constantly transform itself for sustainable growth.

How do you do that? Is there some kind of blueprint out there?

No, there's no blueprint. You can't apply the principles that you have learned and your friends know. You're going through a no-man's land, where the company is too large to be small and too small to be large. This is the stage where more companies die than survive, and they die for different reasons. It's not like start-ups, which typically die because they don't have a good concept or enough money. This is the stage where you already have a proven concept and are making money. If you continue to thrive, you'll go on to have a very good

company. But, no one knows why one company dies and one survives.

The technologies Geomagic is developing are having a profound effect on the way goods are designed, engineered, and made. How do you see design and manufacturing evolving over the next five to 10 years?

I believe the 21st century is the century for customization, where we'll see the end of mass manufacturing. We have the technology to do that today. The bigger changes are behavior and process changes, not technology. It will be the ecosystem enabled by technology that will drive change. Technology by itself is overrated. It enables less than 1% of what needs to happen before your vision becomes realized.

Customization in the 21st century will be less about hits and more about fit. In the last century, you could have a massive hit product with little or no customization. Now it's becoming harder to simply throw things on the wall and see what sticks. We'll be seeing more variety of customized products, and organizations becoming more decentralized. Conglomerates will downsize, and there will be more small-to medium-sized companies.

It sounds a bit like pre-Industrial Revolution.

The difference is that we will have boutique-sized companies with global distribution channels enabled by digital technologies. Before the 20th century, there were a lot of handmade, boutique products but no way to disseminate them widely. If you have a boutique business in today's world you can support it within a huge, worldwide ecosystem. I call it "digitally enabled cottage industry."

What effect will this have on outsourcing?

It will completely change the concept of outsourcing. We now outsource for cheap labor. In the future, outsourcing will be a way to infuse products with local culture; to get authentically made products. We'll not outsource for cost, but for variety, sharing of knowledge, and authenticity.

ACM, Uniting the World's Computing Professionals, Researchers, Educators, and Students



Dear Colleague,

At a time when computing is at the center of the growing demand for technology jobs worldwide, ACM is continuing its work on initiatives to help computing professionals stay competitive in the global community. ACM's increasing involvement in activities aimed at ensuring the health of the computing discipline and profession serve to help ACM reach its full potential as a global and diverse society which continues to serve new and unique opportunities for its members.

As part of ACM's overall mission to advance computing as a science and a profession, our invaluable member benefits are designed to help you achieve success by providing you with the resources you need to advance your career and stay at the forefront of the latest technologies.

I would also like to take this opportunity to mention ACM-W, the membership group within ACM. ACM-W's purpose is to elevate the issue of gender diversity within the association and the broader computing community. You can join the ACM-W email distribution list at <http://women.acm.org/joinlist>.

ACM MEMBER BENEFITS:

- A subscription to ACM's newly redesigned monthly magazine, **Communications of the ACM**
- Access to ACM's **Career & Job Center** offering a host of exclusive career-enhancing benefits
- **Free e-mentoring services** provided by MentorNet®
- **Full access to over 2,500 online courses** in multiple languages, and 1,000 virtual labs
- **Full access to 600 online books** from Safari® Books Online, featuring leading publishers, including O'Reilly (Professional Members only)
- **Full access to 500 online books** from Books24x7®
- Full access to the new **acmqueue** website featuring blogs, online discussions and debates, plus multimedia content
- The option to subscribe to the complete **ACM Digital Library**
- The **Guide to Computing Literature**, with over one million searchable bibliographic citations
- The option to connect with the **best thinkers in computing** by joining **34 Special Interest Groups or hundreds of local chapters**
- **ACM's 40+ journals and magazines** at special member-only rates
- **TechNews**, ACM's tri-weekly email digest delivering stories on the latest IT news
- **CareerNews**, ACM's bi-monthly email digest providing career-related topics
- **MemberNet**, ACM's e-newsletter, covering ACM people and activities
- **Email forwarding service & filtering service**, providing members with a free acm.org email address and **Postini** spam filtering
- And much, much more

ACM's worldwide network of over 92,000 members range from students to seasoned professionals and includes many of the leaders in the field. ACM members get access to this network and the advantages that come from their expertise to keep you at the forefront of the technology world.

Please take a moment to consider the value of an ACM membership for your career and your future in the dynamic computing profession.

Sincerely,

Wendy Hall

A handwritten signature in blue ink that reads "Wendy Hall".

President

Association for Computing Machinery



Association for
Computing Machinery



Association for
Computing Machinery

Advancing Computing as a Science & Profession

membership application & *digital library order form*

Priority Code: ACACM10

You can join ACM in several easy ways:

Online
<http://www.acm.org/join>

Phone
+1-800-342-6626 (US & Canada)
+1-212-626-0500 (Global)

Fax
+1-212-944-1318

Or, complete this application and return with payment via postal mail

Special rates for residents of developing countries:
<http://www.acm.org/membership/L2-3/>

Special rates for members of sister societies:
<http://www.acm.org/membership/dues.html>

Please print clearly

Name _____

Address _____

City _____

State/Province _____

Postal code/Zip _____

Country _____

E-mail address _____

Area code & Daytime phone _____

Fax _____

Member number, if applicable _____

Purposes of ACM

ACM is dedicated to:

- 1) advancing the art, science, engineering, and application of information technology
- 2) fostering the open interchange of information to serve both professionals and the public
- 3) promoting the highest professional and ethics standards

I agree with the Purposes of ACM:

Signature _____

ACM Code of Ethics:

<http://www.acm.org/serving/ethics.html>

choose one membership option:

PROFESSIONAL MEMBERSHIP:

- ACM Professional Membership: \$99 USD
- ACM Professional Membership plus the ACM Digital Library:
\$198 USD (\$99 dues + \$99 DL)
- ACM Digital Library: \$99 USD (must be an ACM member)

STUDENT MEMBERSHIP:

- ACM Student Membership: \$19 USD
- ACM Student Membership plus the ACM Digital Library: \$42 USD
- ACM Student Membership PLUS Print CACM Magazine: \$42 USD
- ACM Student Membership w/Digital Library PLUS Print
CACM Magazine: \$62 USD

All new ACM members will receive an
ACM membership card.

For more information, please visit us at www.acm.org

Professional membership dues include \$40 toward a subscription to *Communications of the ACM*. Member dues, subscriptions, and optional contributions are tax-deductible under certain circumstances. Please consult with your tax advisor.

RETURN COMPLETED APPLICATION TO:

Association for Computing Machinery, Inc.
General Post Office
P.O. Box 30777
New York, NY 10087-0777

Questions? E-mail us at acmhelp@acm.org
Or call +1-800-342-6626 to speak to a live representative

Satisfaction Guaranteed!

payment:

Payment must accompany application. If paying by check or money order, make payable to ACM, Inc. in US dollars or foreign currency at current exchange rate.

Visa/MasterCard American Express Check/money order

Professional Member Dues (\$99 or \$198) \$ _____

ACM Digital Library (\$99) \$ _____

Student Member Dues (\$19, \$42, or \$62) \$ _____

Total Amount Due \$ _____

Card # _____

Expiration date _____

Signature _____

DOI:10.1145/1592761.1592776

 Article development led by ACM QUEUE
queue.acm.org

As the sophistication of wiretapping technology grows, so too do the risks it poses to our privacy and security.

BY WHITFIELD DIFFIE AND SUSAN LANDAU

Communications Surveillance: Privacy and Security at Risk

WE ALL KNOW the scene: It is the basement of an apartment building and the lights are dim. The man is wearing a trench coat and a fedora pulled down low to hide his face. Between the hat and the coat we see headphones, and he appears to be listening intently to the output of a set of alligator clips attached to a phone line. He is a detective eavesdropping on a suspect's phone calls. This is wiretapping—as it was in the film noir era of 1930s Hollywood. It doesn't have much to do with modern electronic eavesdropping, which is about bits, packets, switches, and routers.

Scarcely a generation ago, phone calls traveled through wires between fixed locations, encoded as fluctuating electric signals. Now phones are mobile, and, through most of their journeys, phone calls are

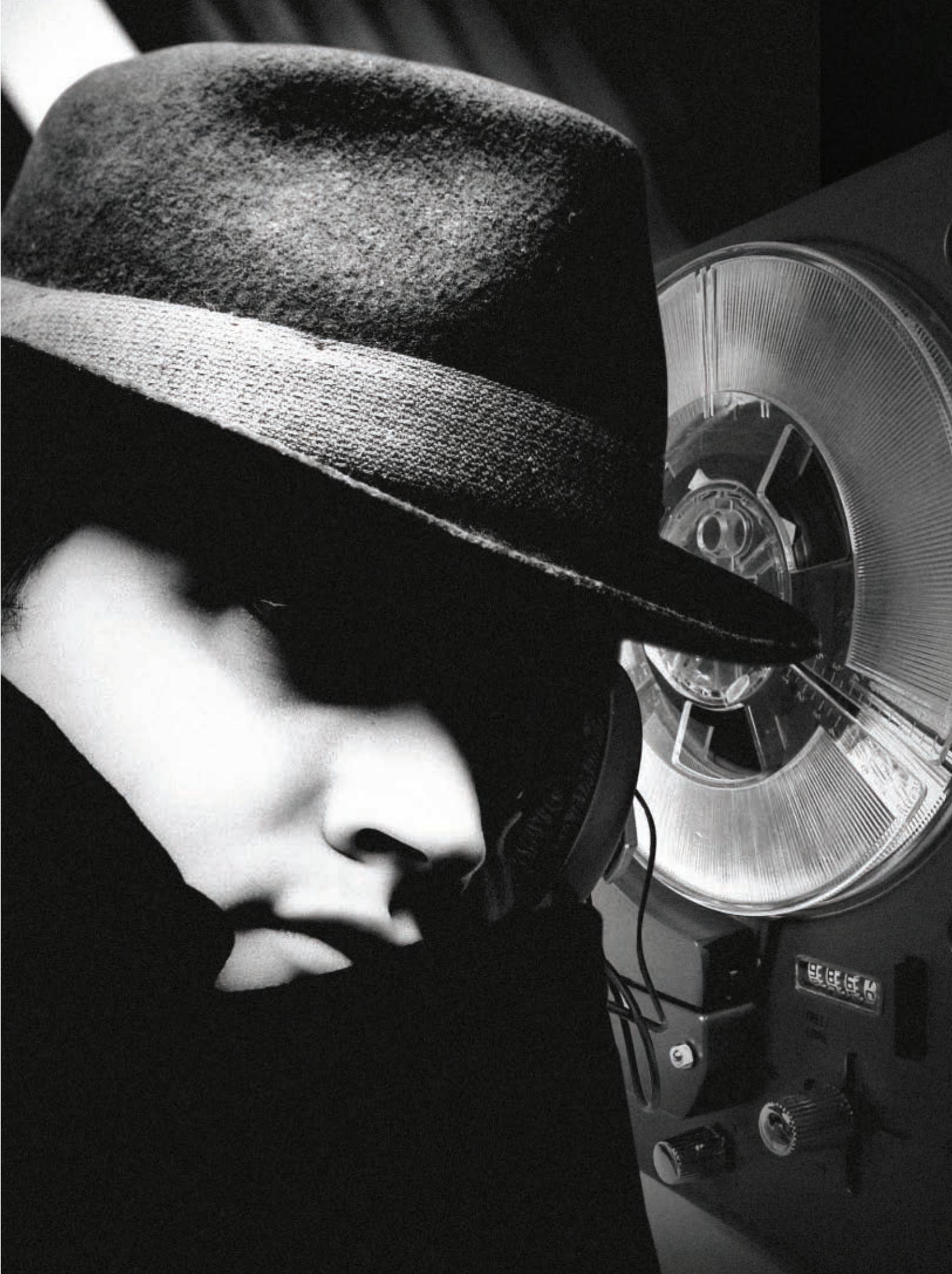
encoded in bits. Voices are digitized shortly after they leave the speaker's lips, carried over an IP network as packets, and returned to analog format for presentation to the listener's ears.

Although big changes in telephony have given rise to equally big changes in wiretapping, the essentials remain the same. The interception and exploitation of communications has three basic components: accessing the signal, collecting the signal, and exfiltrating the signal. Access may come through alligator clips, a radio, or a computer program. *Exfiltration* is moving the results to where they can be used. Collection may be merged with exfiltration or may involve recording or listening.

A phone call can be intercepted at various points along its path. The tap can be in the phone itself, through introduction of a bug or malware that covertly exfiltrates the call, often by radio. The tap can be at the junction box, in a phone closet down the hall, on a telephone pole, or on the *frame* where incoming subscriber lines connect to the telephone company *central office*.

The development in the 1980s of digital switches and the features they made possible created problems for traditional local-loop wiretapping. Call forwarding in particular, which diverts the call to a different number before it ever reaches the frame, was problematic. To avoid the possibility of being bypassed, the tap must be placed at or above the level of the diversion. Fortunately for wiretappers, digital switches also introduced conferencing, which allowed several people to converse at once. Taps could be implemented by conferencing in a silent additional party. Taps on analog circuits can, in principle, be detected by the power they drain. Digital wiretaps are invisible to the target but require changes in the programming of the switch rather than extra connections to the frame.

In the 1990s the FBI, claiming that advanced switching technology threatened the effectiveness of wiretapping, persuaded Congress to require that telephone companies build wiretap-



ping capability into their networks. This resulted in the Communications Assistance for Law Enforcement Act (CALEA) of 1994.

It is a long way from putting clips on wires to having government standards for electronic eavesdropping. These changes, made in the name of security, have created risks. How did this come about? Does wiretapping actually make us more secure?

We start with an overview of the convoluted history of wiretapping, focusing on the U.S., and then turn to issues of privacy and security.

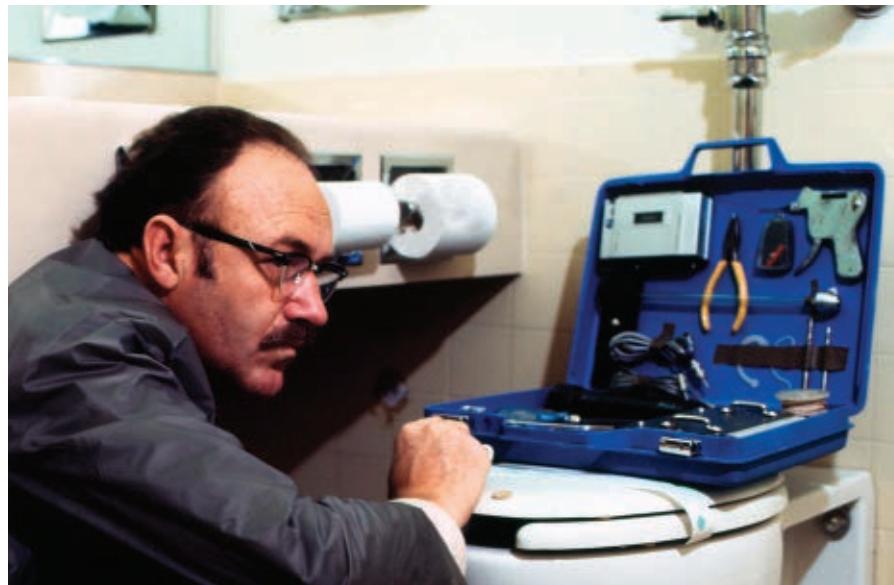
The Legal Side

The telegraph was invented in the 1830s, the telephone in the 1870s. Police wiretapping appeared in the 1890s but saw limited use until Prohibition when the production, sale, and transport of alcoholic beverages were made illegal in 1919. Law or no law, alcohol remained popular, and illegal enterprises grew to serve the demand.

Wiretapping was the perfect tool for investigating crimes such as these that lack victims who complain and give evidence to the police; it performed a search that was invisible and could provide law enforcement with detailed information about the criminal activity. Wiretapping produced search-like results without requiring intrusion into the suspect's property. Was it to be regulated as a violation of the right to be free from unreasonable searches and seizures guaranteed by the Fourth Amendment? Decades were to pass before this question was answered.

In the 1928 *Olmstead* case, the Supreme Court ruled that wiretapping was not a search and therefore did not violate the Fourth Amendment. Following the Federal Communications Act of 1934, which made "interception and divulgence" of wired communications illegal, the Supreme Court changed direction. In a 1937 ruling based on the new law, the court refused to allow wiretap evidence against a bootlegging suspect.

The Communications Act might have put an early end to wiretapping's law-enforcement career, but the Justice Department interpreted the court decision narrowly, permitting interception as long as the results were not divulged outside the federal government. The FBI took advantage of this interpreta-



Iconic images of wiretapping and surveillance were made popular by movies and television; from the memorable film noir approach (page 43) to Francis Ford Coppola's 1974 thriller *The Conversation*, released before and during the Watergate era, and often cited as the classic film on electronic surveillance and potential threat posed by "new" technologies.



"The Wire," a recent U.S. cable television show, hinged around tracking police investigations of criminal activities using modern surveillance technologies, including cellphones and SMS messaging.

tion to continue wiretapping without court orders—sometimes with Department of Justice oversight, sometimes not—for another 30 years.

The Communications Act said nothing about bugs, which listen to sounds in the air rather than signals on wires. This led to an odd split: bugs yes, wiretaps no. Over the decades, the Supreme Court saw less and less distinction. In 1967, in the *Katz* decision, it finally recognized that "the Fourth Amendment protects people, not places." Henceforth, warrants would be as much a requirement of electronic searches as

physical ones.

The following year, Congress addressed wiretapping and bugging in Title III of the Omnibus Crime Control and Safe Streets Act, which set out the circumstances under which wiretap orders could be obtained for criminal investigations. Congress saw wiretaps as a particularly insidious search and made the warrant requirements more stringent than those for normal searches (though these have been relaxed somewhat over the years).

In 1978 Congress passed the Foreign Intelligence Surveillance Act

(FISA), which established a basis for wiretapping quite different from that of Title III wiretap orders intended to collect evidence of a crime and require probable cause to believe that the suspect is involved in serious illegal activity. FISA wiretaps are intended to collect intelligence and require that the suspect be an agent of a foreign power or terrorist group.

Title III wiretaps are summarized in an annual Wiretap Report produced each year by the Administrative Office of the U.S. Courts. It lists the prosecutor, judge, crime, number of intercepted conversations, and number of incriminating conversations. By contrast, except in rare cases in which the evidence they yield is presented in court, only the annual number of FISA wiretaps is made public. There are currently about 2,000 Title III and another 2,000 FISA wiretap warrants each year.

Eavesdropping practices vary from country to country, and since many nations release no information about electronic surveillance, a comprehensive view is difficult to attain. Britain did not pass a wiretap law until 1985 when a European Court ruling faulted the country's lack of a clear warrant procedure. A similar European Court ruling in 1990 led to a French wiretapping law in 1991. Both Britain and France report having significantly more wiretaps than the U.S.

Not all wiretaps are a result of official or acknowledged government action. When SMS messaging went awry in Athens in 2005, an investigation found that for 10 months someone had been wiretapping senior members of the Greek government. The eavesdropping appears to have been stopped after it was discovered. Although no information has surfaced about who did the wiretapping, a good bit is known about how it was done. The 1994 CALEA law requiring telephone systems to be wiretap ready applies only to switches installed in the U.S., but since manufacturers try to have as few versions of their products as possible, it has had worldwide impact. When the Greek Vodafone network purchased a switch from Ericsson, it didn't order wiretapping capabilities; wiretapping software was present in the switch but was supposed to be shut off. In particular, auditing software that would have



The future of privacy will depend on a combination of legal and technical measures by which device-to-device communications are protected.



been operating if the wiretapping feature had been ordered was not present. When unknown parties turned some of the wiretapping features on, their actions went unrecorded.

Wiretapping without a Legal Foundation

The Greek case wasn't the only warrantless wiretapping uncovered during that period. In 2005 the *New York Times* revealed that the U.S. government had been wiretapping communications to and from the U.S. without a warrant. After the passage of FISA, NSA (National Security Agency), America's foreign-intelligence eavesdroppers, had been forbidden to listen in on radio communications inside the U.S. without a warrant unless at least one end of the communication was outside the country and the internal end was not a targeted "U.S. person." Interception of purely domestic communications within the country always required a warrant. As more messages came to travel by fiber-optic cable and fewer by radio, NSA was forced to turn to other, not necessarily legal, approaches.

The vast American investment in communications infrastructure makes it economical for parties in other parts of the world to route their calls through the U.S., and this *transit traffic* seems a reasonable foreign intelligence target. When transiting communications were in the form of a radio signal that could be intercepted from U.S. soil, it was not difficult for NSA to determine what was transit traffic. When traffic moved to optical fibers and IP-based communications, separating out the transit traffic that could be eavesdropped upon without a warrant became more difficult. That was one concern. There was another.

In the post-9/11 anti-terrorist climate, some government elements wanted a substantive change, permitting warrantless interception of communications in which one end was "reasonably believed to be located outside the United States," regardless of the status of the U.S. end. Interception was placed not at the cable heads where calls entered the country, but at switches carrying both internal and transit traffic. This meant that purely domestic calls were likely to be intercepted as well.

A technician at an AT&T switching office in San Francisco leaked documents showing that a fiber-optic signal at the office was being split: a copy of the signal went into a “secret room,” where it was analyzed and part of its contents sent elsewhere for further analysis. The leaked documents—whose authenticity was confirmed by AT&T during a subsequent court case—reveal that the San Francisco office was only one of a number of offices set up this way.

From the wiretapper’s viewpoint, the end of the rainbow would be the ability to store all traffic and then decide later which messages were worthy of further study. Although this is usually not feasible, storing the transactional information about telephone calls—calling and called numbers, time, duration—is. These CDRs (call detail records) are routinely retained by the carriers who use them for planning and billing purposes. Law enforcement had previously been able to obtain call details—in police jargon *pen register* and *trap-and-trace*—collected in response to court orders targeted at individual phones. By comparison, the CDR database provides information on all the subscribers over long periods of time, a rich source of information about customer activities, revealing both the structure of organizations and the behavior of individuals. Several telephone companies appear to have surrendered them in response to government pressure without demanding court orders.

Wiretapping in an IP-based World

Internet communications cannot be effectively exploited using the facilities of traditional telephony, so as early as 2000 the FBI developed a tool for wiretapping at ISPs. The tool—initially named Carnivore but eventually given the less menacing title DCS-3000—examined packets passing through the ISP and copied those that met intercept criteria stored in internal tables. The tables were set through a remote connection to the FBI’s own offices. Surprisingly for law enforcement, which places great store on the chain of custody of evidence, Carnivore had little provision for auditing and overall poor internal security. Rather than having a separate name and password for each user, it relied on a single shared

From the wiretapper’s viewpoint, the end of the rainbow would be the ability to store all traffic and then decide later which messages were worthy of further study.

login. More significant from a privacy standpoint, Carnivore bypassed the traditional process of wiretapping in which the court issues an order but the carrier’s personnel execute the order. This gives the carrier both the obligation and opportunity to challenge the order in court if it believes the order to be illegal. When the order is implemented by a message sent directly from the FBI to the Carnivore box, this additional layer of oversight is lost.

In parallel with its technical activities, the FBI worked to extend wiretapping law to the Internet. CALEA had been passed with an exemption for “information services” (that is, the Internet), and with the rise of VoIP (voice over IP), the FBI feared it would lose an important investigative tool. VoIP comes in many flavors, from the peer-to-peer model employed by Skype to others in which the path between the subscriber and the telephone central office is traditional telephony but IP communications are used throughout most of the call’s path.

The FBI began slicing the salami with the “easy” cases in which VoIP communications behave most like traditional phone calls, and it was successful in getting the courts to agree to this extension. Most IP communications, however, do not behave as telephone calls; peer-to-peer VoIP systems, for example, use a centralized mechanism to provide the communicating parties with each other’s IP addresses but rely on the Internet for actual communication. In this scheme there is no central point at which a wiretap could be authorized. If regulation were to require that IP-based communications adopt a centralized architecture like the telephone network, the innovation that is the engine of high-tech industry could be stifled.

In 2007, Congress legalized warrantless wiretapping; in 2008, it went a long step further, not only legalizing new wiretapping practices but also giving retroactive immunity to telephone companies that had colluded with the government in performing warrantless electronic eavesdropping. The FISA court previously had reviewed individual warrants; now certain classes of wiretaps would not be reviewed individually but conducted under procedures reviewed periodically by the court.

Whither Privacy

At the time the U.S. wiretap laws were passed, real-time access to transactional information of who was talking to whom and when was not easy to acquire. Modern switching technology introduced in the 1980s changed that, and police hungrily pursued the investigative possibilities. Because transactional data—phone number, time of call—are analogous to the information on the outside of a letter, access requires only a subpoena, which is much easier to obtain than a wiretap warrant. Whom you talk to and when may be less intimate than the transcripts of your conversations but can reveal a great deal about you. When your spouse calls you from the office in the late afternoon, do you frequently respond by calling a certain number? Perhaps when you learn your spouse is working late, you let someone else know you are free.

In a cellphone world people are constantly at their telephones. Not only do they make more calls, but they also reveal more information: times and numbers are joined by location in the transactional record. In an Internet world, each connection to a Web site is a transaction. Even though a query string is not transactional data, the sites visited after the search engine frequently make the character of the query clear.

Curiously, the greatest threat to privacy may not be snooping on people but snooping on things. We are moving from a world with a billion people connected to the Internet to one in which 10 or 100 times that many devices will be connected as well. These range from the much-discussed smart refrigerator that knows when it is time to order more milk to RFID (radio-frequency identification) tags in products that enable the tracking of where the goods are located before, and perhaps after, retail sale. Particularly in aggregation, the information reported by these devices will blanket the world with a network whose gaze is difficult to evade. The future of privacy will depend on a combination of legal and technical measures by which device-to-device communications are protected.

Whither Security

It is not just privacy that is at risk under the new regime, it is security as well. National security is much broader than

simply enabling intelligence and law-enforcement investigations. Although undertaken in the name of national security, building wiretapping into our telecommunications system may be a greater threat to that security than the spies and terrorists against whom it is aimed.

First and foremost, information security means protecting public and private computing and communications systems against attacks from both inside and outside. It was the need for that type of protection that caused the European Union in 1999 and the U.S. government in 2000 to relax their export controls on strong cryptography, a change that bolstered the security of Internet communications.

A network may be designed to provide security to its individual users against everything except authorized intrusions by the network itself, a plausible goal for a Department of Defense (DoD) network. Such a model requires centralization of authority that is possible for DoD, and might have been possible for the Internet in 1985—when it was a U.S. project—but is not feasible now.

The Internet has become essential to modern life. Business and personal communications—and even critical infrastructure—rely upon the network to function. Yet the combination of attacks on the network and on network hosts means that we are increasingly reliant upon an unreliable network.

A number of efforts are under way to improve this, from the use of Secure Sockets Layer (SSL) to protect Internet commerce, to the deployment of Internet Protocol security (IPsec) to protect any IP communication, to the implementation of Domain Name System Security Extensions (DNSSEC) to protect the domain-name system. Research is occurring in both Europe and the U.S. on secure Internet protocols and such plans as expounded in the recently released *White House Cyberspace Policy Review*.

The unauthorized use of wiretapping facilities in the Greek Vodafone system shows one level at which surveillance facilities can be misappropriated. NSA's activities under the Bush administration show another. FBI expansion of its wiretapping authority beyond what was originally envisioned

in CALEA shows a third.

Building wiretapping capabilities into communications infrastructures creates serious new risks. The complexity that wiretapping introduces led the Internet Engineering Task Force (IETF) to conclude that it should not “consider requirements for wiretapping as part of the process for creating and maintaining IETF standards” (RFC 2804).

The surveillance we are attempting to build may increase security in some ways, but it also creates serious risks in a network infrastructure that supports all of society. Given the importance of the Internet to society—and given the importance the network has in communications between people and their friends, governments and their citizens, businesses and their customers, and in all of society—communications security is critical, and that should take precedence in the debate over communications security versus communications surveillance. □

Related articles on queue.acm.org

Document and Media Exploitation

Simson Garfinkel

<http://queue.acm.org/detail.cfm?id=1331294>

VoIP Security: Not an Afterthought

Douglas C. Sicker and Tom Lookabaugh

<http://queue.acm.org/detail.cfm?id=1028898>

A Conversation with Donald Peterson

<http://queue.acm.org/detail.cfm?id=1028901>

Whitfield Diffie is a visiting professor in the Information Security Group at Royal Holloway, University of London. For nearly two decades Diffie worked at Sun Microsystems Laboratories, where as Chief Security Officer he was the chief exponent of Sun's security vision and responsible for developing Sun's strategy to achieve that vision. He is best known for his discovery of the concept of public key cryptography and has spent many years of his career working on the public policy aspects of cryptography. He and Susan Landau are joint authors of the book *Privacy on the Line* (MIT Press), which examines the politics of wiretapping and encryption.

Susan Landau is a Distinguished Engineer at Sun Microsystems Laboratories, where she works on security, cryptography, and policy, including surveillance and digital-rights management issues. She serves on the NSF CITSE Advisory Committee, the Commission on Cyber Security for the 44th Presidency, the editorial board of *IEEE Security and Privacy*, and as a Viewpoint section board member for *Communications of the ACM*; she previously served for six years as a member of the National Institute of Standards and Technology's Information Security and Privacy Advisory Board. Landau is the recipient of the 2008 Women of Vision Social Impact Award, a AAAS Fellow, and an ACM Distinguished Engineer.

Participatory sensing technologies could improve our lives and our communities, but at what cost to our privacy?

BY KATIE SHILTON

Four Billion Little Brothers?

Privacy, mobile phones, and ubiquitous data collection

THEY PLACE PHONE calls, surf the Internet, and there are close to four billion of them in the world. Their built-in microphones, cameras, and location awareness can collect images, sound, and GPS data. Beyond chatting and texting, these features could make phones ubiquitous, familiar tools for quantifying personal patterns and habits. They could also be platforms for thousands to document a neighborhood, gather evidence to make a case, or study mobility and health. This data could help you understand your daily carbon footprint, exposure to



air pollution, exercise habits, and frequency of interactions with family and friends.

At the same time, however, this data reveals a lot about your regular locations, habits, and routines. Once such data is captured, acquaintances, friends, or authorities might coerce you to disclose it. Perhaps worse, it could be collected or reused without your knowledge or permission. At the extreme, mobile phones could become the most widespread embedded surveillance tools in history. Imagine carrying a location-aware bug, complete



with a camera, accelerometer, and Bluetooth stumbling everywhere you go. Your phone could document your comings and goings, infer your activities throughout the day, and record whom you pass on the street or who engaged you in conversation. Deployed by governments or compelled by employers, four billion "little brothers" could be watching you.

Whether phones engaged in sensing data are tools for self and community research, coercion, or surveillance depends on who collects the data, how it is handled, and what privacy protections

users are given. As these new forms of data begin to flow over phone networks, application developers will be the first line of defense for protecting the sensitive data collected by always-present, always-on mobile phones.

I should mention that I'm not one of the developers on the front lines. I work in science and technology studies (STS)—a social science interested in the ways people, technologies, and data interact and affect each other.^a The

developers I work with might say STS is about telling them what they should be doing, which I must admit, is the goal of this article. I worry about the consequences of mobile phones as sensors, and have a number of opinions about what programmers, as well as social scientists, might do to make this sort of data collection work without slipping into coercion, surveillance, and control.

Participatory Sensing

Research that uses mobile phones to collect data for personal or social proj-

a See http://en.wikipedia.org/wiki/Science_and_technology_studies

ects is called mobile, urban, or participatory sensing.^{2–4} Participatory sensing is meant to enable (and encourage) anyone to gather and investigate previously invisible data. It tries to avoid surveillance or coercive sensing by emphasizing individuals' participation in the sensing process. Applications designed to enable participatory sensing range from the very personal and self-reflective to shareable data meant to improve an individual's health or a community's experience. This article examines three applications from UCLA's Center for Embedded Networked Sensing (CENS) to illustrate the diversity of possibilities, as well as suggest data collection and sharing concerns.

Personal Environmental Impact Report (PEIR). Participants in PEIR (<http://peir.cens.ucla.edu/>) carry mobile phones throughout their day to calculate their carbon footprints and exposure to air pollution—both big concerns in smoggy Los Angeles, where the project is based. By referencing GPS and cell towers, the phones upload participants' locations every few seconds. Based on these time-location traces, the PEIR system can infer participant activities (for example, walking, biking, driving, or riding the bus) throughout the day. The system maps the combination of location, time, and activity to Southern California regional air quality and weather data to estimate individual carbon footprint and exposure to particulate matter. Sensing a participant's location throughout the day enables more accurate and previously unavailable information about environmental harms people face, as well as the harms they create. To participate, individuals need to record and submit a continuous location trace.

Biketastic. This project (<http://biketastic.com>) was designed to improve bike commuting in Los Angeles, a city notoriously unfriendly to cyclists. Bikers carry a GPS-enabled mobile phone during their commute that automatically uploads their bike routes to a public Web site. The phone also uses its accelerometer to document the roughness of the road, and takes audio samples to analyze volume of noise along the route. Participants can log in to see their routes combined with ex-

isting data, including air quality, time-sensitive traffic conditions, and traffic accidents. They can also use the system to share information about their routes with other riders. By combining existing local conditions with biker-contributed data, Biketastic will enable area bikers to plan routes with the least probability of traffic accidents; with the best air quality; or according to personal preferences, such as road-surface quality or connections with public transportation. While Biketastic shares location data through a public map, individuals use pseudonymous user names.

AndWellness. Currently under development, AndWellness is a personal monitoring tool designed to encourage behavioral change. It helps clients work independently or with a coach to document places and times when they stray from a healthy eating or exercise plan. During an initial week of documentation, AndWellness prompts users to input personal assessments throughout the day. These assessments ask users when they last ate and whether they were on plan. After a week of tracking and data analysis, users can see places and times where they tend to stray from their plan, and plan interventions to combat unwanted variations. AndWellness collects not only location, but also sensitive data about diet and habits. Individuals might choose to share this data with a support group, coach, therapist, doctor, family, or friends.

Taking participatory sensing from a possibility enabled by the mobile-phone network to a coordinated reality is rife with challenges. Among these challenges are the ethics of repurposing phones, now used as communication tools, for data collection and sharing. How can individuals determine when, where, and how they wish to participate? How much say do they get over what they wish to document and share?

Privacy in Participatory Sensing

Privacy—the ability to understand, choose, and control what personal information you share, with whom, and for how long—is a huge challenge for participatory sensing. Privacy decisions have many components, including identity (who is asking for the data?), granularity (how much does the

data reveal about me?), and time (how long will the data be retained?).^{7,10,11} Location traces can document and quantify habits, routines, and personal associations. Your location might reveal your child's school, your regular trips to a therapist or doctor, and times when you arrived late or left early from work. These traces are easy to mine and difficult or impossible to retract once shared.

Sharing such granular and revealing digital data could have a number of risks or negative consequences. Some safety and security threats, such as thieves or stalkers, are obvious. Perhaps less apparent—and probably more likely—are other social consequences. Think about how frequently you beg off a social engagement with a little white lie, or keep your location and activities secret to surprise a friend. Much like Facebook's ill-fated Beacon service, participatory sensing could disrupt the social boundaries we have come to expect. What if someone with authority over you (your employer, the government) collects or accesses your location data? It's easy to imagine a chilling effect on legal, but stigmatized, activities. Would you be as likely to attend a political protest, or visit a plastic surgeon, if you knew your location was visible to others? Large databases of location data accessible by subpoena also could become evidence for everything from minor civil disputes to messy divorce cases.

Maybe most importantly, privacy is a vital part of your identity and self-presentation. Deciding what to reveal to whom is part of deciding who you are. I might want to track when and where I tend to overeat, but I see no reason to share that information with anyone but my doctor. Similarly, I might take part in a political data collection project on the weekend, but that doesn't mean my parents need to know. Respecting the many gradations between public and private, and giving people the ability to negotiate those gradations, are integral to respecting individual privacy.

In the U.S. and Europe, fair information practices are one standard for protecting the privacy of personal data. Originally codified in the 1970s, the Code of Fair Information Practice outlines data-management principles to help organizations protect personal

data.^{12,13} These codes are still considered a gold standard for privacy protection.¹⁴ But the principles, designed for corporations or governments rather than many distributed data collectors, are no longer enough. Data gathered during participatory sensing is more granular than traditional personal data (name, Social Security number, among others). It reveals much more information about an individual's habits and routines. Furthermore, data is no longer gathered solely by large organizations or governments with established data practices. Individuals or community groups might create participatory sensing applications and begin collecting personal data.¹⁵

Enabling Participation in Privacy

This is where the responsibility of developers comes into the equation. How can developers help individuals or small groups launching participatory sensing projects implement appropriate data-protection standards? To create workable standards with data so granular and personal, systems must actively engage individuals in their own privacy decision making. At CENS, we call this participatory privacy regulation—the idea that systems can help users to negotiate disclosure decisions depending on context (who is asking, what is being asked for, and so on). We need to build systems that improve users' ability to make sense of, and thereby regulate, their privacy.

Building such systems is a major, unmet challenge.⁶ As the first steps toward meeting this challenge, we propose three new principles for developers to consider and apply when building mobile data-gathering applications. These principles are purposefully broad, because "acceptable" data practices might vary across applications (a medical project might be justified in collecting much more personal data, with stringent protections, than a community documentation project). These principles are thinking tools to help developers adapt privacy protections to participatory sensing applications.

Participant primacy. The goal of participatory privacy regulation is to give participants as much control over their location data as possible. GPS traces or the secondary traces created by geotagged media should belong to in-

Privacy is a vital part of your identity and self-presentation. Deciding what to reveal to whom is part of deciding who you are.

dividuals. Participants should be able to make and revoke decisions to share subsets of the data with third-party applications. Framed this way, participants are not just subjects of data collection, but take the role of investigators (when they collect data to participate in self-analytic applications) or co-investigators (when they contribute their data to larger research initiatives). As such, they should have input into how data is collected, processed, stored, and discarded.

Developers can enable participants to own and manage their data by tailoring access-control and data-management tools for use by individual participants. Users collecting revealing sensing data are going to need secure storage and intuitive interfaces to manage access and sharing. As an example, CENS researchers are developing a personal data vault (PDV) to give individuals private and robust storage for their sensing data. The PDV provides services such as authentication and access control, allowing participants not only to collect all of their sensing data in one place, but also to specify which individuals and groups in their social network can see which datasets. Similar tools are in development in research labs at Stanford⁸ and AT&T,¹ and are not unlike commercial applications such as Google Health⁵ and Microsoft's HealthVault.⁹

As developers build data-management tools to put personal data control back in the hands of individuals, they will need to think about which controls users will need to make privacy and sharing decisions. At a very basic level, sharing decisions should take into account identity (who's asking?), time (send data only between 9 A.M. and 5 P.M.), location (send data only when I'm on campus), and data type (share only geotagged photos). More advanced techniques for developers to consider include access controls based upon activity (share only driving routes) or routine (don't share anomalous routes).

Application developers can further protect participant primacy by limiting the amount of raw data a participant is required to share outside of the vault. When privacy is at stake, more data is not always better. For example, participants in Biketastic may collect

their location data 24/7 to the PDV, but share data *with* Biketastic only during days and times when they regularly commute by bike. Biketastic doesn't need to know where the participants are during working hours, when they take their lunch breaks, or how they typically spend their evenings. A different example of collecting minimal data is requesting processed, rather than raw, data. Developers could build applications such as PEIR to request only inferred activity data (time spent driving, walking, and indoors) and ZIP code, rather than granular location data. PEIR doesn't need to know what street a participant was on—only what carbon-generating activity they were engaged in. By collecting the minimum amount of information needed for a service, application developers can help participants maintain control over their raw data.

Data legibility. Participatory sensing systems can help participants make sense of, and decisions about, their data by visualizing granular, copious data in ways individuals can understand. Methods to improve data legibility include visualization using tools such as maps, charts, icons, pictures, or scales. Data legibility also includes showing users who has accessed their data and how frequently, and showing participants where their data goes and how long it remains accessible. System features should increase participants' understanding of complex risks and help them make better decisions about data capture, sharing, and retention.

Developers should get creative about what legibility might mean. An application's user interface, for example, could help users not only set data-sharing policies, but also see the results of their policies. Imagine a Facebook pop-up that asks, "Do you really want to share the album 'Party Pics' with your father?" Developing features either for data vaults or for sensing applications that illuminate who can see what data will help users better understand the consequences of data sharing.

Another approach is to show multiple interpretations of collected data. The AndWellness interface, for example, uses both maps and timelines to help users draw conclusions about when and where their eating habits strayed from their plans. Developers

Participatory sensing opens the door to entirely new forms of granular and pervasive data collection. The risks of this sort of data collection are not always self-evident.

might also experiment with natural language, helping translate numerical data or complex algorithms into something easier to understand. Natural language might make inferences from data points (for example, this bike route has a few hills in the middle, most of them easy, and one difficult hill at the end); or plain text descriptions can explain how calculation and processing works (for example, clicking on a route in PEIR takes the participant to a "Trip Journal" with a step-by-step breakdown of how the system calculated the impact and exposure for that route).

Longitudinal engagement. Finally, developers will need to consider time as a factor that affects privacy in participatory sensing. You may end participation in a carbon footprint calculator when you start taking public transportation to work, but enroll in a new health program after receiving a surprising diagnosis. Personal habits and routines change over time, altering the data collected into personal data vaults.

Because time is such a critical factor, application interfaces should encourage participants to engage with the data from the point of collection through analysis, long-term retention, or deletion. Systems should enable continued engagement to allow participants to change their data practices as their context changes. The crux of engaging individuals with decisions about their data is refusing to put that data in a black box. Instead, analyzing, learning from the data, and making ongoing choices about the data become the goals of sensing.

We offer several suggestions for how developers can encourage long-term engagement. Policies that require users to check back in with a vault or application on a regular basis can remind them to update their sharing preferences as their needs change. A data vault could remind users to update their sharing preferences every time they add new contacts or applications. Building adaptive filters can also enable participants to change their data sharing as their preferences change. Such filters could learn from user behavior to respond to privacy preferences. For example, the vault could learn never to share a certain route or could learn to check with users before sharing any routes recorded after 9 P.M.

A TraceAudit is another idea that helps users connect with their data over time. The TraceAudit builds on the idea of an Internet traceroute and relies on careful logging procedures. An interface that allows users access to logs can let them trace how their data is used by an application, where the data has been shared, and who has had access to it. For example, a TraceAudit of data use in PEIR can show participants exactly how their raw location traces become calculations of impact and exposure, and how data is shared during that process. A log could show users that their PDV sent PEIR raw data on weekdays between the hours of 7 A.M. and 8 P.M. PEIR performs activity classification based on this raw data and sends a summary of the activities and the ZIP codes in which they occurred to the California Air Resources Board. PEIR receives back PM2.5 (fine particle) pollution exposure and CO₂ emission values to correspond with these activities and ZIP codes. PEIR then saves and displays these total calculations for users. The TraceAudit provides transparency and accountability, helping individuals to see how PEIR has used and shared their data.

Challenges Beyond Technology

System design that pays attention to participant primacy, longitudinal engagement, and data legibility will help users make data-sharing decisions and protect their privacy in participatory sensing. Technical decisions, however, won't be enough to ensure privacy for sensing participants. Participant engagement in privacy decision making must also be fortified by supporting social structures.

Participatory sensing opens the door to entirely new forms of granular and pervasive data collection. The risks of this sort of data collection are not always self-evident. Even if we give people options for managing their data, they may not understand the benefits of doing so. Data literacy must be acquired over time through many avenues. Public discussion and debate about participatory sensing will be critical to educating participants about the risks and possibilities of sensing data. Discussion forums and blogs play an important role, as do traditional media and even community groups.

Further, partakers in participatory sensing are going to need to understand what happens with their data once it leaves their personal vault and is used by third-party applications. Diverse and plentiful applications for participatory sensing data can help to achieve the potential usefulness of participatory sensing, but will also make it difficult for participants to understand which applications are trustworthy and abide by acceptable data practices. Participants need to know what they are signing up for—and cryptic, fine-print EULAs (end-user license agreements) aren't the answer.

Users should know how long an application will retain their data and whether it will pass the data on to other parties. A voluntary labeling system, much like "Fair Trade" labels on food, could help consumers distinguish applications that abide by a minimum set of responsible data practices. These might include logging data use and keeping audit trails, and discarding location data after a specified period of time. Such measures could help to increase transparency of participatory sensing applications.

Finally, enhanced legal protections for unshared vault data can encourage participation in participatory sensing. Ongoing work is investigating the possibility of a legal privilege for personal-sensing data. Such a privilege could be enabled by statute and modeled on attorney-client or doctor-patient privilege.

Conclusion

While lawyers and social scientists work on structural changes to help ensure privacy in participatory sensing, many of the initial and critically important steps toward privacy protection will be up to application developers. By innovating to put participants first, we can create systems that respect individuals' needs to control sensitive data. We can also augment people's ability to make sense of such granular data, and engage participants in making decisions about that data over the long term. Through attention to such principles, developers will help to ensure that four billion little brothers are not watching us. Instead, participatory sensing can have a future of secure, willing, and engaged participation.

Acknowledgments

Many thanks to collaborators Jeffrey Burke, Deborah Estrin, and Mark Hansen, whose ideas and contributions have shaped this material.

This article is based upon work supported by the National Science Foundation under Grant No. 0832873. □

Related articles on queue.acm.org

Sensing Everywhere (Video)

Deborah Estrin

http://queue.acm.org/detail_video.cfm?id=1554569

A Conversation with Cory Doctorow and Hal Stern

<http://queue.acm.org/detail.cfm?id=1242497>

Logging on with KV

Kode Vicious

<http://queue.acm.org/detail.cfm?id=1142039>

References

1. Cáceres, R., Cox, L., Lim, H., Shakimov, A., Varshavsky, A. Virtual individual servers as privacy-preserving proxies for mobile devices. In *Proceedings of First ACM SIGCOMM Workshop on Networking, Systems, and Applications on Mobile Handhelds* (Barcelona, Spain, 2009).
2. Cuff, D., Hansen, M., Kang, J. Urban sensing: Out of the woods. *Commun. ACM* 51, 3 (Mar. 2008), 24–33.
3. Eagle, N. Behavioral inference across cultures: Using telephones as a cultural lens. *IEEE Intelligent Systems* 23 (2008), 62–64.
4. Eisenman, S.B., Lane, N.D., Miluzzo, E., Peterson, R.A., Ahn, G.S., Campbell, A.T. MetroSense Project: People-centric sensing at scale. *Proceedings of the ACM Sensys World Sensor Web Workshop* (Boulder, CO, 2006).
5. Google Health; <https://www.google.com/health>.
6. Iachello, G., Hong, J. 2007. End-user privacy in human-computer interaction. *Foundations and Trends in Human-Computer Interaction* 1 (2007), 1–137.
7. Kang, J. Privacy in cyberspace transactions. *Stanford Law Review* 50 (1998), 1193–1294.
8. Lam, M. Building a social networking future without Big Brother; <http://suif.stanford.edu/%7Elam/lam-pomi-ws09.pdf>.
9. Microsoft HealthVault; <http://www.healthvault.com/>.
10. Nissenbaum, H. Privacy as contextual integrity. *Washington Law Review* 79 (2004), 119–158.
11. Palen, L., Dourish, P. Unpacking "privacy" for a networked world. In *Proceedings of 2003 SIGCHI Conference* (Ft. Lauderdale, FL), 129–136.
12. *Personal Privacy in an Information Society: The Report of The Privacy Protection Study Commission*. 1977; <http://epic.org/privacy/ppsc1977report/>.
13. U.S. Department of Health, Education, and Welfare. *Records, Computers, and the Rights of Citizens*. MIT Press, Cambridge, MA, 1973.
14. Waldo, J., Lin, H. S., Millett, L. I. *Engaging Privacy and Information Technology in a Digital Age*. The National Academies Press, Washington, D.C., 2007.
15. Zittrain, J. *The Future of the Internet—And How to Stop It*. Yale University Press, New Haven and London, 2008.

Katie Shilton is a doctoral student in information studies at UCLA. Her research explores privacy and ethical challenges raised by ubiquitous sensing technologies, and she coordinates a research project at the Center for Embedded Networked Sensing focused on these questions.

© 2009 ACM 0001-0782/09/1100 \$10.00

Long considered an afterthought, software maintenance is easiest and most effective when built into a system from the ground up.

BY PAUL STACHOUR AND DAVID COLLIER-BROWN

You Don't Know Jack about Software Maintenance

EVERYONE KNOWS MAINTENANCE is difficult and boring, and therefore avoids doing it. It doesn't help that many pointy-haired bosses (PHBs) say things like:

"No one needs to do maintenance—that's a waste of time."

"Get the software out now; we can decide what its real function is later."

"Do the hardware first, without thinking about the software."

"Don't allow any room or facility for expansion. You can decide later how to sandwich the changes in."

These statements are a fair description of development

during the last boom, and not too far from what many of us are doing today. This is not a good thing: when you hit the first bug, all the time you may have "saved" by ignoring the need to do maintenance will be gone.

During a previous boom, General Electric designed a mainframe that it claimed would be sufficient for all the computer uses in Boston, and would never need to be shut down for repair or for software tweaks. The machine it eventually built wasn't nearly big enough, but it did succeed at running continuously without need for hardware or software changes.

Today we have a distributed network of computers provided by thousands of businesses, sufficient for everyone in at least North America, if not the world. Still, we must keep shutting down individual parts of the network to repair or change the software. We do so because we've forgotten how to do software maintenance.

What is Software Maintenance?

Software maintenance is not like hardware maintenance, which is the return of the item to its original state. Software maintenance involves moving an item away from its original state. It encompasses all activities associated with the process of changing software. That includes everything associated with "bug fixes," functional and performance enhancements, providing backward compatibility, updating its algorithm, covering up hardware errors, creating user-interface access methods, and other cosmetic changes.

In software, adding a six-lane automobile expressway to a railroad bridge is considered maintenance—and it would be particularly valuable if you could do it without stopping the train traffic.

Is it possible to design software so it can be maintained in this way? Yes, it is. So, why don't we?

The Four Horsemen of the Apocalypse

There are four approaches to software



maintenance: traditional, never, discrete, and continuous—or, perhaps, war, famine, plague, and death. In any case, 3.5 of them are terrible ideas.

Traditional (or “everyone’s first project”). This one is easy: don’t even think about the possibility of maintenance. Hard-code constants, avoid subroutines, use all global variables, use short and non-meaningful variable names. In other words, make it difficult to change any one thing without changing everything. Everyone knows examples of this approach—and the PHBs who thoughtlessly push you into it, usually because of schedule pressures.

Trying to maintain this kind of software is like fighting a war. The enemy fights back! It particularly fights back when you have to change interfaces, and you find you’ve only changed some of the copies.

Never. The second approach is to decide upfront that maintenance will never occur. You simply write wonderful programs right from the start. This is actually credible in some embedded systems, which will be burned to ROM and never changed. Toasters, video games, and cruise missiles come to mind.

All you have to do is design perfect specifications and interfaces, and never change them. Change only the implementation, and then only for bug fixes before the product is released. The code quality is wildly better than it is for the traditional approach, but never quite good enough to avoid change completely.

Even for very simple embedded sys-

tems, the specification and designs aren’t quite good enough, so in practice the specification is frozen while it’s still faulty. This is often because it cannot be validated, so you can’t tell if it’s faulty until too late. Then the specification is not adhered to when code is written, so you can’t prove the program follows the specification, much less prove it’s correct. So, you test until the program is late, and then ship. Some months later you replace it as a complete entity, by sending out new ROMs. This is the typical history of video games, washing machines, and embedded systems from the U.S. Department of Defense.

Discrete. The discrete change approach is the current state of practice: define hard-and-fast, highly configuration-controlled interfaces to elements of software, and regularly carry out massive all-at-once changes. Next, ship an entire new copy of the program, or a “patch” that silently replaces entire executables and libraries. (As we write this, a new copy of Open Office is asking us please to download it.)

In theory, the process accepts (reluctantly) the fact of change, keeps a parts list and tools list on every item, allows only preauthorized changes under strict configuration control, and forces all servers/users’ changes to take place in one discrete step. In practice, the program is running multiple places, and each must kick off its users, do the upgrade, and then let them back on again. Change happens more often and in more places than predicted, all the components of an

item are not recorded, and patching is alive (and, unfortunately, thriving) because of the time lag for authorization and the rebuild time for the system.

Furthermore, while official interfaces are controlled, unofficial interfaces proliferate; and with C and older languages, data structures are so available that even when change is desired, too many functions “know” that the structure has a particular layout. When you change the data structure, some program or library that you didn’t even know existed starts to crash or return ENOTSUP. A mismatch between an older Linux kernel and newer glibc once had getuid returning “Operation not supported,” much to the surprise of the recipients.

Experience shows that it is completely unrealistic to expect all users to whom an interface is visible will be able to change at the same time. The result is that single-step changes cannot happen: multiple change interrelationships conflict, networks mean multiple versions are simultaneously current, and owners/users want to control change dates.

Vendors try to force discrete changes, but the changes actually spread through a population of computers in a wave over time. This is often likened to a plague, and is every bit as popular.

Customers use a variant of the “never” approach to software maintenance against the vendors of these plagues: they build a known working configuration, then “freeze and forget.” When an update is required, they build a completely new system from the ground up and freeze it. This works unless you get an urgent security patch, at which time you either ignore it or start a large unscheduled rebuild project.

Continuous change. At first, this approach to maintenance sounds like just running new code willy-nilly and watching what happens. We know at least one company that does just that: a newly logged-on user will unknowingly be running different code from everyone else. If it doesn’t work, the user’s system will either crash or be kicked off by the sysadmin, then will have to log back on and repeat the work using the previous version.

Real-world structure for managing interface changes.

```
struct item_loc_t {
    struct {
        unsigned short major; /* = 1 */
        unsigned short minor; /* = 0 */
    } version;
    unsigned part_no;
    unsigned quantity;
    struct location_t {
        char           state[4];
        char           city[8];
        unsigned warehouse;
        short          area;
        short          pigeonhole;
    } location;
    ...
}
```

However, that is not the real meaning of continuous. The real continuous approach comes from Multics, the machine that was never supposed to shut down and that used controlled, transparent change. The developers understood the only constant is change and that migration for hardware, software, and function during system operation is necessary. Therefore, the ability to change was designed from the very beginning.

Software in particular must be written to evolve as changes happen, using a weakly typed high-level language and, in older programs, a good macro assembler. No direct references are allowed to anything if they can be avoided. Every data structure is designed for expansion and self-identifying as to version. Every code segment is made self-identifying by the compiler or other construction procedure. Code and data are changeable on a per-command/process/system basis, and as few as possible copies of anything are kept, so single copies could be dynamically updated as necessary.

The most important thing is to manage interface changes. Even in the Multics days, it was easy to forget to change every single instance of an interface. Today, with distributed programs, changing all possible copies of an interface at once is going to be insanely difficult, if not flat-out impossible.

Who Does it Right?

BBN Technologies was the first company to perform continuous controlled change when they built the ARPANET backbone in 1969. They placed a 1-bit version number in every packet. If it changed from 0 to 1, it meant that the IMP (router) was to switch to a new version of its software and set the bit to 1 on every outgoing packet. This allowed the entire ARPANET to switch easily to new versions of the software without interrupting its operation. That was very important to the pre-TCP Internet, as it was quite experimental and suffered a considerable amount of change.

With Multics, the developers did all of these good things, the most important of which was the discipline used with data structures: if an interface took more than one parameter,



Software maintenance is not like hardware maintenance, which is the return of the item to its original state. Software maintenance involves moving an item away from its original state.



all the parameters were versioned by placing them in a structure with a version number. The caller set the version, and the recipient checked it. If it was completely obsolete, it was flatly rejected. If it was not quite current, it was processed differently, by being upgraded on input and probably downgraded on return.

This meant that many different versions of a program or kernel module could exist simultaneously, while upgrades took place at the user's convenience. It also meant that upgrades could happen automatically and that multiple sites, multiple suppliers, and networks didn't cause problems.

An example of a structure used by a U.S.-based warehousing company (translated to C from Multics PL/1) is illustrated in the accompanying box. The company bought a Canadian competitor and needed to add inter-country transfers, initially from three of its warehouses in border cities. This, in turn, required the state field to split into two parts:

```
char country_code[4];
char state_province[4];
```

To identify this, the company incremented the version number from 1.0 to 2.0 and arranged for the server to support both types. New clients used version 2.0 structures and were able to ship to Canada. Old ones continued to use version 1.0 structures. When the server received a type 1 structure, it used an "updater" subroutine that copied the data into a type 2 structure and set the country code to U.S.

In a more modern language, you would add a new subclass with a constructor that supports a country code, and update your new clients to use it. The process is this:

1. Update the server.
2. Change the clients that run in the three border-state warehouses. Now they can move items from U.S. to Canadian warehouses.
3. Deploy updated clients to those Canadian locations needing to move stock.
4. Update all of the U.S.-based clients at their leisure.

Using this approach, there is never a need to stop the whole system, only the individual copies, and that can be

scheduled around a business's convenience. The change can be immediate, or can wait for a suitable time.

Once the client updates have occurred, we simultaneously add a check to produce a server error message for anyone who accidentally uses an outdated U.S.-only version of the client. This check is a bit like the "can't happen" case in an `else-if`: it's done to identify impossibly out-of-date calls. It fails conspicuously, and the system administrators can then hunt down and replace the ancient version of the program. This also discourages the unwise from permanently deferring fixes to their programs, much like the coarse version numbers on entire programs in present practice.

Modern Examples

This kind of fine-grain versioning is sometimes seen in more recent programs. Linkers are an example, as they read files containing numbered records, each of which identifies a particular kind of code or data. For example, a record number 7 might contain the information needed to link a subroutine call, containing items such as the name of the function to call and a space for an address. If the linker uses record types 1 through 34, and later needs to extend 7 for a new compiler, then create a type 35, use it for the new compiler, and schedule changes from type 7 to type 35 in all the other compilers, typically by announcing the date on which type 7 records would no longer be accepted.

Another example is in networking protocols such as IBM SMB (Server Message Block), used for Windows networking. It has both protocol versions and packet types that can be used exactly the same way as the record types of a linker.

Object languages can also support controlled maintenance by creating new versions as subclasses of the same parent. This is a slightly odd use of a subclass, as the variations you create aren't necessarily meant to persist, but you can go back and clean out unneeded variants later, after they're no longer in use.

With AJAX, a reasonably small client can be downloaded every time the program is run, thus allowing change without versioning. A larger client

would need only a simple versioning scheme, enough to allow it to be downloaded whenever it was out of date.

An elegant modern form of continuous maintenance exists in relational databases: one can always add columns to a relation, and there is a well-known value called `null` that stands for "no data." If the programs that use the database understand that any calculation with a `null` yields a `null`, then a new column can be added, programs changed to use it over some period of time, and the old column(s) filled with `null`s. Once all the users of the old column are no more, as indicated by the column being `null` for some time, then the old column can be dropped.

Another elegant mechanism is a markup language such as SGML or XML, which can add or subtract attributes of a type at will. If you're careful to change the attribute name when the type changes, and if your XML processor understands that adding 3 to a `null` value is still `null`, you've an easy way to transfer and store mutating data.

Maintenance Isn't Hard, It's Easy

During the last boom, (author) Collier-Brown's team needed to create a single front end to multiple back ends, under the usual insane time pressures. The front end passed a few parameters and a C structure to the back ends, and the structure repeatedly needed to be changed for one or another of the back ends as they were developed.

Even when all the programs were on the same machine, the team couldn't change them simultaneously because they would have been forced to stop everything they were doing and apply a structure change. Therefore, the team started using version numbers. If a back end needed version 2.6 of the structure, it told the front end, which handed it the new one. If it could use only version 2.5, that's what it asked for. The team never had a "flag day" when all work stopped to apply an interface change. They could make those changes when they could schedule them.

Of course, the team did have to make the changes eventually, and

their management had to manage that, but they were able to make the changes when it wouldn't destroy our schedule. In an early precursor to test-directed design, they had a regression test that checked whether all the version numbers were up to date and warned them if updates were needed.

The first time the team avoided a flag day, they gained the few hours expended preparing for change. By the 12th time, they were winning big.

Maintenance really is easy. More importantly, investing time to prepare for it can save you and your management time in the most frantic of projects. 

Related articles on queue.acm.org

[The Meaning of Maintenance](#)

Kode Vicious

<http://queue.acm.org/detail.cfm?id=1594861>

[The Long Road to 64 Bits](#)

John Mashey

<http://queue.acm.org/detail.cfm?id=1165766>

[A Conversation with David Brown](#)

<http://queue.acm.org/detail.cfm?id=1165764>

Paul Stachour is a software engineer equally at home in development, quality assurance, and process. One of his focal areas is how to create correct, reliable, functional software in effective and efficient ways in many programming languages. Most of his work has been with life-, safety-, and security-critical applications from his home base in the Twin Cities of Minnesota.

David Collier-Brown is an author and systems programmer, formerly with Sun Microsystems, who mostly does performance and capacity work from his home in Toronto.

You've come a long way. Share what you've learned.



ACM has partnered with MentorNet, the award-winning nonprofit e-mentoring network in engineering, science and mathematics. MentorNet's award-winning **One-on-One Mentoring Programs** pair ACM student members with mentors from industry, government, higher education, and other sectors.

- Communicate by email about career goals, course work, and many other topics.
- Spend just **20 minutes a week** - and make a huge difference in a student's life.
- Take part in a lively online community of professionals and students all over the world.



Make a difference to a student in your field.

Sign up today at: www.mentor.net

Find out more at: www.acm.org/mentor.net

MentorNet's sponsors include 3M Foundation, ACM, Alcoa Foundation, Agilent Technologies, Amylin Pharmaceuticals, Bechtel Group Foundation, Cisco Systems, Hewlett-Packard Company, IBM Corporation, Intel Foundation, Lockheed Martin Space Systems, National Science Foundation, Naval Research Laboratory, NVIDIA, Sandia National Laboratories, Schlumberger, S.D. Bechtel, Jr. Foundation, Texas Instruments, and The Henry Luce Foundation.

contributed articles

DOI:10.1145/1592761.1592779

“Digital fluency” should mean designing, creating, and remixing, not just browsing, chatting, and interacting.

BY MITCHEL RESNICK, JOHN MALONEY, ANDRÉS MONROY-HERNÁNDEZ, NATALIE RUSK, EVELYN EASTMOND, KAREN BRENNAN, AMON MILLNER, ERIC ROSENBAUM, JAY SILVER, BRIAN SILVERMAN, AND YASMIN KAFAI

Scratch: Programming for All

WHEN MOSHE Y. VARDI, Editor-in-Chief of *Communications*, invited us to submit an article, he recalled how he first learned about Scratch: “A colleague of mine (CS faculty),” he said, “told me how she tried to get her 10-year-old daughter interested in programming, and the only thing that appealed to her was Scratch.”

That’s what we were hoping for when we set out to develop Scratch six years ago. We wanted to develop an approach to programming that would appeal to people who hadn’t previously imagined themselves as programmers. We wanted to make it easy for everyone, of all ages, backgrounds, and interests, to program their own interactive stories, games, animations, and simulations, and share their creations with one another.

Since the public launch in May 2007, the Scratch Web site (<http://scratch.mit.edu>) has become a vibrant online community, with people sharing,

discussing, and remixing one another’s projects. Scratch has been called “the YouTube of interactive media.” Each day, Scratchers from around the world upload more than 1,500 new projects to the site, with source code freely available for sharing and remixing. The site’s collection of projects is wildly diverse, including video games, interactive newsletters, science simulations, virtual tours, birthday cards, animated dance contests, and interactive tutorials, all programmed in Scratch.

The core audience on the site is between the ages of eight and 16 (peaking at 12), though a sizeable group of adults participates as well. As Scratchers program and share interactive projects, they learn important mathematical and computational concepts, as well as how to think creatively, reason systematically, and work collaboratively: all essential skills for the 21st century. Indeed, our primary goal is not to prepare people for careers as professional programmers but to nurture a new generation of creative, systematic thinkers comfortable using programming to express their ideas.

In this article, we discuss the design principles that guided our development of Scratch and our strategies for making programming accessible and engaging for everyone. But first, to give a sense of how Scratch is being used, we describe a series of projects developed by a 13-year-old girl with the Scratch screen name BalaBethany.

BalaBethany enjoys drawing anime characters. So when she started using Scratch, it was natural for her to program animated stories featuring these characters. She began sharing her projects on the Scratch Web site, and other members of the community responded positively, posting glowing comments under her projects (such as “Awesome!” and “OMG I LUV IT!!!!!!”), along with questions about how she achieved certain visual effects (such as “How do you make a sprite look see-through?”). Encouraged, BalaBethany then created and shared new Scratch projects on a regular basis, like episodes in a TV series.



ask What's your name? and wait

say join Hello answer for 2 secs

y position -150

wait until touching > 80

switch to costume jump ▾

say game over





Figure 1. Screenshots from BalaBethany's anime series, contest, and tutorial.

She periodically added new characters to her series and at one point asked why not involve the whole Scratch community in the process? She created and uploaded a new Scratch project that announced a “contest,” asking other community members to design a sister for one of her characters (see Figure 1). The project listed a set of requirements for the new character, including “Must have red or blue hair, please choose” and “Has to have either cat or ram horns, or a combo of both.”

The project received more than 100 comments. One was from a community member who wanted to enter the contest but said she didn’t know how to draw anime characters. So BalaBethany produced another Scratch project,

a step-by-step tutorial, demonstrating a 13-step process for drawing and coloring anime characters.

Over the course of a year, BalaBethany programmed and shared more than 200 Scratch projects, covering a range of project types (stories, contests, tutorials, and more). Her programming and artistic skills progressed, and her projects clearly resonated with the Scratch community, receiving more than 12,000 comments.

Why Programming?

It has become commonplace to refer to young people as “digital natives” due to their apparent fluency with digital technologies.¹⁵ Indeed, many young people are very comfortable sending text messages, playing online games, and browsing the Web. But does that really make them fluent with new technologies? Though they interact with digital media all the time, few are able to create their own games, animations, or simulations. It’s as if they can “read” but not “write.”

As we see it, digital fluency requires not just the ability to chat, browse, and interact but also the ability to design, create, and invent with new media,¹⁶ as BalaBethany did in her projects. To do so, you need to learn some type of programming. The ability to program provides important benefits. For example, it greatly expands the range of what you

can create (and how you can express yourself) with the computer. It also expands the range of what you can learn. In particular, programming supports “computational thinking,” helping you learn important problem-solving and design strategies (such as modularization and iterative design) that carry over to nonprogramming domains.¹⁸ And since programming involves the creation of external representations of your problem-solving processes, programming provides you with opportunities to reflect on your own thinking, even to think about thinking itself.²

Previous Research

When personal computers were first introduced in the late 1970s and 1980s, there was initial enthusiasm for teaching all children how to program. Thousands of schools taught millions of students to write simple programs in Logo or Basic. Seymour Papert’s 1980 book *Mindstorms*¹³ presented Logo as a cornerstone for rethinking approaches to education and learning. Though some children and teachers were energized and transformed by these new possibilities, most schools soon shifted to other uses of computers. Since that time, computers have become pervasive in children’s lives, but few learn to program. Today, most people view computer programming as a narrow, technical activity, appropriate for only

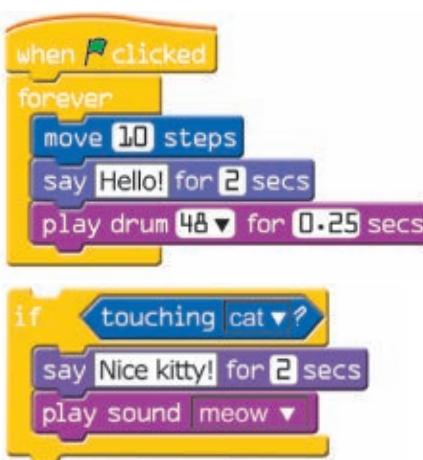


Figure 2. Sample Scratch scripts.

a small segment of the population.

What happened to the initial enthusiasm for introducing programming to children? Why did Logo and other initiatives not live up to their early promise? There were several factors:

- Early programming languages were too difficult to use, and many children simply couldn't master the syntax of programming;

- Programming was often introduced with activities (such as generating lists of prime numbers and making simple line drawings) that were not connected to young people's interests or experiences; and

- Programming was often introduced in contexts where no one could provide guidance when things went wrong—or encourage deeper explorations when things went right.

Papert argued that programming languages should have a "low floor" (easy to get started) and a "high ceiling" (opportunities to create increasingly complex projects over time). In addition, languages need "wide walls" (supporting many different types of projects so people with many different interests and learning styles can all become engaged). Satisfying the triplet of low-floor/high-ceiling/wide-walls hasn't been easy.³

In recent years, new attempts have sought to introduce programming to children and teens.⁷ Some use professional programming languages like Flash/ActionScript; others use new languages (such as Alice⁷ and Squeak Etoys⁵) developed specifically for younger programmers. They have inspired and informed our work on Scratch. But we weren't fully satisfied with the existing options. In particular, we felt it was important to make the floor even lower and the walls even wider while still supporting development of computational thinking.

To achieve these goals, we established three core design principles for Scratch: Make it more tinkerable, more meaningful, and more social than other programming environments. In the following sections, we discuss how each of these principles guided our design of Scratch.

More Tinkerable

Our Lifelong Kindergarten research group at the MIT Media Lab (<http://llk.media.mit.edu>) has worked closely with the Lego Company (<http://www.lego.com/>) for many years, helping develop Lego Mindstorms and other robotics kits.¹⁷ We have always been intrigued and inspired by the way children play and build with Lego bricks. Given a box full of them, they immediately start tinkering, snapping together a few bricks, and the emerging structure then gives them new ideas. As they play and build, plans and goals evolve organically, along with the structures and stories.

We wanted the process of programming in Scratch to have a similar feel. The Scratch grammar is based on a collection of graphical "programming blocks" children snap together to create programs (see Figure 2). As with Lego bricks, connectors on the blocks suggest how they should be put together. Children can start by simply tinkering with the bricks, snapping them together in different sequences and combinations to see what happens. There is none of the obscure syntax or punctuation of traditional programming languages. The floor is low and the experience playful.

Scratch blocks are shaped to fit together only in ways that make syntactic sense. Control structures (like forever and repeat) are C-shaped to suggest that blocks should be placed inside them. Blocks that output values are shaped according to the types of values they return: ovals for numbers and hexagons for Booleans. Conditional blocks (like if and repeat-until)

have hexagon-shaped voids, indicating a Boolean is required.

The name "Scratch" itself highlights the idea of tinkering, as it comes from the scratching technique used by hip-hop disc jockeys, who tinker with music by spinning vinyl records back and forth with their hands, mixing music clips together in creative ways. In Scratch programming, the activity is similar, mixing graphics, animations, photos, music, and sound.

Scratch is designed to be highly interactive. Just click on a stack of blocks and it starts to execute its code immediately. You can even make changes to a stack as it is running, so it is easy to experiment with new ideas incrementally and iteratively. Want to create parallel threads? Simply create multiple stacks of blocks. Our goal is to make parallel execution as intuitive as sequential execution.

The scripting area in the Scratch interface is intended to be used like a physical desktop (see Figure 3). You can even leave extra blocks or stacks lying around in case you need them later. The implied message is that it's OK to be a little messy and experimental. Most programming languages (and computer science courses) privilege top-down planning over bottom-up tinkering. With Scratch, we want tinkerers to feel just as comfortable as planners.

The emphasis on iterative, incremental design is aligned with our own development style in creating Scratch. We selected Squeak as an implementation language since it is well-suited for



Figure 3. Scratch user interface.



Figure 4. Screenshots from sample Scratch projects.

rapid prototyping and iterative design. Before we launched Scratch in 2007, we continually field-tested prototypes in real-world settings, revising over and over based on feedback and suggestions from the field.⁴

More Meaningful

We know that people learn best, and enjoy most, when working on personally meaningful projects. So in developing Scratch, we put a high priority on two design criteria:

Diversity. Supporting many different types of projects (stories, games, animations, simulations), so people with

widely varying interests are all able to work on projects they care about; and

Personalization. Making it easy for people to personalize their Scratch projects by importing photos and music clips, recording voices, and creating graphics.¹⁴

These priorities influenced many of our design decisions. For example, we decided to focus on 2D images, rather than 3D, since it is much easier for people to create, import, and personalize 2D artwork. While some people might see the 2D style of Scratch projects as somewhat outdated, Scratch projects collectively exhibit a visual diversity and

personalization missing from 3D authoring environments.

The value of personalization is captured nicely in this blog post from a computer scientist who introduced Scratch to his two children: “I have to admit that I initially didn’t get why a kids’ programming language should be so media-centric, but after seeing my kids interact with Scratch it became much more clear to me. One of the nicest things I saw with Scratch was that it personalized the development experience in new ways by making it easy for my kids to add personalized content and actively participate in the development process. Not only could they develop abstract programs to do mindless things with a cat or a box, etc... but they could add their own pictures and their own voices to the Scratch environment, which has given them hours of fun and driven them to learn.”

We continue to be amazed by the diversity of projects that appear on the Scratch Web site. As expected, there are lots of games, ranging from painstakingly recreated versions of favorite video games (such as *Donkey Kong*) to totally original games. But there are many other genres, too (see Figure 4). Some Scratch projects document life experiences (such as a family vacation in Florida); others document imaginary wished-for experiences (such as a trip to meet other Scratchers). Some Scratch

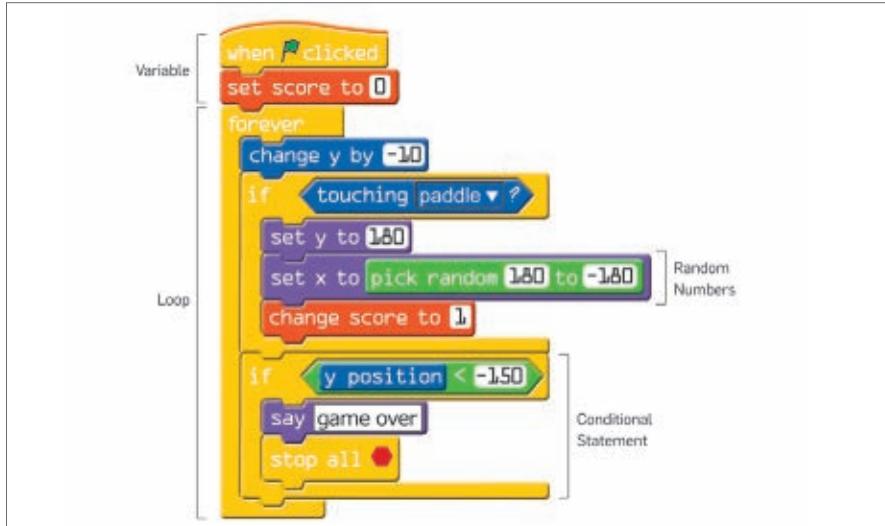


Figure 5. Sample Scratch script (from Pong-like paddle game) highlighting computational and mathematical concepts.

projects (such as birthday cards and messages of appreciation) are intended to cultivate relationships. Others are designed to raise awareness on social issues (such as global warming and animal abuse). During the 2008 U.S. presidential election, a flurry of projects featured Barack Obama and John McCain and later a series of projects promoted members of the Scratch online community for the not-quite-defined position of "President of Scratch."

Some Scratch projects grow out of school activities. For an Earth-science class, a 13-year-old boy from India created a project in which an animated character travels to the center of the Earth, with a voice-over describing the different layers along the way. As part of a social-studies class, a 14-year-old boy from New Jersey created a simulation of life on the island of Rapa Nui, designed to help others learn about the local culture and economy.

As Scratchers work on personally meaningful projects, we find they are ready and eager to learn important mathematical and computational concepts related to their projects (see Figure 5). Consider Raul, a 13-year-old boy who used Scratch to program an interactive game in his after-school center.⁹ He created the graphics and basic actions for the game but didn't know how to keep score. So when a researcher on our team visited the center, Raul asked him for help. The researcher showed Raul how to create a variable in Scratch, and Raul immediately saw how he could use it for keeping score. He began playing with the blocks for incrementing variables, then reached out and shook the researcher's hand, saying "Thank you, thank you, thank you." The researcher wondered how many eighth-grade algebra teachers get thanked by their students for teaching them about variables?

More Social

Development of the Scratch programming language is tightly coupled with development of the Scratch Web site.¹² For Scratch to succeed, the language needs to be linked to a community where people can support, collaborate, and critique one another and build on one another's work.¹

The concept of sharing is built into the Scratch user interface, with a prominent "Share" menu and icon at the top of the screen. Click the Share icon and your project is uploaded to the Scratch Web site (see Figure 6) where it is displayed at the top of the page, along with the "Newest Projects." Once a project is on the Web site, anyone can run it in a browser (using a Java-based player), comment on it, vote for it (by clicking the "Love It?" button), or download it to view and revise the scripts. (All projects shared on the site are covered by Creative Commons license.)



Three core design principles for Scratch: Make it more tinkerable, more meaningful, and more social than other programming environments.



In the 27 months following the Scratch launch, more than 500,000 projects were shared on the Scratch Web site. For many Scratchers, the opportunity to put their projects in front of a large audience—and receive feedback and advice from other Scratchers—is strong motivation. The large library of projects on the site also serves as inspiration. By exploring projects there, Scratchers get ideas for new projects and learn new programming techniques. Marvin Minsky once said that Logo had a great grammar but not much literature.¹¹ Whereas young writers are often inspired by reading great works of literature, there was no analogous library of great Logo projects to inspire young programmers. The Scratch Web site is the beginning of a "literature" for Scratch.

The site is also fertile ground for collaboration. Community members are constantly borrowing, adapting, and building on one another's ideas, images, and programs. Over 15% of the projects there are remixes of other projects on the site. For example, there are dozens of versions of the game Tetris, as Scratchers continue to add new features and try to improve gameplay. There are also dozens of dress-up-doll projects, petitions, and contests, all adapted from previous Scratch projects.

At first, some Scratchers were upset when their projects were remixed, complaining that others were "stealing" from them. That led to discussions on the Web site's forums about the value of sharing and the ideas behind open source communities. Our goal is to create a culture in which Scratchers feel proud, not upset, when their projects are adapted and remixed by others. We have continually added new features to the site to support and encourage this mind-set. Now, when someone remixes

a project, the site automatically adds a link back to the original project, so the original author gets credit. Also, each project includes links to its “derivatives” (projects remixed from it), and the “Top Remixed” projects are featured prominently on the Scratch homepage.

Some projects focus on the site itself, providing reviews and analyses of other projects there. One early example was called SNN, for Scratch News Network, featuring the Scratch cat (the default character in Scratch) delivering news about the Scratch community, much like a CNN anchor. At first, we saw it as a “simulated newscast” but then realized it was a real newscast, providing news of interest to a real community—the Scratch online community. The SNN project inspired others, leading to a proliferation of online newsletters, magazines, and TV shows, all programmed in Scratch, reporting on the Scratch community.

Other Scratchers formed online “companies,” working together to create projects that their individual members could not have produced on their own. One company got its start when a 15-year-old girl from England, with screen name BeeBop, created a project full of animated sprites and encouraged others to use them in their projects or place special requests for custom-made sprites. She was setting up a no-fee consulting business. A 10-year-old girl, also from England, with screen name MusicalMoon, liked BeeBop’s animations and asked if she’d be willing to create a background for one of her projects. This collaboration gave rise to Mesh Inc., a self-proclaimed “miniature company” to produce “top quality games” in Scratch. A few days later, a 14-year-old boy from New Jersey, screen name Hobbit, discovered the Mesh Inc. gallery and offered his services, saying, “I’m a fairly good programmer, and I could help with debugging and stuff.” Later, an 11-year-old boy from Ireland, with screen name Marty, was added to the Mesh Inc. staff due to his expertise in scrolling backgrounds.

Such collaborations open opportunities for many different types of learning. Here’s how a 13-year-old girl from California, who started a Scratch company called Blue Elk Productions, described her experience:

“What is fun about Scratch and

The Scratch Web site has become a vibrant online community, with people sharing, discussing, and remixing one another’s projects.

about organizing a company to write games together is that I’ve made a lot of friends and learned lots of new things. I’ve learned a lot about different kinds of programming by looking at other games with interesting effects, downloading them, and looking at and modifying the scripts and sprites. I really like programming! Also, when I started with Scratch I didn’t think I was a very good artist. But since then, just by looking at other people’s art projects, asking them questions, and practicing drawing using programs like Photoshop and the Scratch paint editor, I’ve gotten a lot better at art... Another thing I’ve learned while organizing Blue Elk is how to help keep a group of people motivated and working together... I like Scratch better than blogs or social networking sites like Facebook because we’re creating interesting games and projects that are fun to play, watch, and download. I don’t like to just talk to other people online, I like to talk about something creative and new.”

To encourage international sharing and collaboration, we’ve placed a high priority on translating Scratch into multiple languages. We created an infrastructure that allows the Scratch programming blocks to be translated into any language with any character set. A global network of volunteers has provided translations for more than 40 languages. Children around the world now share Scratch projects with one another, each viewing the Scratch programming blocks in their own language.

Future Directions

A growing number of K-12 schools around the world, and even some universities (including Harvard and the University of California, Berkeley),⁸ use Scratch as a first step into programming. A natural question is What comes next? In the Scratch discussion forums, there are ongoing debates about what programming language should be used after Scratch. We receive many requests to add more advanced features to Scratch (such as object inheritance and recursive list structures), hoping that Scratch itself could be the “next step.”

We plan to keep our primary focus on lowering the floor and widening the walls, not raising the ceiling. For some Scratchers, especially those who want to pursue a career in programming or com-

puter science, it is important to move on to other languages. But for many other Scratchers, who see programming as a medium for expression, not a path toward a career, Scratch is sufficient for their needs. With Scratch, they can continue to experiment with new forms of self-expression, producing a diverse range of projects while deepening their understanding of a core set of computational ideas. A little bit of programming goes a long way.

As we develop future versions, our goal is to make Scratch even more tinkerable, meaningful, and social. With our Scratch Sensor Board (http://info.scratch.mit.edu/Sensor_Boards), people can create Scratch projects that sense and react to events in the physical world. We are also developing a version of Scratch that runs on mobile devices and a Web-based version that enables people to access online data and program online activities.

Probably the biggest challenges for Scratch are not technological but cultural and educational.¹⁰ Scratch has been a

success among early adopters, but we need to provide better educational support for it to spread more broadly. We recently launched a new online community, called Scratch-Ed (<http://scratch-ed.media.mit.edu>), where educators share their ideas, experiences, and lesson plans for Scratch. More broadly, there needs to be a shift in how people think about programming, and about computers in general. We need to expand the notion of "digital fluency" to include designing and creating, not just browsing and interacting. Only then will initiatives like Scratch have a chance to live up to their full potential.

Acknowledgments

Many people have contributed to the development of Scratch and even more to the ideas underlying Scratch. We'd like to thank friends and former members of the Lifelong Kindergarten group who have worked on Scratch, especially Tammy Stern, Dave Feinberg, Han Xu, Margarita Dekoli, Leo Burd, Oren Zuckerman, Nick Bushak, and Paula Bonta.

We are grateful to Kylie Peppler, Grace Chui, and other members of Yasmin Kafai's research team, who conducted and participated in field studies in Scratch's early development. Scratch was deeply influenced and inspired by the work of Seymour Papert and Alan Kay. We appreciate financial support from the National Science Foundation (grant ITR-0325828), Microsoft, Intel Foundation, Nokia, and MIT Media Lab research consortia. The names of all children mentioned here are pseudonyms. □

References

1. Bransford, J., Brown, A., and Cocking, R. *How People Learn: Mind, Brain, Experience, and School*. National Academies Press, Washington, D.C., 2000.
2. diSessa, A. *Changing Minds: Computers, Learning, and Literacy*. MIT Press, Cambridge, MA, 2000.
3. Guzdial, M. Programming environments for novices. In *Computer Science Education Research*, S. Fincher and M. Petre, Eds. Taylor & Francis, Abingdon, U.K., 2004, 127–154.
4. Kafai, Y., Peppler, K., and Chiu, G. High-tech programmers in low-income communities: Seeding reform in a community technology center. In *Communities and Technologies*, C. Steinfield, B. Pentland, M. Ackerman, and N. Contractor, Eds. Springer, New York, 2007, 545–564.
5. Kay, A. Squeak etoys, children, and learning; <http://www.squeakland.org/resources/articles>.
6. Kelleher, C. and Pausch, R. Using storytelling to motivate programming. *Commun. ACM* 50, 7 (July 2007), 58–64.
7. Kelleher, C. and Pausch, R. Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *ACM Computing Surveys* 37, 2 (June 2005), 83–137.
8. Malan, D. and Leitner, H. Scratch for budding computer scientists. *ACM SIGCSE Bulletin* 39, 1 (Mar. 2007), 223–227.
9. Maloney, J., Peppler, K., Kafai, Y., Resnick, M., and Rusk, N. Programming by choice: Urban youth learning programming with Scratch. *ACM SIGCSE Bulletin* 40, 1 (Mar. 2008), 367–371.
10. Margolis, J. *Stuck in the Shallow End: Education, Race, and Computing*. MIT Press, Cambridge, MA, 2008.
11. Minsky, M. *Introduction to LogoWorks*. In *LogoWorks: Challenging Programs in Logo*, C. Solomon, M. Minsky, and B. Harvey, Eds. McGraw-Hill, New York, 1986.
12. Monroy-Hernández, A. and Resnick, M. Empowering kids to create and share programmable media. *Interactions* 15, 2 (Mar.–Apr. 2008), 50–53.
13. Papert, S. *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books, New York, 1980.
14. Peppler, K. and Kafai, Y. From SuperGoo to Scratch: Exploring creative media production in informal learning. *Journal on Learning, Media, and Technology* 32, 7 (2007), 149–166.
15. Prensky, M. Digital natives, digital immigrants. *On the Horizon* 9, 5 (Oct. 2001), 1–6.
16. Resnick, M. Sowing the seeds for a more creative society. *Learning and Leading with Technology* (Dec. 2007), 18–22.
17. Resnick, M. Behavior construction kits. *Commun. ACM* 36, 7 (July 1993), 64–71.
18. Wing, J. Computational thinking. *Commun. ACM* 49, 3 (Mar. 2006), 33–35.

Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, and Jay Silver are all researchers and members of the Scratch Team (<http://scratch.mit.edu>) at the Media Laboratory of the Massachusetts Institute of Technology, Cambridge, MA. **Brian Silverman** is president of the Playful Invention Company, Montreal, Quebec, Canada. **Yasmin Kafai** is a professor in the Graduate School of Education of the University of Pennsylvania, Philadelphia, PA.

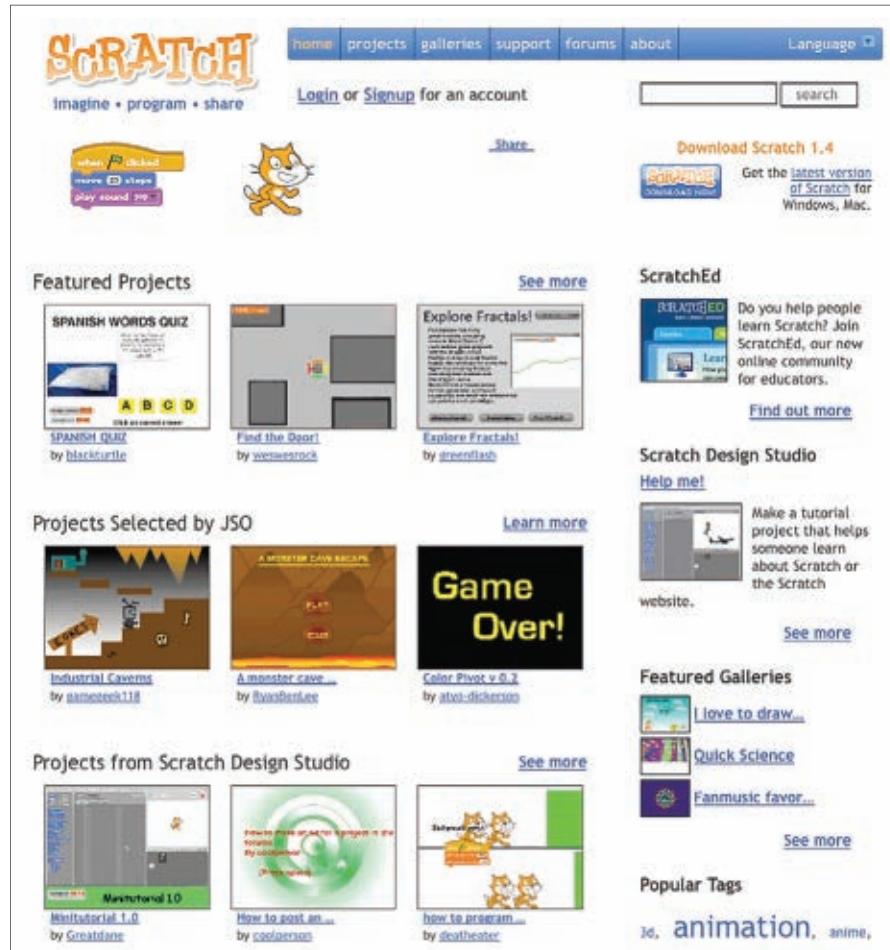


Figure 6. Scratch Web site.

contributed articles

DOI:10.1145/1592761.1592780

Proposed contracts tend to be overpriced because insurers are unable to anticipate customers' secondary losses.

BY TRIDIB BANDYOPADHYAY, VIJAY S. MOOKERJEE, AND RAM C. RAO

Why IT Managers Don't Go for Cyber-Insurance Products

DESPITE POSITIVE EXPECTATIONS, cyber-insurance products have failed to take center stage in the management of IT security risk. Market inexperience, leading to conservatism in pricing cyber-insurance instruments, is often cited as the primary reason for the limited growth of the cyber-insurance market. In contrast, here we provide a demand-side explanation for why cyber-insurance products have not lived up to their initial expectations. We highlight the presence of information asymmetry between customers and providers, showing how it leads to overpricing cyber-

insurance contracts and helps explain why cyber insurance might have failed to deliver its promise as a cornerstone of IT security-management programs.

Technological controls often lag hackers' skills at circumvention. As a result, residual IT security risks cannot be completely eliminated through technological advancement alone. Investment models⁹ of information security suggest that residual IT security risks are transferable to a willing party through cyber insurance. Academic research² also corroborates the economic value of cyber insurance in managing the cyber risks integral to a firm's operations. Cyber insurance refers to insurance contracts designed to mitigate liability issues, property loss and theft, data damage, loss of income from network outage and computer failures, Web-site defacement, and cyberextortion.¹² Current cyber-insurance products tend to provide three basic types of coverage: liability arising from theft of data; remediation in response to the breach; and legal and regulatory fines and penalties.¹

The size of the U.S. cyber-insurance market (annual premiums) was expected to reach \$2.5 billion by 2005,¹¹ and insurance giants like AIG and Chubb created numerous cyber-insurance products for managing IT risk. However, IT managers still show little interest in cyber insurance for their risk-management programs; in 2008, the size of cyber-insurance market was estimated at \$450 million.¹ The 2006 CSI/FBI computer crime and security survey⁸ reported that although firms use cyber insurance more than before, the annual rate of increase is not substantial; respondents indicating utilization of cyber-insurance products increased from 25% to 29% between 2005 and 2006.

Scant attack-loss data, lack of product-market experience, and accounting difficulties are the most commonly cited reasons for the market's slow growth. These factors have led to conservatism by providers that err on the safe side by overpricing their products. However, in a competitive market,

overpricing is generally corrected over time, as risks/uncertainties are better understood. However, even after more than a decade of commercialization, cyber-insurance products remain underutilized. Here, we argue that the demand-side problem with cyber insurance is deeper than the supply-side problem. Moreover, unless the former is addressed, it is unlikely to correct itself naturally over time.

We further highlight the difference between the way a cyber-insurance contract is structured and the way it is used by IT managers, exploring the decisions behind a disclosure and an indemnity claim of a breach. We differentiate the types of breach based on the way they affect firms. We also explain how they might alter the contract-intended claiming behavior of IT managers. When insurers are unaware of such off-contract behavior or choose to not incorporate such behavior in pricing their offerings, information asymmetry prevails in cyber-insurance contracts. The result is an overpriced cyber-insurance contract and less risk being transferred.

Disclosure and Claim of a Realized Breach

With the help of an event study, H. Cavusoglu et al.⁶ showed that publicly

disclosed IT security breaches reduce breached firms' stock prices, at least in the short term, because breaches convey questionable health of an IT security program to stakeholders, who then downgrade their risk perception of the firm. Elsewhere, K. Campbell et al.⁵ showed that investors discriminate against the type of breach in valuing a breach's economic effect. It is not surprising that the CSI/FBI computer crime and security survey⁸ found that only a fraction of the realized breaches are publicly disclosed. Firms apparently use discretion in disclosing realized breaches, depending on the requirements of legal compliance, types of breach, professional norms, and accounting materiality.

Suppose there is no regulatory requirement for disclosure. When a firm lacks cyber-insurance coverage, the information flow regarding a realized breach remains strictly internal to the firm (see Figure 1). On the other hand, if the firm has a cyber-insurance contract in place, it is able to claim its losses from a breach, but the claiming process involves additional external organizations. The increased information flow through external firms greatly affects the firm's ability to keep breach information private. Integrating these ideas with insight from H.

Cavusoglu et. al.⁶ and K. Campbell et. al.,⁵ consider the following observations about claiming indemnity from IT security breaches:

The grapevine. Word of an undisclosed breach can reach stakeholders indirectly via interorganizational grapevines and independent analysts;

Stakeholder perception. As a subsequent effect of the breach, a firm may also suffer secondary loss in terms of reduced stakeholder (investors and customers) valuation; and

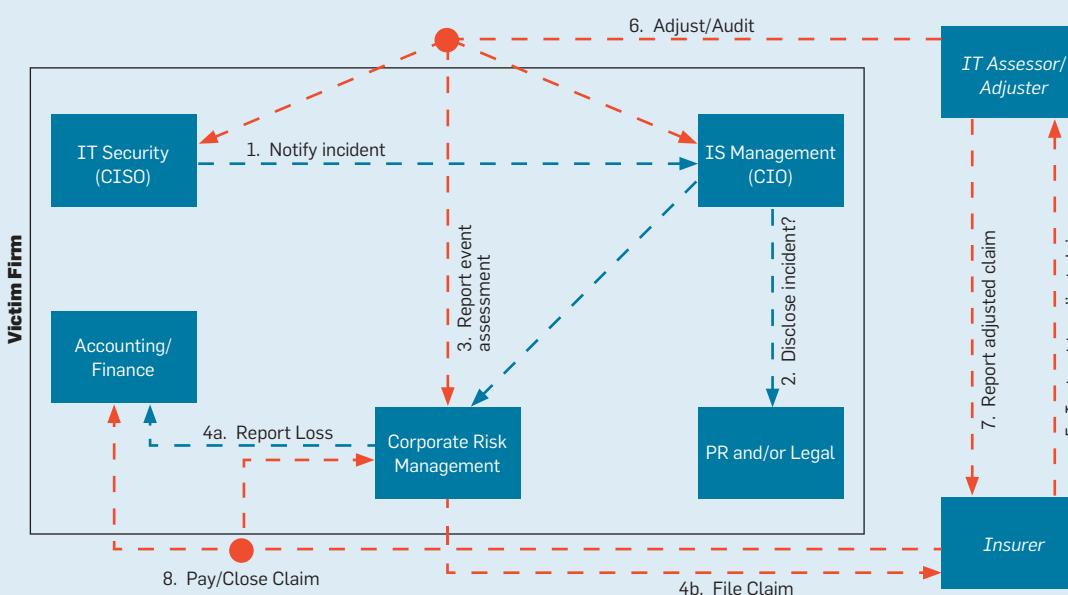
Managers' decisions. Because breach information might trigger further secondary losses, IT managers' decisions (whether or not to file a claim) depend on the primary and secondary losses, as weighed against the contract's potential indemnity payout.

Breaches and Losses

Because the process of reclamation through cyber-insurance contracts involves compromise, post-breach definitions are pertinent, starting with the breach:

Symptomatic. A breach is symptomatic when a firm is breached through exploitation of firm-specific vulnerabilities (such as hackers in 2005 accessing the T.J. Maxx stores database of customer credit and debit card information, an exploitation of the vul-

Figure 1. Information flow in a cyber-insurance claim process.



nerabilities of the stores' data-storage arrangements). Such compromises suggest questionable health of a firm's security program. Consequently, stakeholders downgrade their perception of the firm's IT security;

Systemic. A systemic breach occurs when the affected firm has no reasonable or even known way to defend itself against a new threat vector, especially when the threat is transmitted through the business networks; for example, in January 2004, the MyDoom virus spread primarily via email and severely slowed or shut down email servers with excess traffic (http://reviews.cnet.com/4520-6600_7-5118745-1.html). In this case, stakeholders do not alter their perception of a firm's IT security for systemic breaches, IT security programs plan only for known threats, and firms are all understood to be part of the internetworked global economy where such unknowns are always possible;

Public. A breach is public if it is publicly observed (such as a Web page being defaced), or an observable distributed denial-of-service attack disables a firm's e-commerce transactions or is disclosed through legal requirements or accounting norms (such as the California disclosure requirement for loss of customer data and accounting material loss). By this definition, breaches that are not made public are private.

Here are the potential losses:

Primary. Breaches lead to primary loss (such as direct loss of information or data and operating loss). As an uncontrollable first-degree effect arising from the unuse, disuse, abuse, and misuse of information assets, a primary loss arises under all breach scenarios, or under all combinations of public/private and systemic/symptomatic breaches; and

Secondary. A secondary loss is a second-degree effect, indirectly triggered by information concerning a firm's security inflicting further losses¹⁴ un-

der certain contingent scenarios (see Figure 2). Such losses include indirectly lost or diminished reputation, goodwill, consumer confidence, competitive strength, credit rating, and/or customer churn.

With a cyber-insurance contract in place, the compromised firm claims the primary loss from a public breach, though the secondary loss occurs anyway. The firm also claims its losses for systemic breaches but does not incur secondary loss. For a private symptomatic breach, the secondary loss could occur but only if the firm chooses to file a claim, as in Figure 2. IT managers thus realize that situations could arise where the claiming decision and hence potential indemnity payout must be weighed against the sum of the primary and secondary loss. The secondary losses could be subjectively estimated with the help of extant research outcomes^{5,6,14} or assessed/perceived by experiential or other benchmarking processes by the firm's IT managers.

Armed with such foresight, the managers could revise the deductible for the optimal cyber-insurance contract, in turn influencing the amount of insurance that is purchased.

Altered Claiming Strategy

Considering claiming strategy, we begin with a basic insurance contract characterized by cyber-risk-specific circumstances of potential breach and loss scenarios. The firm faces an arbitrarily distributed primary loss x we assume is transparently known to the insurer. Note that this assumption is significant, as it neutralizes all cited and accepted difficulties of cyber insurance in our treatment. That is, our insight is valid irrespective of the correctness of the friction in the cyber-insurance market.

The insurer presents a deductible (d)-based cyber-insurance contract that can be bought for an up-front premium (P), with the promise that

primary losses greater than d will be compensated in full by the indemnity payment (I) (see Figure 3). The premium P decreases as the deductible d increases (see Figure 4), a standard observation concerning insurance contracts. In practice, the premium P also includes a market-loading factor that takes care of contract-writing costs, as well as other overhead, including profit margins, if there are any. Figures 3 and 4 together depict the presented contract and the contract-intended claiming behavior of the insured firm.

The prospect firm must optimize and communicate a unique optimal deductible (d^*) to the insurer. The deductible then fixes the premium (P^*) to be paid up front. However, an attempt to arrive at a unique optimal deductible d^* must consider and consolidate several scenarios:

Systemic and Public breach. The firm could submit a claim as per the contract, in case it realizes a systemic breach (no secondary loss) or public breach (the secondary loss occurs automatically); and

Symptomatic breach. Because a symptomatic breach suggests lack of awareness, inadequate technology control, failure to observe policy or procedure, lack of manager oversight, or insider breach, one of the following options is pertinent:

- If the firm discloses the breach, it has a symptomatic public breach for which a decision to submit a claim could follow;

- If the firm decides not to disclose the breach, the decision criterion is further binary:

- It could receive a claim for primary losses from the breach but incur the expected secondary loss or

- It might not claim the primary loss, avoid the secondary loss, or forgo the indemnity payout.

- Assume the act of claiming the symptomatic private breach reveals the breach to the stakeholders with probability p , and that the resultant (adverse) revision of the IT security risk costs^a the firm y . Thus the expected

Figure 2. Secondary losses from a realized breach.

	Private breach	Public breach
Symptomatic breach	Depends on claiming decision (Probabilistic?)	Yes
Systemic breach	No	No
Secondary Loss Scenario		

^a Here we assume that an IT security breach yields a fixed amount of downward risk revision by stakeholders. We also separately analyzed the case (not included here) in which

secondary loss is py . Clearly, the firm has no reason to claim for losses up to $r = d + py$. By the second binary criterion, all symptomatic private breaches causing primary loss of magnitude between d and r are now likely to be unclaimed (see Figure 5). Note that r , not the contracted deductible d , is the de facto deductible for the symptomatic private breaches. Assuming that a portion of the realized breaches would be symptomatic private breaches, the unique optimized deductible d^* lies somewhere between d and r ($d < r$). This happens for any arbitrary deductible d the firm might choose. The overall optimized deductible d^* the firm must optimally use is always greater than d . More important, whenever a cyber-insurance contract with an arbitrary deductible d is operationalized at d^* , the insured firm stands to lose part of the expected indemnity payout over the contract horizon; see Figure 5 for the location of this unique deductible.

► Only when the firm faces no secondary loss or symptomatic private breaches (or both), d and d^* coincide, and the insured firm exhibits contract-intended behavior under all circumstances. Two interesting observations follow when a firm selectively uses the contracted or de facto deductible depending on the type of realized breach:

► The higher the secondary loss, the farther apart are d and r , meaning d and d^* are farther apart as well; and

► A greater proportion of symptomatic private breaches over the contract horizon increases the relative frequency when the de facto deductible r (not the contract intended d) are used. This proportionally raises the amount of indemnity to be lost in the process, as in Figure 5.

In effect, IT managers looking toward the contract horizon anticipate too little expected indemnity from cyber-insurance products, so the contract appears overpriced. For the same premium, the firm must use a higher overall deductible (see Figure 6).

Figure 7 outlines the complete cyber-insurance utilization scenario. No behavior and underclaiming behavior

downward risk revision is a function of the magnitude of the primary loss. The fundamental insights from each case are the same.

Figure 3. Relationship among loss, deductible, and indemnity in a cyber-insurance contract.

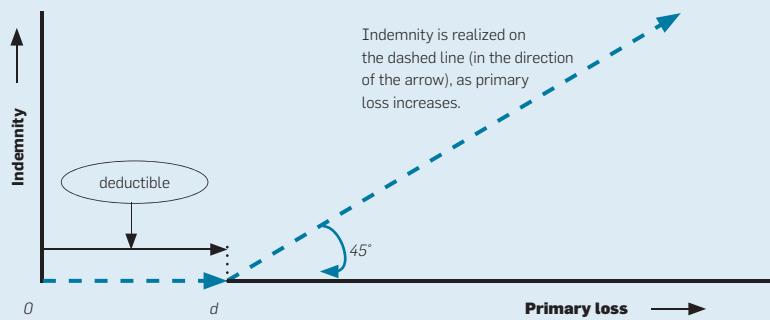


Figure 4. Relationship between premium and deductible in a cyber-insurance contract.

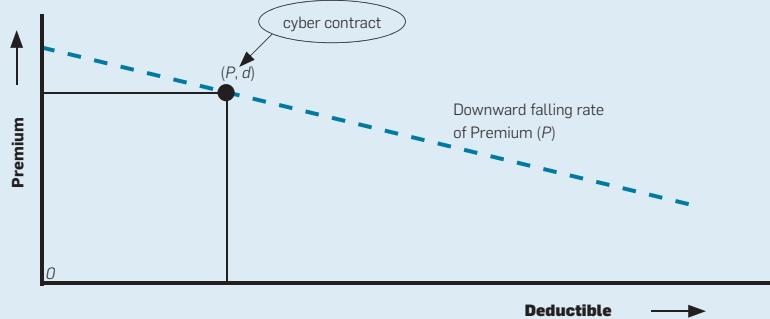


Figure 5. Relationship between de facto deductible r and realized indemnity I .

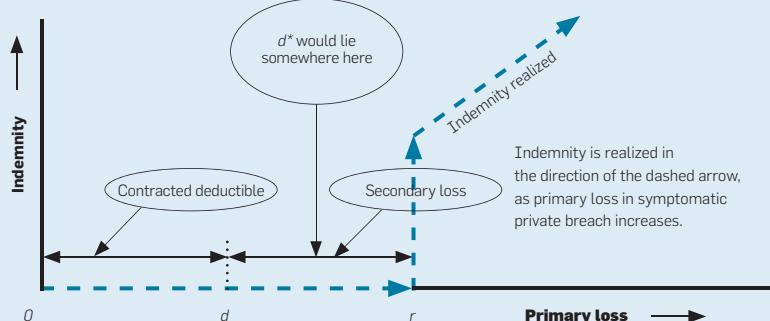
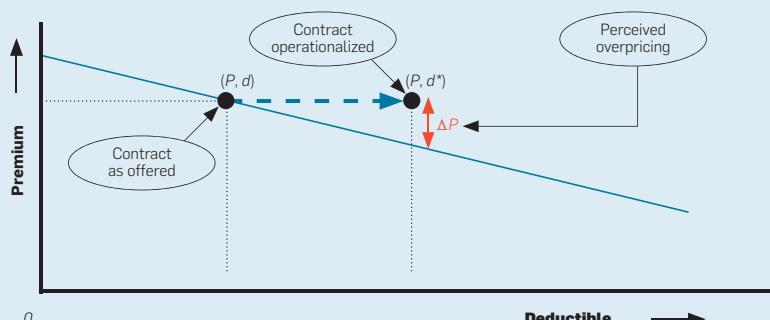
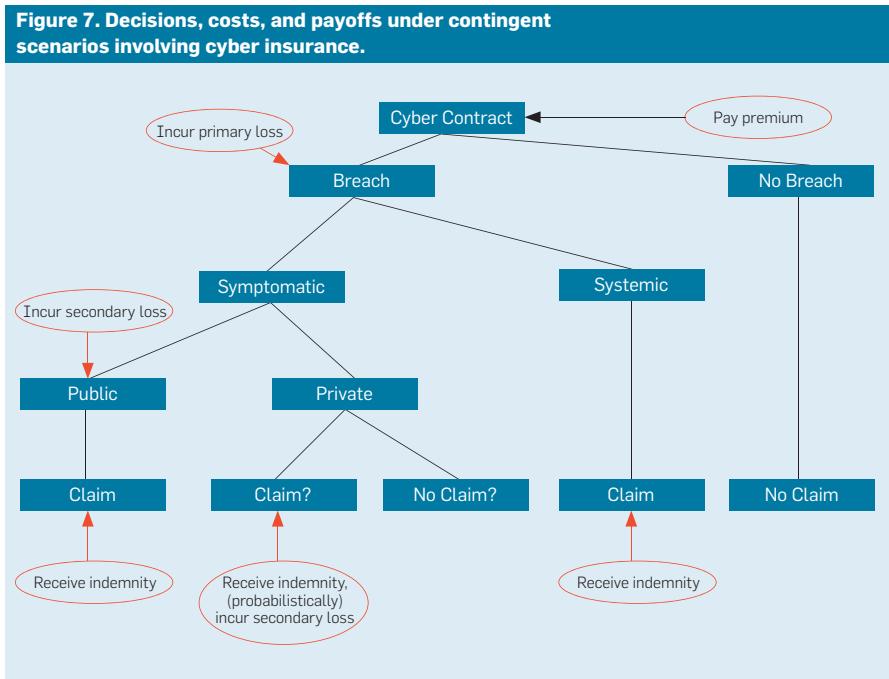


Figure 6. Perceived overpricing of a cyber-insurance contract.



Drawn for uniform primary loss and constant market loading factor

Figure 7. Decisions, costs, and payoffs under contingent scenarios involving cyber insurance.



are thus rational for IT managers under certain circumstances; their lack of interest in cyber-insurance products is rational as well. More important, an underclaiming strategy remains off-contract, possibly heralding information asymmetry between insurers and insured firms in the cyber-insurance market.

Information Asymmetry

Figure 8 outlines how the cyber-insurance market could move through possible scenarios of information asymmetry. Initially, the market could begin in naïve symmetry (quadrant I) where neither the insured nor the insurer knows the existence, nature, or magnitude of the secondary loss. As such, a cyber-insurance contract is written with business prudence in light of other established insurance markets. As the insured firm utilizes information assets in its business processes, the value of asset unuse, disuse, abuse, and misuse become clearer. The insured firm realizes there could be attendant secondary losses following direct losses, as stakeholders reassess the firm's post-breach security. The insured firm now internalizes the ex-post definitions of the types of breach discussed earlier, and managers formalize their optimized claiming strategy for symptomatic private breaches also discussed earlier. This differs from the contract-intended

behavior, and the market moves from naïve symmetry to information asymmetry (quadrant II).

Under information asymmetry, either the insured firm fails to credibly signal its off-contract behavior or the insurer ignores the signal while structuring the cyber-insurance contract. Either way, the market is in a state of information asymmetry, and the insured firm pays for the ensuing inefficiency.

The cyber-insurance market is, in part, locked in a state of information asymmetry. Only when the insurer considers the fact that the insured firm selectively uses the contracted and de facto deductibles when pricing the contract, does the market move to information symmetry (quadrant III). When the insurer corrects its premium structure this way, the contract

is no longer overpriced, and the cyber-insurance product is able to efficiently transfer more IT risk from insured to insurer.

Risk Transfer

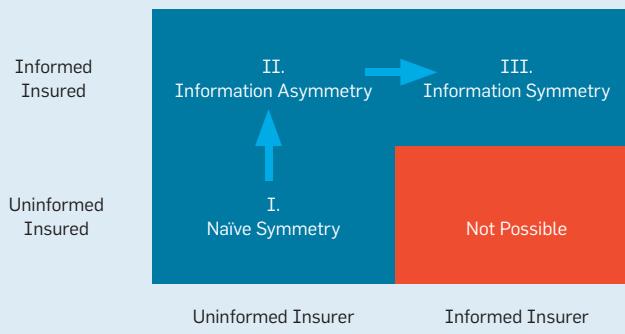
Employing the underclaiming strategy for symptomatic private breaches has a profound effect on cyber insurance as an instrument for transferring IT risk. Applicable for only some realized breaches, it reduces the expected indemnity payout for a given level of premium, causing firms to find the instrument overpriced and hence unattractive. Since firms lack a credible way to communicate their off-contract claiming strategy under current contract provisions, they are forced to pay for information asymmetry.

A detailed analysis of our mathematical model suggests that a cyber-insurance contract optimally transfers a lower amount of IT security risk under information asymmetry. It also suggests that further reducing risk transfer depends on the level of secondary loss. It is important to realize that the major consumers of cyber-insurance products are IT-intensive firms that could face relatively high secondary losses.

Unfortunately, IT-intensive firms also likely find the proposed premium structure overpriced in the presence of information asymmetry, as in Figure 5. On the other hand, firms with low IT security exposure may find cyber-insurance products less overpriced in light of their lower secondary loss.

We have shown that in the presence of secondary loss in symptomatic private breaches, the optimal deductible d^* is between d and r , as in Figure 5. Further analysis shows the smaller

Figure 8. Information asymmetry and market transition.



the ratio of secondary loss to the deductible, the lower is the relative overpricing of a cyber-insurance product. Thus, it appears that managers make a rational choice when using cyber-insurance products with high deductible d such that the effect of relative overpricing on their contracts is minimized. IT managers tend to self-insure the smaller losses yet attempt to provide assurance to their stakeholders of low-probability catastrophic breaches.^b

This analysis suggests that firms with IT-intensive business processes find themselves better off self-insuring a high proportion of their cyber-risk, whereas those with low-intensity IT processes could find cyber-insurance products less pricey under today's market conditions. In light of these outcomes, it becomes apparent why cyber insurance, as a market instrument, has seen little utilization or growth as a financial instrument in managing firms' IT security risk.

Outlook

The cyber-insurance market is characterized by information asymmetry in contracts resulting in the suboptimal transfer of IT risk. From a market perspective, moving to information symmetry (Figure 8, quadrant III) is desirable. Because insured firms pay the price for information asymmetry (quadrant II), a move to information symmetry necessarily increases the utility of the insured firm, with other conditions the same. However, the same may not hold for the insurer. A detailed analysis of the contingency tree (see Figure 7) suggests that under certain conditions (such as significant secondary loss) the insurer is better off under information asymmetry. Under other conditions, the insurer could be better off under information symmetry. This means the insurer would find it beneficial to lower premiums and thus grow the market for cyber insurance.

^b That firms buy cyber insurance with high deductibles was also pointed out by the IT director of a Dallas firm during a discussion with us at the University of Texas, Dallas. He explained that firms often buy cyber insurance to allay investors' fear of major losses from IT security breaches yet depend on the policies, procedures, and technical controls of IT security to manage more frequent but smaller losses.

Firms with a significant amount of IT in their core business processes largely constitute the demand side of the cyber-insurance market. The market is thus relatively homogeneous with respect to (high) secondary loss, and the insurer is better off in a market characterized by information asymmetry. This situation suggests that market mechanisms alone may not produce information symmetry in the cyber-insurance market. Because insured firms likely utilize high levels of deductible in cyber-insurance contracts and do not claim small yet frequent losses, the accumulation of claim data suffers. Lack of claim data may be one reason why after even the past 10 years, cyber insurance is not a major component of corporate IT security initiatives. On the other hand, the relatively small size of the market keeps the costs of writing cyber-insurance contracts high, forcing insurers to impose high margins on individual contracts. Unless it expands, insurers cannot gain more experience or accumulate significant actuarial data and feel no pressing motivation to move to information symmetry. This could mean the market stays locked in information asymmetry.

The structural problem with the market can be resolved if secondary loss were included in contracts. Exotic bundled contracts (individual contracts for primary and secondary losses designed in tandem and bundled together) could be a viable solution. It might take care of the fact that the primary (secondary) losses are determined before (after) the breach, so IT managers are able to take independent decisions concerning disclosures and claims. Even so, valuing secondary loss is more challenging than valuing primary loss, so there appears no easy solution, even if bundled contracts are written.

It is possible that along with increased regulatory compliance and oversight, the relative proportion of private breaches decreases, along with the information asymmetry between insurer and insured. Similarly, separating contracts on the basis of disclosure (compliance or discretionary) might also be a move in a positive direction. However, contracts offered by major insurers today are either ar-

chitecture-oriented (such as a network breach), asset-based (such as a data breach), attack-specific (such as viruses and worms), or liability-focused. No offered contract considers secondary loss or accommodates the complexities of a firm's decision to file a claim in the face of secondary loss. It appears that without significant changes in the design of the contracts, there is little hope for the continued growth of the overall cyber-insurance market. □

References

1. Betterley Report. *Cyberrisk Market Survey 2008*; <http://www.betterley.com>.
2. Bohme, R. Cyber insurance revisited. In *Proceedings of the Workshop on the Economics of Information Security* (Boston, MA, June 2–3, 2005); <http://infosecon.net/workshop/index.php>.
3. Borch, K. *Economics of Insurance. Advanced Textbooks in Economics* 29, K.K. Aase and A. Sandmo, Eds. North Holland, Amsterdam, 1990.
4. Borch, K. *The Mathematical Theory of Insurance*. Lexington Books, Lexington, MA, 1974.
5. Campbell, K., Gordon, L.A., Loeb, M.P., and Zhou L. The economic cost of publicly announced information security breaches: Empirical evidence from the stock market. *Journal of Computer Security* 11, 3 (2003), 431–438.
6. Cavusoglu, H., Mishra, B., and Raghunathan, S. The effect of a security breach announcement on market value: Capital market reactions for breached firms and Internet security developers. *International Journal of Electronic Commerce* 9, 1 (Fall 2004), 69–104.
7. Goltier, C. and Pratt, J.W. Risk vulnerability and the tempering effect of background risk. *Econometrica* 64, 5 (Sept. 1996), 1109–1123.
8. Gordon, L.A., Loeb, M.P., Lucyshyn, W., and Richardson, R. *The 11th Annual CSI/FBI Computer Crime and Security Survey* (2006); http://icmpnet.com/gocsi/db_area/pdfs/fbi/CSI2006.pdf.
9. Gordon, L.A., Loeb, P.M., and Sohail, T. A framework for using insurance for cyber-risk management. *Commun. ACM* 46, 3 (Mar. 2003), 81–85.
10. Kesan, P.J., Majuca, R.P., and Yurcik, W.J. *The Economic Case for Cyber Insurance. Securing Privacy in the Internet Age*. Stanford University Press, Palo Alto, CA, 2005.
11. Majuca, R.P., Yurcik, W., and Kesan, J.P. *The Evolution of Cyber Insurance* (2006); <http://arxiv.org/ftp/cs/papers/0601/0601020.pdf>.
12. Marsh, Inc. E-Commerce E-Business; <http://global.marsh.com/risk/ecommerce/>.
13. Mossin, J. and Smith, T. Aspects of rational insurance purchasing. *Journal of Political Economy* 76, 4 (July/Aug. 1968), 553–568.
14. Ponemon Institute. *The Fourth Annual U.S. Cost of Data Breach Study* (2008); <http://www.ponemon.org>.
15. Raviv, A. The design of an optimal Insurance policy. *American Economic Review* 69, 1 (Mar. 1979), 84–96.
16. Schlesinger, H. The optimal level of deductibility in insurance contracts. *Journal of Risk and Insurance* 48, 3 (Sept. 1981), 465–481.

Tridib Bandyopadhyay (tbandyop@kennesaw.edu) is an assistant professor in the Department of Computer Science and Information Systems at Kennesaw State University, Kennesaw, GA.

Vijay S. Mookerjee (vijaym@utdallas.edu) is the Charles and Nancy Davidson Distinguished Professor of Information Systems and Operations Management in the School of Management at the University of Texas at Dallas.

Ram C. Rao (rroa@utdallas.edu) is the Founders Professor and a professor of marketing in the School of Management at the University of Texas at Dallas.

turing lecture

DOI:10.1145/1592761.1592781

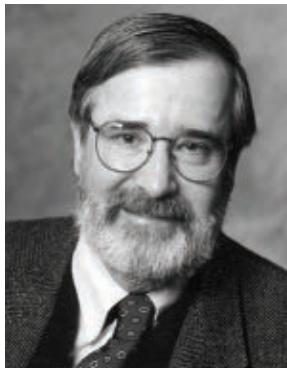
Turing Lecture from the winners of the 2007 ACM A.M. Turing Award.

In 1981, Edmund M. Clarke and E. Allen Emerson, working in the USA, and Joseph Sifakis working independently in France, authored seminal papers that founded what has become the highly successful field of model checking. This verification technology provides an algorithmic means of determining whether an abstract model—representing, for example, a hardware or software design—satisfies a formal specification expressed as a temporal logic (TL) formula. Moreover, if the property does not hold, the method identifies a counterexample execution that shows the source of the problem.

The progression of model checking to the point where it can be successfully used for complex systems has required the development of sophisticated means of coping with what is known as the state explosion problem. Great strides have been made on this problem over the past 28 years by what is now a very large international research community. As a result many major hardware and software companies are beginning to use model checking in practice. Examples of its use include the verification of VLSI circuits, communication protocols, software device drivers, real-time embedded systems, and security algorithms.

The work of Clarke, Emerson, and Sifakis continues to be central to the success of this research area. Their work over the years has led to the creation of new logics for specification, new verification algorithms, and surprising theoretical results. Model checking tools, created by both academic and industrial teams, have resulted in an entirely novel approach to verification and test case generation. This approach, for example, often enables engineers in the electronics industry to design complex systems with considerable assurance regarding the correctness of their initial designs. Model checking promises to have an even greater impact on the hardware and software industries in the future.

—Moshe Y. Vardi, *Editor-in-Chief*



Edmund M. Clarke



E. Allen Emerson



Joseph Sifakis

Model Checking: Algorithmic Verification and Debugging

By Edmund M. Clarke, E. Allen Emerson, and Joseph Sifakis

1. E. ALLEN EMERSON

MODEL CHECKING: A BIRD'S-EYE VIEW

1.1. Formal Verification

Formal verification of program correctness hinges on the use of mathematical logic. A program is a mathematical object with well-defined, although possibly complex and intuitively unfathomable, behavior. Mathematical logic can be used to describe precisely what constitutes correct behavior. This makes it possible to contemplate mathematically establishing that the program behavior conforms to the correctness specification. In most early work this involved constructing a formal proof of correctness. In contradistinction, model checking avoids proofs.

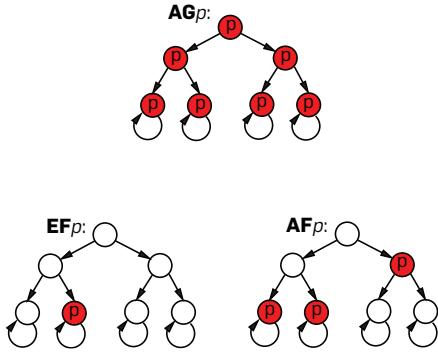
Floyd-Hoare-style deductive verification was the prevailing mode of formal verification going back to the 1960s. This classic and elegant approach entailed manual proof construction, typically using axioms and inference rules in a formal deductive system, often oriented toward sequential programs. Such proof construction was tedious, difficult, and required human ingenuity. This field was a great intellectual success, spawning work on compositional or modular proof systems, soundness of program proof systems, and their completeness; see Section 3. Case studies confirmed that this approach really worked for small programs, although a short program might require a long proof. However, manual verification did not scale up well to large programs. The proofs were just too hard to construct.

1.2. Temporal Logics

In view of the difficulties in trying to construct program proofs it seemed like there ought to be a better way. The way was inspired by the use of *Temporal Logic* (TL), a formalism for describing change over time. If a program can be specified in TL, it can be realized as a finite state system. This suggested the idea of model checking—to check if a finite state graph is a model of a TL specification.

The critical suggestion of using TL for reasoning about ongoing concurrent programs was made in Pnueli's landmark paper.³⁹ Such systems ideally exhibit nonterminating behavior so that they do not conform to the Hoare-style paradigm. They are also typically nondeterministic. Examples include hardware circuits, microprocessors, operating systems, banking networks, communication protocols, automotive electronics, and many modern medical devices. Pnueli used a TL with basic temporal operators **F** (*sometime*) and **G** (*always*). Augmented with **X** (*next-time*) and **U** (*until*), this is today known as LTL (Linear Time Logic).

Another widely used logic is CTL (Computation Tree Logic)¹⁰ (cf. Emerson and Clarke, and Ben-Ari et al.^{20, 4}). Its basic temporal modalities are **A** (for *all* futures) or **E** (for *some* future) followed by one of **F** (*sometime*), **G** (*always*), **X** (*next-time*), and **U** (*until*); compound formulae are built up from nestings and propositional combinations of CTL subformulae. CTL is a branching time logic as it can distinguish between **AFp** (along all futures, *P* eventually holds and

Figure 1. Basic temporal operators.

is thus inevitable) and $\text{EF}p$ (along some future, P eventually holds and is thus possible). The branching time logic CTL* subsumes both CTL and LTL. (See Figure 1.)

Temporal logic formulae are interpreted over a given finite state graph, also called a (*Kripke*) structure, M comprised of a set S of states, a total binary transition relation $R \subseteq S \times S$, and a labelling L of states with atomic facts (propositions such as P) true there. There may also be a distinguished (start) state s_0 . As usual in mathematical logic, to be precise in defining a logic we use the meta-notation $M, s_0 \models f$ as shorthand for “in structure M at state s_0 formula f is true,” for a CTL (or CTL*) formula. When s_0 is understood, we may write $M \models f$. For example, $M, s_0 \models \text{AF}P$ iff for all paths $x = s_0, s_1, s_2, \dots$ in M we have $\exists i \geq 0, P \in L(s_i)$.

When doing specification in practice we may write just $\text{AF}p$ to assert that formula p is inevitable. An LTL formula h is interpreted over a path and then over a structure by implicit universal path quantification: in practical specification we write h but mean Ah .

The LTL formula $\text{G}(\neg(C_1 \wedge C_2))$ captures mutual exclusion for the critical sections, corresponding to assertions C_1 and C_2 , of processes 1 and 2, respectively. In CTL, we would write $\text{AG}(\neg(C_1 \wedge C_2))$ for mutual exclusion, and $\text{AG}(T_1 \Rightarrow \text{AFC}_1)$ for “whenever process 1 enters its trying region (T_1) it inevitably enters its critical section (C_1).” The CTL formula AGEFstart asserts the system can always be restarted; this is not expressible in LTL. The CTL* formula EGFsend asserts the existence of a *fair* behavior along which the *send* condition occurs repeatedly. Such fairness conditions are important in ensuring that goals are fulfilled in concurrent systems.

The logics LTL, CTL, and CTL* have turned out to be very influential, spawning industrial extensions and uses, plus many academic applications as well as theoretical results. There are prominent industrial logics, tailored for hardware verification using special “macros,” i.e., compact high-level operators that expand into longer combinations of basic operators. These include IBM Sugar based on CTL, Intel For-Spec based on LTL, and PSL (IEEE-1850 standard), incorporating features from CTL*.

Finally, there is also the (propositional) mu-calculus³¹ (cf. Emerson and Clarke²⁰), a particular but very general TL. It permits temporal correctness properties to be

characterized as fixed points or *fixpoints* of recursive definitions. For example $\text{EF}p = p \vee \text{EX}(\text{EF}p)$. The mu-calculus plays a vital role in model checking. It is very expressive: CTL, CTL*, as well as LTL, can be encoded in the mu-calculus. The fixed point characterizations of temporal correctness properties underlie many conventional and symbolic model checking algorithms, as well as tools used in practice.

1.3. Model Checking

In the early 1980s Clarke and Emerson proposed *model checking*, a method for automatic (and algorithmic) verification of finite state concurrent systems¹⁰; independently Quielle and Sifakis proposed essentially the same method.⁴¹ In model checking, TL is used to specify correct system behavior. An efficient, flexible search procedure is used to find correct temporal patterns in the finite state graph of the concurrent system. The orientation of the method is to provide a practical verification method. The technical formulation of the model checking problem is simply: Given a finite structure M , state s , and a TL formula f , does $M, s \models f$? An alternative formulation is, given M and f , calculate $\{s : M, s \models f\}$. The main result of Clarke and Emerson¹⁰ is that CTL model checking can be done in time $O(|f| \cdot |M|^2)$; that is, in time polynomial in the formula and the structure sizes. (The result in Queille and Sifakis⁴¹ was with respect to a slightly weaker TL.)

The algorithm was based on fixpoint characterizations of basic temporal modalities. For example, let $f(Z)$ denote $p \vee \text{AX}Z$. We see that $\text{AF}p = f(\text{AF}p)$ is a fixpoint of $f(Z)$, since $\text{AF}p$ holds iff p holds or $\text{AXAF}p$ holds. In general, there may be multiple fixpoints. It can be shown that $\text{AF}p$ is the *least fixpoint*, which we shall write $\mu Z = f(Z)$, with $f(Z)$ as above. Intuitively, least fixpoints capture only well-founded or finite behaviors. The fixpoint characterization $\mu Z = f(Z)$ of a property makes it possible to calculate *iteratively* the set of states where $\text{AF}p$ is true. This utilizes the fact that every formula corresponds to the set of states in which it is true. We compute the maximum of the ascending chain of increasingly larger under-approximations to the desired set of states: $\text{false} \subseteq f(\text{false}) \subseteq f^2(\text{false}) \subseteq \dots \subseteq f^k(\text{false}) = f^{k+1}(\text{false})$, where k is at most the size of the (finite) state space. More generally, the *Tarski-Knaster Theorem* (cf. Tarski⁴⁴) permits the ascending iterative calculation $\cup f^i(\text{false})$ of any temporal property r characterized as a least fixpoint $\mu Z = f(Z)$, provided that $f(Z)$ is *monotone*, which is ensured by Z only appearing un-negated. For greatest fixpoints, one starts the calculation at *true*. Essentially the same algorithm was given in Queille and Sifakis.⁴¹

The following are noteworthy extensions. CTL model checking can be done in time $O(|M| \cdot |f|)$,¹¹ i.e., linear in the size of the state graph and linear in the size of the formula. LTL model checking can be done in time $O(|M| \cdot \exp(|f|))$; since M is usually very large while f is small, the exponential factor may be tolerable.³³ The automata-theoretic approach to LTL model checking is described in Vardi and Wolper.⁴⁶ A succinct fixpoint characterization of fairness from Emerson and Lei²³ is used to make LTL model checking more efficient in practice. Branching time CTL* model

checking can be efficiently reduced to linear time LTL model checking for the same overall bound.²⁴

1.4. Expressiveness

An important criterion for a logic is expressiveness, reflecting what correctness properties can and cannot be captured by the logic. Interesting properties include safety properties (“nothing bad happens,” e.g., $G\text{-}bad$), liveness properties (“something good happens,” e.g., $Egoal$), and fairness properties (“something is recurrent”, e.g., $GFtry$). It is arguable that expressiveness in model checking is the most fundamental characteristic, perhaps even more critical than efficiency. It is imperative that one be able to express all the correctness properties that are needed. If this basic requirement is not met, there is no point in using the verification method in the first place. In actual usage, a particular formalism, commonly a system of TL, provides the needed expressive power. It includes a few basic temporal operators, which can be combined to yield virtually limitless assertions. Another benefit of TL is that it is related to natural language, which can facilitate its use.

The ability to describe complex patterns of system behavior is basic. LTL is naturally suited to the task. Along paths, it is in a sense expressively complete, equivalent to the First Order Language of Linear Order,¹⁹ e.g. $GP = \forall t(t \geq 0 \Rightarrow P(t))$. A property such as G_2P , meaning that P holds at all even moments 0, 2, 4,... is not expressible in LTL. It can be useful in hardware verification applications where it is needed to count clock cycles. The (linear time) mu-calculus as well as PSL can express this property (cf. Wolper⁴⁷).

CTL is well suited to capture correctness over computation trees. The branching time capability of distinguishing between necessary and possible behaviors using explicit path quantifiers (A, E) provides significant expressive power. The existence of a bad path, $EFbad$, is not expressible by any formula Ah where h is in LTL, nor even any universal CTL* formula where all path quantifiers are A (and only atomic propositions appear negated). Thus, LTL is not closed under semantic negation: writing the invariant $G\text{-}bad$ means $AG\text{-}bad$ whose semantic negation is $EFbad$ which, as above, is not expressible by any Ah formula.²¹ There has been an ongoing debate as to whether LTL or branching time logic is better for program reasoning. Linear time offers the advantage of simplicity, but at the cost of significantly less expressiveness. Branching time’s potentially greater expressiveness may incur greater conceptual (and computational) complexity.

A related criterion is succinctness, reflecting how compactly a property can be expressed. The CTL* formula $E(FP_1 \wedge FP_2)$ is not a CTL formula, but is semantically equivalent to the longer CTL formula $EF(P_1 \wedge EFP_2) \vee EF(P_2 \wedge EFP_1)$. For n conjuncts, the translation is exponential in n . In practice, the most important is the criterion of convenience, reflecting how easily and naturally properties can be expressed. Expressiveness and succinctness may be partially amenable to mathematical definition and investigation. Succinctness and convenience often correlate but not always. Convenience, however, is inherently informal.

Yet it is extremely important in actual use. That is why, e.g., many person-years were devoted to formulating industrial-strength logics such as PSL.

1.5. Efficiency

Another important criterion, efficiency, is related to questions of the complexity of the model checking problem for a logic and the performance of model checking algorithms for the logic. An algorithm that has potentially high complexity in theory but is repeatedly observed to exhibit significantly lower complexity in actual use is likely to be preferred to one with better theoretical complexity but inferior observed performance. Moreover, there are trade-offs. For instance, a more expressive logic is likely to be less efficient. A more succinct logic is likely to be more convenient yet even less efficient. Some experience is required to reach a good trade-off. For many model checking applications, M is sufficiently small that it can be explicitly represented in computer memory. Such basic enumerative model checking may be adequate for systems with 10^6 states.

However, many more systems M have an astronomically or even infinitely large state space. There are some fundamental strategies to cope with large state spaces. Foremost, is the use of abstraction where the original, large, complex system M is simplified, by suppressing inessential detail (cf. Clarke and Emerson¹⁰), to get a (representation of a) smaller and simpler system \bar{M} .

Compact representations of the state graph yield another important strategy. The advent of symbolic model checking, combining CTL, fixpoint computation, and data structures for compact representation of large state sets, made it possible to check many systems with an astronomical number of states (cf. Burch et al.⁸).

If there are many replicated or similar subcomponents, it is often possible to factor out the inherent symmetry in the original M resulting in an exponentially reduced abstract \bar{M} ⁴³ (cf. Sistla et al. and Clarke et al.^{43, 9}). Most work on symmetry has required the use of explicit representation of M . Natural attempts to combine symmetry and symbolic representation were shown inherently infeasible.¹⁴ However, a very advantageous combination based on dynamically reorganizing the symbolic representation overcomes these limitations.²⁵ Finally, one may have an infinite state system comprised of, e.g., a (candidate) dining philosophers solution M_n for all sizes $n > 1$. In many situations, this parameterized correctness problem is reducible to model checking a fixed finite-size system M_c (cf. Clarke et al., and Emerson and Kahlon^{9, 22}).

1.6. Evolution of Model Checking

The early reception of model checking was restrained. Model checking originated in the theoretical atmosphere of the early 1980s. There was a field of study known as Logics of Programs, which dealt with the theory and sometime use of logic for reasoning about programs. Various modal and temporal logics played a prominent role. The key technical issue under investigation for such a logic was satisfiability: Given any formula f , determine whether there exists some structure M such that $M \models f$. Analyzing the decidability and

complexity of satisfiability for these logics was the major focus. However, model checking refers to the truth under *one* given interpretation M of a given formula f . This notion was implicit in the Tarskian definition of truth but, classically, was not viewed as an interesting problem. The idea that model checking should provide for verification of finite state systems was not appreciated. The early reaction to model checking then was mostly one of confusion and disinterest. It seemed a disconcerting novelty. It was not satisfiability. It was not validity. What was it? It was even dubbed “disorienting.” Many felt it could not possibly work well in practice. In more recent times, some more favorable comments have been made. Model checking is “an acceptable crutch”—Edsger W. Dijkstra; it is “a first step toward engineerization of the field”—A. Pnueli.⁴⁰

What factors contributed to model checking’s successful deployment? First, the initial framework was feasible and comprehensible. It built on a helpful combination of TL and algorithms. It provided a “push-button,” i.e., automated, method for verification. It permitted bug detection as well as verification of correctness. Since most programs are wrong, this is enormously important in practice. Incidentally, the limited support for bug detection in proof-theoretic verification approaches contributes to their slower adoption rate. Moreover, while a methodology of constructing a program hand-in-hand with its proof certainly has its merits, it is not readily automatable. This hampers its deployment. With model checking, the separation of system development from verification and debugging (see Section 2) has undoubtedly facilitated model checking’s industrial acceptance. The development team can go ahead and produce various aspects of the system under design. The team of verifiers or verification engineers can conduct verification independently. Hopefully, many subtle bugs will be detected and fixed. As a practical matter, the system can go into production at whatever level of “acceptable correctness” prevails at deadline time. Lastly, Moore’s Law has engendered larger computer main memory, which enabled the development of ever more powerful model checking tools.

1.7. Discussion and Summary

What are the key accomplishments of model checking? The key contribution is that verification using model

checking is now done *routinely* on a widespread basis for many large systems, including industrial-strength systems. Large organizations from hardware vendors to government agencies depend on model checking to facilitate achieving their goals. In contrast to 28 years ago, we no longer just talk about verification; we do it. The somewhat surprising conceptual finding is that verification can be done extremely well by automated search rather than manual proofs.

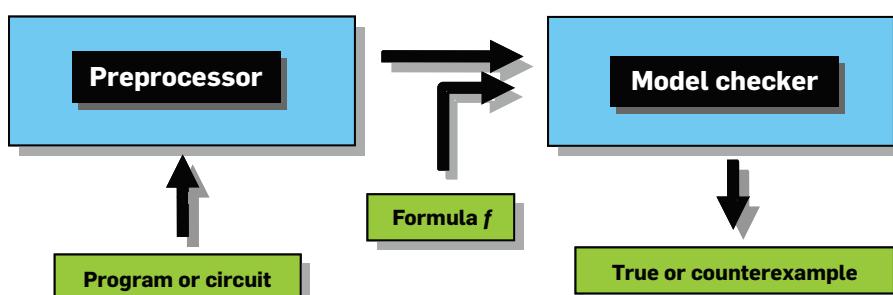
Model checking realizes in small part the *Dream of Leibniz [1646–1716]* (cf. Davis¹⁸). This was a proposal for a universal reasoning system. It was comprised of a *lingua characteristica universalis*, a language in which all knowledge could be formally expressed. TL plays a limited formulation of this role. There was also a *calculus ratiocinator*, a method of calculating the truth value of such a formalized assertion. Model checking algorithms provide the means of calculating truth. We hope that, over time, model checking will realize an increasingly large portion of Leibniz’ Dream.

2. EDMUND M. CLARKE MY 28-YEAR QUEST TO CONQUER THE STATE EXPLOSION PROBLEM

2.1. Model Checkers and Debugging

Model checkers typically have three main components: (1) a *specification language*, based on propositional TL,³⁹ (2) a way of encoding a state machine representing the system to be verified, and (3) a *verification procedure*, that uses an *intelligent* exhaustive search of the state space to determine if the specification is true or not. If the specification is not satisfied, then most model checkers will produce a counterexample execution trace that shows why the specification does not hold. It is impossible to overestimate the importance of this feature. The counterexamples are invaluable in debugging complex systems. Some people use model checking just for this feature. The EMC model checker¹¹ did not give counterexamples for universal CTL properties that were false or witnesses for existential properties that were true. Michael C. Browne added this feature to the MCB model checker in 1984. It has been an important feature of model checkers ever since. (See Figure 2.)

Figure 2. A model checker with counterexamples.



2.2. State Explosion Problem

State explosion is the major problem in model checking. The number of global states of a concurrent system with many processes can be enormous. It is easy to see why this is true. The asynchronous composition of n processes, each having m states, may have m^n states. A similar problem occurs with data. The state-transition system for an n -bit counter will have 2^n states. All model checkers suffer from this problem. Complexity-theoretic arguments can be used to show that the problem is unavoidable in the worst case (assuming P is different than PSPACE). Fortunately, steady progress has been made over the past 28 years for special types of systems that occur frequently in practice. In fact, the state explosion problem has been the driving force behind much of the research in model checking and the development of new model checkers. We discuss below key breakthroughs that have been made and some of the important cases where additional research is needed.

2.3. Major Breakthroughs

2.3.1. Symbolic Model Checking with OBDDs. In the original implementation of the model checking algorithm, transition relations were represented explicitly by adjacency lists.¹¹ For concurrent systems with small numbers of processes, the number of states was usually fairly small, and the approach was often quite practical. In systems with many concurrent parts the number of states in the global state-transition system was too large to handle. In the fall of 1987, McMillan, then a graduate student at Carnegie Mellon, realized that by using a symbolic representation for the state-transition systems, much larger systems could be verified. The new symbolic representation was based on Bryant's *ordered binary decision diagrams* (OBDDs). OBDDs provide a canonical form for Boolean formulas that is often substantially more compact than conjunctive or disjunctive normal form, and very efficient algorithms have been developed for manipulating them. Because the symbolic representation captures some of the regularity in the state space determined by circuits and protocols, it is possible to verify systems with an extremely large number of states—many orders of magnitude larger than could be handled by the explicit-state algorithms. With the new representation for state-transition systems, we could verify some examples that had more than 10^{20} states.^{8,35} Since then, various refinements of the OBDD-based techniques have pushed the state count up to more than 10^{120} .

2.3.2. Partial Order Reduction. Verifying software poses significant problems for model checking. Software tends to be less structured than hardware. In addition, concurrent software is usually *asynchronous*, i.e., most of the activities taken by different processes are performed independently, without a global synchronizing clock. For these reasons, the state explosion problem is particularly serious for software. Consequently, model checking has been used less frequently for software verification than for hardware verification. One of the most successful techniques for dealing with asynchronous systems is the *partial order reduction*. This technique exploits the independence of concurrently

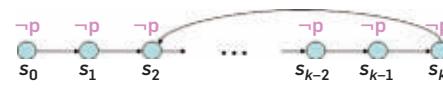
executed events. Intuitively, two events are *independent* of each other when executing them in either order results in the same global state. In this case, it is possible to avoid exploring certain paths in the state-transition system. Model checking algorithms that incorporate the partial order reduction are described in several different papers. The *stubborn sets* of Valmari,⁴⁵ the *persistent sets* of Godefroid²⁷ and the *ample sets* of Peled,³⁸ differ on the actual details, but contain many similar ideas. The SPIN model checker developed by Holzmann uses the ample-set reduction to great advantage.

2.3.3. Bounded Model Checking with SAT. Although symbolic model checking with OBDDs was the first big breakthrough on the state explosion problem and is still widely used, OBDDs have a number of problems that limit the size of the models that can be checked with this technique. The ordering of variables on each path from the root of the OBDD to a leaf has to be the same. Finding an ordering that results in a small OBDD is quite difficult. In fact, for some Boolean formulas no space-efficient ordering is possible. A simple example is the formula for the middle output bit of a combinational multiplier for two n -bit numbers. It is possible to prove that the OBDD for this formula has size that is exponential in n for all variable orderings.

Propositional satisfiability (SAT) is the problem of determining whether a propositional formula in conjunctive normal form (“product of sums form” for Boolean formulas) has a truth assignment that makes the formula true. The problem is NP-complete (in fact, it is usually the first example of this class that students see). Nevertheless, the increase in power of modern SAT solvers over the past 15 years on problems that occur in practice has been phenomenal. It has become the key enabling technology in applications of model checking to both computer hardware and software. Bounded Model Checking (BMC) of computer hardware using a fast SAT solver is now probably the most widely used model checking technique. The counterexamples that it finds are just the satisfying instances of the propositional formula obtained by unwinding to some fixed depth the state-transition system for the circuit and the negation of its specification in linear TL.

The basic idea for BMC is quite simple (cf. Biere et al.⁷). The extension to full LTL obscures the simplicity so we will just describe how to check properties of the form FP where the property P is an atomic proposition (e.g., “Message_Received”). BMC determines whether there is a counterexample of length k (we assume $k \geq 1$). In other words, it checks if there is a path of length k ending in a cycle in which each state is labeled with $\neg P$ (see Figure 3). Assume that the state-transition system M has n states. Each state can be encoded by a vector \vec{v} of $\lceil \log(n) \rceil$ Boolean variables.

Figure 3. Counterexample of length at most k .



The set of initial states can be specified by a propositional formula $I(\vec{v})$ which holds for exactly those assignments to \vec{v} that correspond to initial states. Likewise, the transition relation can be given by a propositional formula $R(\vec{v}, \vec{v}')$. A path of length k starting in an initial state can be encoded by means of the following formula:

$$\text{path}(k) = I(\vec{v}_0) \wedge R(\vec{v}_0, \vec{v}_1) \wedge \cdots \wedge R(\vec{v}_{k-1}, \vec{v}_k). \quad (1)$$

The path ends in a cycle if and only if

$$\text{cycle}(k) = R(\vec{v}_k, \vec{v}_0) \vee \cdots \vee R(\vec{v}_k, \vec{v}_{k-1}) \vee R(\vec{v}_k, \vec{v}_k). \quad (2)$$

The property P is false in each of the k steps if and only if

$$\text{property}(k) = \neg P(\vec{v}_0) \wedge \neg P(\vec{v}_1) \wedge \cdots \wedge \neg P(\vec{v}_k). \quad (3)$$

Thus, the liveness property FP has a counterexample of length k if and only if the conjunction $\Omega(k)$ of Formulas 1, 2, and 3 is satisfiable.

$$\Omega(k) = \text{path}(k) \wedge \text{cycle}(k) \wedge \text{property}(k). \quad (4)$$

We start with $k = 1$. If the formula $\Omega(k)$ is satisfiable, we know that FP has a counterexample of length k . A counterexample execution trace can be extracted from the satisfying assignment to $\Omega(k)$. If the formula $\Omega(k)$ is not satisfiable, then it could be the case that either the temporal formula FP holds on all paths starting from an initial state (and our specification is true) or there is a counterexample that is longer than k . When $\Omega(k)$ is unsatisfiable, we can do one of two things: Either increase the value of k and look for longer counterexamples or stop if time or memory constraints are exceeded.

We note that the idea of checking safety properties such as GP by reduction to propositional satisfiability is implicit in the work of Kautz and Selman.³⁰ However, they do not consider more general temporal properties such as the liveness property that we consider above.

In practice, BMC can often find counterexamples in circuits with thousands of latches and inputs. Armin Biere recently reported an example in which the circuit had 9510 latches and 9499 inputs. This resulted in a propositional formula with 4×10^6 variables and 1.2×10^7 clauses. The shortest bug of length 37 was found in 69 seconds! Many others have reported similar results.

Can BMC ever be used to prove correctness if no counterexamples are found? It is easy to see that for safety and liveness properties of the form FP and GP where P is a propositional formula, if there is a counterexample, then there is one that is less than the diameter (i.e., the longest shortest path between any two states) of the state-transition system. So, the diameter could be used to place an upper bound on how much the transition relation would need to be unwound. Unfortunately, it appears to be computationally difficult to compute the diameter when the state-transition system is given implicitly as a circuit or in terms of propositional formulas for the set of initial states, the transition relation, and the set of bad states. Other ways for making BMC complete are based on cube

enlargement,³⁶ circuit co-factoring,²⁶ induction,⁴² and Craig interpolants.³⁷ But, the problem remains a topic of active research. Meanwhile, an efficient way of finding subtle counterexamples is still quite useful in debugging circuit designs.

2.3.4. The Abstraction Refinement Loop. This technique uses counterexamples to refine an initial abstraction. We begin by defining what it means for one state-transition system to be an abstraction of another. We write $M_\alpha = \langle S_\alpha, s_0^\alpha, R_\alpha, L_\alpha \rangle$ to denote the *abstraction* of state-transition system $M = \langle S, s_0, R, L \rangle$ with respect to an *abstraction mapping* α . (Here we include the start states s_0 and s_0^α as parts of the state-transition systems.) We assume that the states of M are labeled with atomic propositions from a set A of atomic propositions, and that M_α is labeled with atomic propositions from a set A_α that is a subset of A . We call M the *concrete system* and M_α the *abstract system*.

DEFINITION 1. A function $\alpha: S \rightarrow S_\alpha$ is an abstraction mapping from the concrete system M to the abstract system M_α with respect to the propositions in A_α if and only if

- $\alpha(s_0) = s_0^\alpha$.
- If there is a transition from state s to state t in M , then there is a transition from $\alpha(s)$ to $\alpha(t)$ in M_α .
- For all states s , $L(s) \cap A_\alpha = L_\alpha(\alpha(s))$.

The three conditions ensure that M_α simulates M . Note that only identically labeled states of the concrete model (modulo propositions absent from A_α) will be mapped into the same state of the abstract model (see Figure 4). The key theorem relating concrete and abstract systems is the *Property Preservation Theorem*:

THEOREM 1 (CLARKE, GRUMBERG, and LONG¹³). If a universal CTL* property holds on the abstract model, then it holds on the concrete model.

Here, a universal CTL* property is one that contains no existential path quantifiers when written in negation-normal form. For example, AFP is a universal property but EFP is not.

Figure 4. A concrete system and its abstraction.

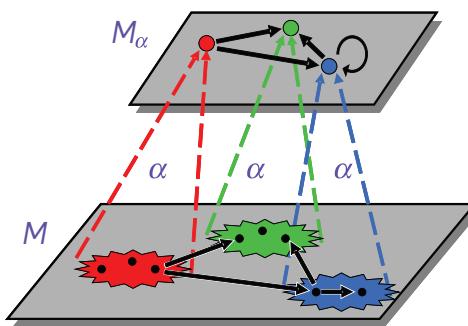
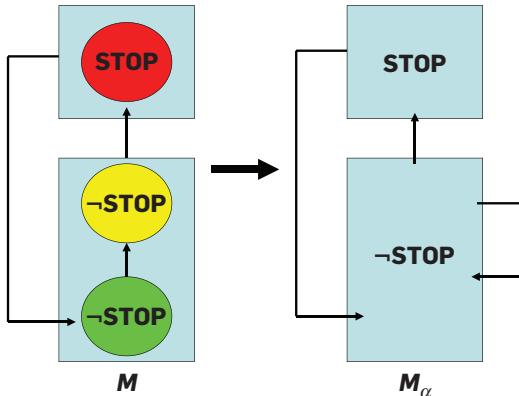


Figure 5. Spurious counterexample.

The converse of the theorem is not true as Figure 5 illustrates. A universal property that holds in the concrete system may fail to hold in the abstract system. For example, the property **AGF STOP (infinitely often STOP)** holds in M , but not in M_α . Thus, a counterexample to the property in the abstract system may fail to be a counterexample in the concrete system. Such counterexamples are said to be *spurious* counterexamples. This leads to a verification technique called *Counterexample Guided Abstraction Refinement* (CEGAR).¹² Universal properties are checked on a series of increasingly precise abstractions of the original system. If the property holds, then by the Property Preservation Theorem, it must hold on the concrete system and we can stop. If it does not hold and we get a counterexample, then we must check the counterexample on the concrete system in order to make sure that it is not spurious. If the counterexample checks on the concrete system, then we have found an error and can also stop. If the counterexample is

spurious, then we use information in the counterexample to refine the abstraction mapping and repeat the loop. The CEGAR Loop in Figure 6 generalizes an earlier abstraction technique for sequential circuits called the *localization reduction*, which was developed by R. Kurshan.³² CEGAR is used in many software model checkers including the SLAM Project at Microsoft.¹

2.4. State Explosion Challenges for the Future

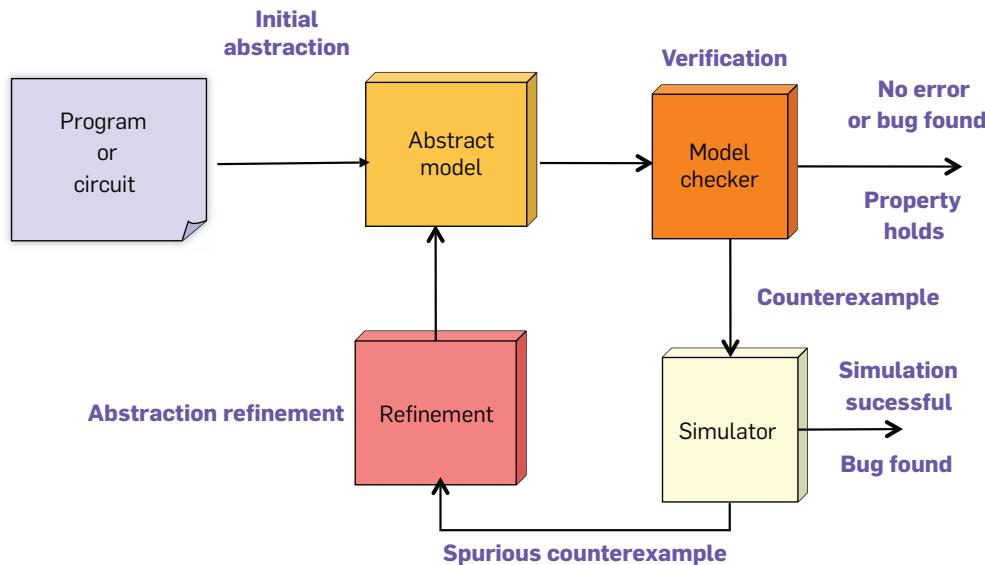
The state explosion problem is likely to remain the major challenge in model checking. There are many directions for future research on this problem, some of which are listed below.

- Software model checking, in particular, combining model checking and static analysis
- Effective model checking algorithms for real-time and hybrid systems
- Compositional model checking of complex systems
- Symmetry reduction and parameterized model checking
- Probabilistic and statistical model checking
- Combining model checking and theorem proving
- Interpreting long counterexamples
- Scaling up even more

3. JOSEPH SIFAKIS THE QUEST FOR CORRECTNESS: CHALLENGES AND PERSPECTIVES

3.1. Where Are We Today?

Verification techniques have definitely found important applications. After the first two decades of intensive research and development, recent years have been characterized by a shift in focus and intensity.

Figure 6. The CEGAR loop.

Algorithmic verification involves three different tasks: (1) requirements specification, (2) building executable system models, and (3) developing scalable algorithms both for checking requirements and for providing diagnostics when requirements are not met. The status for each of these tasks is discussed below.

3.1.1. Requirements Specification. Requirements characterize the expected behavior of a system. They can be expressed following two paradigms. *State-based* requirements specify a system's observable behavior by using transition systems. *Property-based* requirements use a declarative style. These requirements are expressed as sets of formulas in a formalism such as a TL. A combination of the two paradigms is necessary for enhanced expressiveness, such as in the PSL language. The state-based paradigm is adequate for characterizing causal dependencies between events, e.g., sequences of actions. In contrast, the property-based paradigm is more appropriate for global properties, e.g., liveness and mutual exclusion. For concurrent systems, an important trend is toward semantic variations of state-based formalisms such as Live Sequence Charts.¹⁷

Using TLs has certainly been a breakthrough in understanding and formalizing requirements for concurrent systems. Nonetheless, subtle differences in the formulation of common concepts such as liveness and fairness, which depend on the underlying time model (e.g., branching or linear time), show that writing rigorous logic specifications is not trivial.

Furthermore, the declarative and dense style in the expression of property-based requirements is not always easy to master and understand. Requirements must be *sound*. That is, they must be satisfiable by some model. In addition, they must be *complete*. That is, no important information is omitted about the specified system. In contrast to soundness, which is a well-understood property and can be checked automatically by using decision procedures, there is no consensus as to what precisely constitutes completeness in requirements specifications, nor how to go about achieving it. Absolute completeness, which means that specifications describe the system exactly, has only a theoretical interest and is probably unattainable for non-trivial systems.

Existing requirements specification formalisms are mainly appropriate for expressing functional requirements. We lack rigorous formalisms for extra-functional requirements for security properties (e.g., privacy), reconfigurability properties (e.g., noninterference of configurable features), and quality of service (e.g., degree of jitter).

3.1.2. Building Executable Models. Successful application of verification methods requires techniques for building executable models that *faithfully* represent a system or an abstraction of it. Faithfulness means that the system to be verified and its model are related through a checkable semantics-preserving relation. This will ensure soundness of the model. In other words, any property that we can verify for the model will hold for the real system. Furthermore, to avoid errors in building models and to cope with their complexity, models should be generated automatically from system descriptions.

For hardware verification, it is relatively straightforward to generate exact logical finite-state models, expressed as systems of boolean equations, e.g., from RTL descriptions. This probably explains the strong and immediate success of model checking in the area. For software, the problem is more difficult. In contrast to logical hardware models, we need to define formally the semantics of the programming language. This may not be an easy task for languages such as C or Java, as it requires some clarification of concepts and additional assumptions about their semantics. Once the semantics is fixed, tractable models can be extracted from real software through abstraction. This allows us to cope with complexity of data and dynamic features. Currently, we do not know how to build faithful models for systems consisting of hardware and software, at the same level of detail as for pure hardware or software. Ideally, for a system consisting of application software running on a platform, the corresponding model could be obtained as the composition of models for the software and the platform. The main difficulty is in understanding and formalizing the interaction between these two types of models, in particular by taking into account timing aspects and resources such as memory and energy. In addition, this should be done at some adequate level of abstraction, allowing tractable models.

Today, we can specify and verify only high-level timed models with tools such as Uppaal³ for schedulability analysis. These models take into account hardware timing aspects and some abstraction of the application software. The validation of even relatively simple systems such as a node in a wireless sensor network is carried out by testing physical prototypes or by ad-hoc simulation. We need theory, methods, and tools for modeling complex heterogeneous systems.² Weaknesses in the state of the art are also seen in standards and languages for system modeling. Efforts for extending UML to cover scheduling and resource management issues have failed to provide a rigorous basis for this. At the same time, extensions of hardware description languages to encompass more asynchronous execution models such as SystemC and TLM can be used only for simulation, due to a lack of formal semantic foundations.

3.1.3. Scalable Verification Methods. Today we have fairly efficient verification algorithms. However, all suffer from well-known inherent complexity limitations when applied to large systems. To cope with this complexity, I see two main avenues.

The first avenue is to develop new abstraction techniques, in particular for specific semantic domains depending on the data handled by the system and on the properties to be verified. The convergence between model checking and abstract interpretation¹⁶ could lead to significant breakthroughs. These two main algorithmic approaches, which have developed rather independently for almost three decades, have a common foundation: solving fixpoint equations in specific semantic domains.

Initially, model checking focused on the verification of finite state systems such as hardware or complex

control-intensive reactive systems such as communication protocols. Later, research on model checking addressed verification of infinite state systems by using abstractions.^{13, 34} The evolution of abstract interpretation is driven by the concern for finding adequate abstract domains for efficient verification of program properties by computing approximations of reachability sets. Model checking has had a broader application scope, including hardware, software, and systems. Furthermore, depending on the type of properties to be checked, model checking algorithms may involve computation of multiple fixed points. I believe that the combination of the two algorithmic approaches can still lead to significant progress in the state of the art, e.g., by using libraries of abstract domains in model checking algorithms.

The second avenue addresses significant long-term progress in defeating complexity. It involves moving from monolithic verification to compositional techniques. We need divide-and-conquer approaches for inferring global properties of a system from the properties of its components. The current state of the art does not meet our initial expectations. The main approach is by “assume-guarantee,” where properties are decomposed into two parts. One is an assumption about the global behavior of the system within which the component resides; the other is a property guaranteed by the component when the assumption about its environment holds. As discussed in a recent paper,¹⁵ many issues make it difficult to apply assume-guarantee rules, in particular because synthesis of assumptions (when feasible) may cost as much as monolithic verification.

In my opinion, any *general* compositional verification theory will be highly intractable and will be of theoretical interest only. We need to study compositionality results for particular classes of properties and/or particular classes of systems as explained below.

3.2. From a posteriori Verification to Constructivity

A big difference between Computer Engineering and more mature disciplines based on Physics, e.g., Electrical Engineering, is the importance of verification for achieving correctness. These disciplines have developed theory guaranteeing by construction the correctness and predictability of artifacts. For instance, the application of Kirchoff’s laws allows building circuits that meet given properties.

My vision is to investigate links between compositional verification for specific properties and results allowing constructivity. Currently, there exists in Computer Science an important body of constructivity results about architectures and distributed algorithms.

1. We need theory and methods for building faithful models of complex systems as the composition of heterogeneous components, e.g., mixed software/hardware systems. This is a central problem for ensuring correct interoperation, and meaningful refinement and integration of heterogeneous viewpoints. Heterogeneity has three fundamental sources which appear when composing components with different (a) execution models, e.g., synchronous and asynchronous execu-

tion, (b) interaction mechanisms such as locks, monitors, function calls, and message passing, and (c) granularity of execution, e.g., hardware and software.²⁹

We need to move from composition frameworks based on the use of a single low-level parallel composition operator, e.g., automata-based composition, to a unified composition paradigm encompassing architectural features such as protocols, schedulers, and buses.

2. In contrast to existing approaches, we should investigate compositionality techniques for high-level composition operators and specific classes of properties. I propose to investigate two independent directions:

- One direction is studying techniques for specific classes of properties. For instance, finding compositional verification rules guaranteeing deadlock-freedom or mutual exclusion instead of investigating rules for safety properties in general. Potential deadlocks can be found by analysis of dependencies induced by interactions between components.²⁸ For proving mutual exclusion, a different type of analysis is needed.
- The other direction is studying techniques for particular architectures. Architectures characterize the way interaction among a system’s components is organized. For instance, we might profitably study compositional verification rules for ring or star architectures, for real-time systems with preemptable tasks and fixed priorities, for time-triggered architectures, etc. Compositional verification rules should be applied to high-level coordination mechanisms used at the architecture level, without translating them into a low-level automata-based composition.

The results thus obtained should allow us to identify “verifiability” conditions (i.e., conditions under which verification of a particular property and/or class of systems becomes scalable). This is similar to finding conditions for making systems testable, adaptable, etc. In this manner, compositionality rules can be turned into correct-by-construction techniques.

Recent results implemented in the D-Finder tool^{15, 6} provide some illustration of these ideas. D-Finder uses heuristics for proving compositionally global deadlock-freedom of a component-based system, from the deadlock-freedom of its components. The method is compositional and proceeds in two steps.

- First, it checks that individual components are deadlock-free. That is, they may block only at states where they are waiting for synchronization with other components.
- Second, it checks if the components’ interaction graph is acyclic. This is a sufficient condition for establishing global deadlock-freedom at low cost. It depends only on the system architecture. Otherwise, D-Finder symbolically

computes increasingly strong global deadlock-free invariants of the system, based on results from the first step. Deadlock-freedom is established if there exists some invariant that is satisfied by the system's initial state.

Benchmarks published in Bensalem et al.⁶ show that such a specialization for deadlock-freedom, combined with compositionality techniques, leads to significantly better performance than is possible with general-purpose monolithic verification tools.

A posteriori verification is not the only way to guarantee correctness. System designers develop complex systems, by carefully applying architectural principles that are operationally relevant and technically successful. Verification should advantageously take into account architectures and their features. There is a large space to be explored, between full constructivity and a posteriori verification. This vision can contribute to bridging the gap between Formal Methods and the body of constructivity results in Computer Science. □

References

- Ball, T., Rajamani, S.K. The SLAM toolkit. In *Computer-Aided Verification (CAV'01)*. Volume 2102 of *Lecture Notes in Computer Science* (2001), 260–264.
- Basu, A., Bozga, M., Sifakis, J. Modeling heterogeneous real-time components in BIP. In *SEFM* (2006), 3–12.
- Behrmann, G., Coguñard, A., David, A., Fleury, E., Larsen, K.G., Lime, D. Uppaal-tiga: Time for playing games! In *CAV*. W. Damm and H. Hermanns, eds. Volume 4590 of *Lecture Notes in Computer Science* (Springer, 2007), 121–125.
- Ben-Ari, M., Pnueli, A., Manna, Z. The temporal logic of branching time. *Acta Inf.* 20 (1983), 207–226.
- Bensalem, S., Bozga, M., Nguyen, T.-H., Sifakis, J. D-finder: A tool for compositional deadlock detection and verification. In *CAV*. A. Bouajjani and O. Maler, eds. Volume 5643 of *Lecture Notes in Computer Science* (Springer, 2009), 614–619.
- Bensalem, S., Bozga, M., Sifakis, J., Nguyen, T.-H. Compositional verification for component-based systems and application. In *ATVA*. S.-D. Cha, J.-Y. Choi, M. Kim, I. Lee, and M. Viswanathan, eds. Volume 5311 of *Lecture Notes in Computer Science* (Springer, 2008), 64–79.
- Biere, A., Cimatti, A., Clarke, E.M., Zhu, Y. Symbolic model checking without BDDs. In *Proceedings of the 5th International Conference on Tools and Algorithms for Construction and Analysis of Systems (TACAS'99)*. R. Cleaveland, ed. Volume 1579 of *Lecture Notes in Computer Science* (Springer-Verlag, Mar. 1999), 193–207.
- Burch, J.R., Clarke, E.M., McMillan, K.L., Dill, D.L., Hwang, L.J. Symbolic Model Checking: 10^{20} states and beyond. *Inf. Comput.* 98, 2 (June 1992), 142–170. Originally presented at the 1990 Symposium on Logic in Computer Science (LICS'90).
- Clarke, E., Grumberg, O., Peled, D. *Model Checking*. MIT Press, 1999.
- Clarke, E.M., Emerson, E.A. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Logics of Programs: Workshop, Yorktown Heights, NY, May 1981*. Volume 131
- of *Lecture Notes in Computer Science* (Springer, 1981), 52–71.
- Clarke, E.M., Emerson, E.A., Sistla, A.P. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. Prog. Lang. Syst.* 8, 2 (1986), 244–263. Originally presented at the 1983 Symposium on Principles of Programming Languages (POPL'83).
- Clarke, E.M., Grumberg, O., Jha, S., Lu, Y., Veith, H. Counterexample-guided abstraction refinement for symbolic Model Checking. *J. ACM* 50, 5 (2003), 752–794. Originally presented at the 2000 Conference on Computer-Aided Verification (CAV'00).
- Clarke, E.M., Grumberg, O., Long, D.E. Model checking and abstraction. *ACM Trans. Program. Lang. Syst.* 16, 5 (1994), 1512–1542.
- Clarke, E.M., Jha, S., Enders, R., Filkorn, T. Exploiting symmetry in temporal logic model checking. *Formal Methods Sys Design* 9, 1/2 (1996), 77–104.
- Cobleigh, J.M., Avrunin, G.S., Clarke, L.A. Breaking up is hard to do: An evaluation of automated assume-guarantee reasoning. *ACM Trans. Softw. Eng. Methodol.* 17, 2 (2008), 1–52.
- Cousot, P., Cousot, R. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL* (1977), 238–252.
- Damm, W., Harel, D. LSCs: Breathing life into message sequence charts. *Formal Methods Sys. Design* 19, 1 (2001), 45–80.
- Davis, M. *The Universal Computer: The Road from Leibniz to Turing*. W.W. Norton & Co., 2000.
- Emerson, E.A. Temporal and modal logic. In *Handbook of Theoretical Computer Science*. J. van Leeuwen, ed. Volume B, chapter 16, Elsevier Science (1990), 995–1072.
- Emerson, E.A., Clarke, E.M. Characterizing correctness properties of parallel programs using fixpoints. In *Lecture Notes in Computer Science* 85. Automata, Languages and Programming (July 1980), 169–181.
- Emerson, E.A., Halpern, J.Y. "Sometimes" and "Not Never" revisited: On branching time versus linear time. *J. ACM* 33 (1986), 151–178.
- Emerson, E.A., Kahlon, V. Reducing model checking of the many to the few. In *CADE*. D.A. McAllester, ed. Volume 1831 of *Lecture Notes in Computer Science* (Springer, 2000), 236–254.
- Emerson, E.A., Lei, C.-L. Efficient model checking in fragments of the propositional mu-calculus (extended abstract). In *Proceedings, Symposium on Logic in Computer Science, 16–18 June 1986*, Cambridge, MA, USA, 1986, 267–278.
- Emerson, E.A., Lei, C.-L. Modalities for model checking: Branching time logic strikes back. *Sci. Comput. Progr.* 8, 3 (1987), 275–306.
- Emerson, E.A., Wahl, T. Dynamic symmetry reduction. In *TACAS*. N. Halbwachs and L.D. Zuck, eds. Volume 3440 of *Lecture Notes in Computer Science* (Springer, 2005), 382–396.
- Ganai, M.K., Gupta, A., Ashar, P. Efficient SAT-based unbounded symbolic model checking using circuit cofactoring. In *International Conference on Computer-Aided Design (ICCAD'04)* (2004), 510–517.
- Godefroid, P. Using partial orders to improve automatic verification methods. In *Computer-Aided Verification (CAV'90)*. Volume 531 of *Lecture Notes in Computer Science* (1990), 176–185.
- Gößler, G., Sifakis, J. Composition for component-based modeling. *Sci. Comput. Progr.* 55, 1–3 (2005), 161–183.
- Henzinger, T.A., Sifakis, J. The discipline of embedded systems design. *IEEE Comp.* 40, 10 (2007), 32–40.
- Kautz, H.A., Selman, B. Planning as satisfiability. In *10th European Conference on Artificial Intelligence (ECAI'92)*, 359–363.
- Kozen, D. Results on the propositional mu-calculus. *Theor. Comput. Sci.* 27 (Dec. 1983), 333–354.
- Kurshan, R.P. *Computer-Aided Verification of Coordinating Processes*. Princeton University Press, 1994.
- Lichtenstein, O., Pnueli, A. Checking that finite state concurrent programs satisfy their linear specification. In *POPL* (1985), 97–107.
- Loiseaux, C., Graf, S., Sifakis, J., Bouajjani, A., Bensalem, S. Property preserving abstractions for the verification of concurrent systems.
- Sistla, A.P., Gyuris, V., Emerson, E.A. SMC: a symmetry-based model checker for verification of safety and liveness properties. *ACM Trans. Softw. Eng. Methodol.* 9, 2 (2000), 133–166.
- Tarski, A. A lattice-theoretical fixpoint theorem and its applications. *Pacific J. Math.* 5 (1955), 285–309.
- Valmari, A. A stubborn attack on the state explosion problem. In *Computer-Aided Verification (CAV'90)*. Volume 531 of *Lecture Notes in Computer Science* (1990).
- Vardi, M.Y., Wolper, P. An automata-theoretic approach to automatic program verification (preliminary report). In *Proceedings, Symposium on Logic in Computer Science, 16–18 June 1986*, Cambridge, MA, USA, 1986, 332–344.
- Wolper, P. Temporal logic can be more expressive. *Inform. Control* 56 (1983), 72–99.

Edmund M. Clarke

FORE Systems University Professor
Computer Science Department
Carnegie Mellon University, Pittsburgh, PA.

E. Allen Emerson

Regents Chair and Professor
Department of Computer Science
University of Texas, Austin, TX.

Joseph Sifakis

Schneider-INRIA Chair
Verimag Laboratory Gieres, France.

Formal Methods Sys Design 6, 1 (1995), 11–44.

35. McMillan, K.L. *Symbolic Model Checking: An Approach to the State Explosion Problem*. Kluwer Academic Publishers, 1993.

36. McMillan, K.L. Applying SAT methods in unbounded symbolic model checking. In *Computer-Aided Verification (CAV'02)*. Volume 2404 of *Lecture Notes in Computer Science* (2002), 250–264.

37. McMillan, K.L. Interpolation and SAT-based model checking. In *Computer-Aided Verification (CAV'03)*. Volume 2725 of *Lecture Notes in Computer Science* (2003), 1–13.

38. Peled, D. Combining partial order reductions with on-the-fly Model-Checking. In *Computer Aided Verification (CAV'94)*. Volume 818 of *Lecture Notes in Computer Science* (1994), 377–390.

39. Pnueli, A. The temporal logic of programs. Presented at FOCS, Oct. 1977.

40. Pnueli, A. Verification engineering: A future profession (A. M. Turing Award Lecture). Presented at PODC (Aug. 1997).

41. Quelle, J.P., Sifakis, J. Specification and verification of concurrent systems in CESAR. In *Proceedings of the 5th International Symposium on Programming* (1982), 337–350.

42. Sheeran, M., Singh, S., Stålmarck, G. Checking safety properties using induction and a SAT-solver. In *Formal Methods in Computer-Aided Design (FMCAD'02)*. Volume 1954 of *Lecture Notes in Computer Science* (2000), 108–125.

43. Sistla, A.P., Gyuris, V., Emerson, E.A. SMC: a symmetry-based model checker for verification of safety and liveness properties. *ACM Trans. Softw. Eng. Methodol.* 9, 2 (2000), 133–166.

44. Tarski, A. A lattice-theoretical fixpoint theorem and its applications. *Pacific J. Math.* 5 (1955), 285–309.

45. Valmari, A. A stubborn attack on the state explosion problem. In *Computer-Aided Verification (CAV'90)*. Volume 531 of *Lecture Notes in Computer Science* (1990).

46. Vardi, M.Y., Wolper, P. An automata-theoretic approach to automatic program verification (preliminary report). In *Proceedings, Symposium on Logic in Computer Science, 16–18 June 1986*, Cambridge, MA, USA, 1986, 332–344.

47. Wolper, P. Temporal logic can be more expressive. *Inform. Control* 56 (1983), 72–99.

research highlights

P. 86

Technical Perspective Narrowing the Semantic Gap In Distributed Programming

By Peter Druschel

P. 87

Declarative Networking

By Boon Thau Loo, Tyson Condie, Minos Garofalakis, David E. Gay, Joseph M. Hellerstein, Petros Maniatis, Raghu Ramakrishnan, Timothy Roscoe, and Ion Stoica

P. 96

Technical Perspective Machine Learning for Complex Predictions

By John Shawe-Taylor

P. 97

Predicting Structured Objects with Support Vector Machines

By Thorsten Joachims, Thomas Hofmann, Yisong Yue, and Chun-Nam Yu

Technical Perspective Narrowing the Semantic Gap In Distributed Programming

By Peter Druschel

IN SCIENCE, SIGNIFICANT advances are often made when researchers from different communities join forces. The following paper by Hellerstein et al. is a good example. Declarative networking builds on ideas from the database, programming languages, networking, and distributed systems communities to create a novel programming paradigm for distributed systems. With declarative networking, designers express their protocols concisely using a high-level, declarative language. The program is then automatically compiled to executable code, which can be deployed in a real network.

To see why this is important, consider how distributed systems are currently implemented. Developers program the behavior of individual nodes at a low level. They use a language like C++ or Java to program a state machine that relates each possible local event (for example, message arrival, timeout) and the current state of the node to a set of local actions (for example, message transmission) and an associated change in the local state. Ultimately, designers are interested in the high-level properties of a system, such as its availability, the consistency of the results it produces, and its ability to tolerate faults. However, these properties cannot be explicitly specified nor directly observed, because they emerge from the complex interaction of the low-level, local behavior of individual nodes with each other and with the network.

When implementing a system, the programmer must specify a low-level action for each local event, while attempting to achieve the high-level, global properties. During debugging, the designer would like to assert global

properties, but can observe directly only the low-level behavior and local state of individual nodes. This semantic gap between the global, high-level properties we want to achieve and the local, low-level behavior we must specify contributes to the difficulty of implementing, debugging, and monitoring distributed systems.

Declarative networking aims to narrow this gap, thereby simplifying the task of developing correct distributed software. Developers use an extended database query language to program in a declarative fashion, which has a number of benefits.

Declarative programs are concise, can specify non-local properties directly, and abstract away many low-level details. As a result, the programs look very similar to the pseudo-code

Thanks to the authors' efforts, know-how from the database, declarative programming and verification communities can be brought to bear on the problem of developing correct distributed systems.

often used to describe a system's design. Runtime queries can be added to monitor the global state of the system at the same level of abstraction as the program, which aids in the debugging and monitoring of a system. Once a distributed system is specified in this language, query optimizations and implementation techniques from the database world can be used to increase the efficiency and adaptivity of the resulting system. Lastly, declarative query languages are amenable to automated program analysis and reasoning, which can help to bring formal verification techniques to the design of practical distributed systems.

Declarative networking is a young research area and the P2 system described by the authors is not likely to be the final word on the subject. Questions remain, for instance, about the appropriate semantics of the query language, the class of distributed programs that can be expressed naturally in the declarative style, and the efficiency of distributed programs compiled from a declarative specification. However, the authors deserve much credit for breaking new ground and challenging conventional wisdom. Thanks to their efforts, know-how from the database, declarative programming and verification communities can be brought to bear on the problem of developing correct distributed systems. Moreover, developments in these areas are bound to bring further advances to declarative networking. □

Peter Druschel (druschel@mpi-sws.org) is the founding director of the Max Planck Institute for Software Systems, Kaiserslautern and Saarbrücken, Germany, where he leads the distributed systems research group.

© 2009 ACM 0001-0782/09/1100 \$10.00

Declarative Networking

By Boon Thau Loo, Tyson Condie, Minos Garofalakis, David E. Gay, Joseph M. Hellerstein, Petros Maniatis, Raghu Ramakrishnan, Timothy Roscoe, and Ion Stoica

Abstract

Declarative Networking is a programming methodology that enables developers to concisely specify network protocols and services, which are directly compiled to a dataflow framework that executes the specifications. This paper provides an introduction to basic issues in declarative networking, including language design, optimization, and dataflow execution. We present the intuition behind declarative programming of networks, including roots in Datalog, extensions for networked environments, and the semantics of long-running queries over network state. We focus on a sublanguage we call *Network Datalog (NDlog)*, including execution strategies that provide crisp eventual consistency semantics with significant flexibility in execution. We also describe a more general language called *Overlog*, which makes some compromises between expressive richness and semantic guarantees. We provide an overview of declarative network protocols, with a focus on routing protocols and overlay networks. Finally, we highlight related work in declarative networking, and new declarative approaches to related problems.

1. INTRODUCTION

Over the past decade there has been intense interest in the design of new network protocols. This has been driven from below by an increasing diversity in network architectures (including wireless networks, satellite communications, and delay-tolerant rural networks) and from above by a quickly growing suite of networked applications (peer-to-peer systems, sensor networks, content distribution, etc.)

Network protocol design and implementation is a challenging process. This is not only because of the distributed nature and large scale of typical networks, but also because of the need to balance the extensibility and flexibility of these protocols on one hand, and their robustness and efficiency on the other hand. One needs to look no further than the Internet for an illustration of these hard trade-offs. Today's Internet routing protocols, while arguably robust and efficient, make it hard to accommodate the needs of new applications such as improved resilience and higher throughput. Upgrading even a single router is hard. Getting a distributed routing protocol implemented correctly is even harder. Moreover, in order to change or upgrade a deployed routing protocol today, one must get access to *each* router to modify its software. This process is made even more tedious and error-prone by the use of conventional programming languages.

In this paper, we introduce *declarative networking*, an application of database query language and processing techniques to the domain of networking. Declarative networking is based on the observation that network protocols deal at

their core with computing and maintaining distributed state (e.g., routes, sessions, performance statistics) according to basic information locally available at each node (e.g., neighbor tables, link measurements, local clocks) while enforcing constraints such as local routing policies. Recursive query languages studied in the deductive database literature²⁷ are a natural fit for expressing the relationship between base data, derived data, and the associated constraints. As we demonstrate, simple extensions to these languages and their implementations enable the natural expression and efficient execution of network protocols.

In a series of papers with colleagues, we have described how we implemented and deployed this concept in the *P2* declarative networking system.²⁴ Our high-level goal has been to provide software environments that can accelerate the process of specifying, implementing, experimenting with and evolving designs for network architectures.

As we describe in more detail below, declarative networking can reduce program sizes by orders of magnitude relative to traditional approaches, in some cases resulting in programs that are line-for-line translations of pseudocode in networking research papers. Declarative approaches also open up opportunities for automatic protocol optimization and hybridization, program checking, and debugging.

2. LANGUAGE

In this section, we present an overview of the Network Datalog (*NDlog*) language for declarative networking. The *NDlog* language is based on extensions to traditional Datalog, a well-known recursive query language designed for querying graph-structured data in a centralized database. *NDlog*'s integration of networking and logic is unique from the perspectives of both domains. As a network protocol language, it is notable for the absence of any communication primitives like "send" or "receive"; instead, communication is implicit in a simple high-level specification of data partitioning. In comparison to traditional logic languages, it is enhanced to capture typical network realities including distribution, link-layer constraints on communication (and hence deduction), and soft-state⁸ semantics.

We step through an example to illustrate the standard execution model for Datalog, and demonstrate its close connections to routing protocols, recursive network graph computations, and distributed state management. We then describe the *Overlog*²¹ extensions to the *NDlog* language that support soft-state data and events.

A previous version of this paper was published in *Proceedings of ACM SIGMOD's International Conference of Management of Data* (2006).

2.1. Introduction to Datalog

We first provide a short review of Datalog, following the conventions in Ramakrishnan and Ullman's survey.²⁷ A Datalog program consists of a set of declarative *rules* and an optional *query*. Since these programs are commonly called “*recursive queries*” in the database literature, we use the term “query” and “program” interchangeably when we refer to a Datalog program.

A Datalog *rule* has the form $p :- q_1, q_2, \dots, q_n$, which can be read informally as “ q_1 and q_2 and ... and q_n implies p .” p is the *head* of the rule, and q_1, q_2, \dots, q_n is a list of *literals* that constitutes the *body* of the rule. Literals are either *predicates* over *fields* (variables and constants), or functions (formally, *function symbols*) applied to fields. The rules can refer to each other in a cyclic fashion to express recursion. The order in which the rules are presented in a program is semantically immaterial. The commas separating the predicates in a rule are logical conjuncts (*AND*); the order in which predicates appear in a rule body also has no semantic significance, though most implementations (including ours) employ a left-to-right execution strategy. Predicates in the rule body are matched (or *joined*) based on their common variables to produce the output in the rule head. The *query* (denoted by a reserved rule label *Query*) specifies the output of interest.

The predicates in the body and head of traditional Datalog rules are relations, and we refer to them interchangeably as predicates or relations. In our work, every relation has a *primary key*, which is a set of fields that uniquely identifies each tuple within the relation. In the absence of other information, the primary key is the full set of fields in the relation.

By convention, the names of predicates, function symbols, and constants begin with a lowercase letter, while variable names begin with an uppercase letter. Most implementations of Datalog enhance it with a limited set of side-effect-free function calls including standard infix arithmetic and various simple string and list manipulations (which start with “f_” in our syntax). Aggregate constructs are represented as aggregation functions with field variables within angle brackets ($\langle \rangle$).

2.2. NDLog by example

We introduce *NDlog* using an example program shown below that implements the *path-vector protocol*, which computes in a distributed fashion, for every node, the shortest paths to all other nodes in a network. The path-vector protocol is used as the base routing protocol for exchanging routes among Internet Service Providers.

```

sp1 path(@Src, Dest, Path, Cost) :- link(@Src, Dest, Cost),
  Path=f_init(Src, Dest).
sp2 path(@Src, Dest, Path, Cost) :- link(@Src, Nxt, Cost1),
  path(@Nxt, Dest, Path2, Cost2), Cost=Cost1+Cost2,
  Path=f_concatPath(Src, Path2).
sp3 spCost(@Src, Dest, min<Cost>) :- path(@Src, Dest, Path, Cost).
sp4 shortestPath(@Src, Dest, Path, Cost) :-
  spCost(@Src, Dest, Cost), path(@Src, Dest, Path, Cost).
Query shortestPath(@Src, Dest, Path, Cost).

```

The program has four rules (which for convenience we label *sp1-sp4*), and takes as input a base (*extensional*) relation *link*(*Src*, *Dest*, *Cost*). Rules *sp1-sp2* are used to derive “paths” in the graph, represented as tuples in the derived (*intensional*) relation *path*(*Src*, *Dest*, *Path*, *Cost*). The *Src* and *Dest* fields represent the source and destination endpoints of the path, and *Path* is the actual path from *Src* to *Dest*. The number and types of fields in relations are inferred from their (consistent) use in the program’s rules.

Since network protocols are typically computations over distributed network state, one of the important requirements of *NDlog* is the ability to support rules that express distributed computations. *NDlog* builds upon traditional Datalog by providing control over the storage location of tuples explicitly in the syntax via *location specifiers*. Each location specifier is a field within a predicate that dictates the partitioning of the table. To illustrate, in the above program, each predicate has an “@” symbol prepended to a single field denoting the location specifier. Each tuple generated is stored at the address determined by its location specifier. For example, each *path* and *link* tuple is stored at the address held in its first field @*Src*.

Rule *sp1* produces *path* tuples directly from existing *link* tuples, and rule *sp2* recursively produces *path* tuples of increasing cost by matching (joining) the destination fields of existing links to the source fields of previously computed paths. The matching is expressed using the repeated *Nxt* variable in *link*(*Src*, *Nxt*, *Cost1*) and *path*(*Nxt*, *Dest*, *Path2*, *Cost2*) of rule *sp2*. Intuitively, rule *sp2* says that “if there is a link from node *Src* to node *Nxt*, and there is a path from node *Nxt* to node *Dest* along a path *Path2*, then there is a path *Path* from node *Src* to node *Dest* where *Path* is computed by prepending *Src* to *Path2*.” The matching of the common *Nxt* variable in *link* and *path* corresponds to a *join* operation used in relational databases.

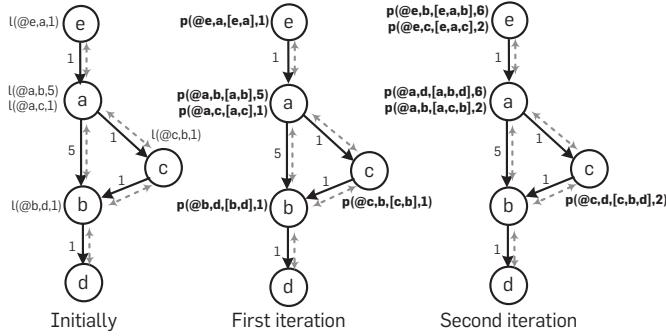
Given the *path* relation, rule *sp3* derives the relation *spCost*(*Src*, *Dest*, *Cost*) by computing the minimum cost *Cost* for each source and destination for all input paths. Rule *sp4* takes as input *spCost* and *path* tuples and then finds *shortestPath*(*Src*, *Dest*, *Path*, *Cost*) tuples that contain the shortest path *Path* from *Src* to *Dest* with cost *Cost*. Last, as denoted by the *Query* label, the *shortestPath* table is the output of interest.

2.3. Shortest path execution example

We step through an execution of the *shortest-path NDlog* program above to illustrate derivation and communication of tuples as the program is computed. We make use of the example network in Figure 1. Our discussion is necessarily informal since we have not yet presented our distributed implementation strategies; in the next section, we show in greater detail the steps required to generate the execution plan. Here, we focus on a high-level understanding of the data movement in the network during query processing.

For ease of exposition, we will describe communication in synchronized *iterations*, where at each iteration, each

Figure 1. Nodes in the network are running the shortest-path program. We only show newly derived tuples at each iteration.



network node generates *paths* of increasing hop count, and then propagates these paths to neighbor nodes along links. We show only the derived paths communicated along the solid lines. In actual query execution, derived tuples can be sent along the bidirectional network links (dashed links).

In the first iteration, all nodes initialize their local path tables to 1-hop paths using rule *sp1*. In the second iteration, using rule *sp2*, each node takes the input paths generated in the previous iteration, and computes 2-hop paths, which are then propagated to its neighbors. For example, *path(@a, d, [a, b, d], 6)* is generated at node *b* using *path(@b, d, [b, d], 1)* from the first iteration, and propagated to node *a*. In fact, many network protocols propagate only the *nextHop* and avoid sending the entire path vector.

As paths are computed, the shortest one is incrementally updated. For example, node *a* computes the cost of the shortest path from *a* to *b* as 5 with rule *sp3*, and then finds the corresponding shortest path *[a, b]* with rule *sp4*. In the next iteration, node *a* receives *path(@a, b, [a, c, b], 2)* from node *c*, which has lower cost compared to the previous shortest cost of 5, and hence *shortestPath(@a, b, [a, c, b], 2)* replaces the previous tuple (the first two fields of source and destination are the primary key of this relation).

Interestingly, while *NDlog* is a language to describe networks, there are no explicit communication primitives. All communication is implicitly generated during rule execution as a result of data placement specifications. For example, in rule *sp2*, the *path* and *link* predicates have different location specifiers, and in order to execute the rule body of *sp2* based on their matching fields, *link* and *path* tuples have to be shipped in the network. It is the movement of these tuples that generates the messages for the resulting network protocol.

2.4. Language extensions

We describe two extensions to the *NDlog* language: *link-restricted rules* that limit the expressiveness of the language in order to capture physical network constraints, and a *soft-state storage model* commonly used in networking protocols. **Link-Restricted Rules:** In the above path vector protocol, the evaluation of a rule must depend only on communication

along the physical links. In order to send a message in a low-level network, there needs to be a link between the sender and receiver. This is not a natural construct in Datalog. Hence, to model physical networking components where full connectivity is not available, *NDlog* provides restrictions ensuring that rule execution results in communication only among nodes that are physically connected with a bidirectional link. This is syntactically achieved with the use of the special *link* predicate in the form of *link-restricted rules*. A *link-restricted rule* is either a *local rule* (having the same location specifier variable in each predicate), or a rule with the following properties:

1. There is exactly one *link* predicate in the body.
2. All other predicates (including the head predicate) have their location specifier set to either the first (source) or second (destination) field of the *link* predicate.

This syntactic constraint precisely captures the requirement that we be able to operate directly on a network whose link connectivity is not a full mesh. Further, as we demonstrate in Section 3, link-restriction also guarantees that all programs with only link-restricted rules can be rewritten into a canonical form where every rule body can be evaluated on a single node, with communication to a head predicate along links. The following is an example of a link-restricted rule:

```
p(@Dest,...) :- link(@Src, Dest...), p1(@Src, ...),
               p2(@Src, ...), ..., pn(@Src, ...).
```

The rule body of this example is executed at *@Src* and the resulting *p* tuples are sent to *@Dest*, preserving the communication constraints along links. Note that the body predicates of this example all have the same location specifier: *@Src*, the source of the link. In contrast, rule *sp2* of the *shortest path* program is link-restricted but has some relations whose location specifier is the source, and others whose location specifier is the destination; this needs to be rewritten to be executable in the network, a topic we return to in Section 3.2.

In a fully connected network environment, an *NDlog* parser can be configured to bypass the requirement for link-restricted rules.

Soft-State Storage Model: Many network protocols use the soft-state approach to maintain distributed state. In the soft-state storage model, stored data have an associated *lifetime* or time-to-live (TTL). A soft-state datum needs to be periodically refreshed; if more time than a TTL passes without a datum being refreshed, that datum is deleted. Soft state is often favored in networking implementations because in a very simple manner it provides well-defined eventual consistency semantics. Intuitively, periodic refreshes to network state ensure that the eventual values are obtained even if there are transient errors such as reordered messages, node disconnection, or link failures. However, when persistent failures occur, no coordination is required to register the

failure: any data provided by failed nodes are organically “forgotten” in the absence of refreshes.

We introduced soft-state into the *Overlog*²¹ declarative networking language, an extension of *NDlog*. One additional feature of *Overlog* is the availability of a materialized keyword at the beginning of each program to specify the TTL of predicates. For example, the definition `materialized(link, {1,2}, 10)` specifies that the link table has its primary key set to the first and second fields (denoted by `{1,2}`), and each link tuple has a lifetime of 10 seconds. If the TTL is set to infinity, the predicate will be treated as *hard state*, i.e., a traditional relation that does not involve timeout-based deletion.

The *Overlog* soft-state storage semantics are as follows. When a tuple is derived, if there exists another tuple with the same primary key but differences on other fields, an *update* occurs, in which the new tuple replaces the previous one. On the other hand, if the two tuples are identical, a *refresh* occurs, in which the existing tuple is extended by its TTL.

If a given predicate has no associated `materialize` declaration, it is treated as an *event* predicate: a soft-state predicate with `TTL=0`. Event predicates are transient tables, which are used as input to rules but not stored. They are primarily used to “trigger” rules periodically or in response to network events. For example, utilizing *Overlog*'s built-in `periodic` event predicate, the following rule enables node X to generate a ping event every 10 seconds to its neighbor Y denoted in the `link(@X,Y)` predicate:

```
ping(@Y,X) :- periodic(@X,10), link(@X,Y).
```

Subtleties arise in the semantics of rules that mix event, soft-state and hard-state predicates across the head and body. One issue involves the expiry of soft-state and event tuples, as compared to deletion of hard-state tuples. In a traditional hard-state model, deletions from a rule's body relations require revisions to the derived head relation to maintain consistency of the rule. This is treated by research on materialized view maintenance.¹³ In a pure soft-state model, the head and body predicates can be left inconsistent with each other for a time, until head predicates expire due to the lack of refreshes from body predicates. Mixtures of the two models become more subtle. We provided one treatment of this issue,¹⁹ which has subsequently been revised with a slightly different interpretation.⁹ There is still

some debate about the desired semantics, focusing on attempts to provide an intuitive declarative representation while enabling familiar event-handler design patterns used by protocol developers.

3. EXECUTION PLAN GENERATION

Our runtime execution of *NDlog* programs differs from the traditional implementation patterns for both network protocols and database queries. Network protocol implementations often center around local state machines that emit messages, triggering state transitions at other state machines. By contrast, the runtime systems we have built for *NDlog* and *Overlog* are distributed dataflow execution engines, similar in spirit to those developed for parallel database systems, and echoed in recent parallel map-reduce implementations. However, the recursion in Datalog introduces cycles into these dataflows. The combination of recursive flows and the asynchronous communication inherent in wide-area systems presents new challenges that we had to overcome.

In this section, we describe the steps required to automatically generate a distributed dataflow execution plan from an *NDlog* program. We first focus on generating an execution plan in a centralized implementation, before extending the techniques to the network scenario.

3.1. Centralized plan generation

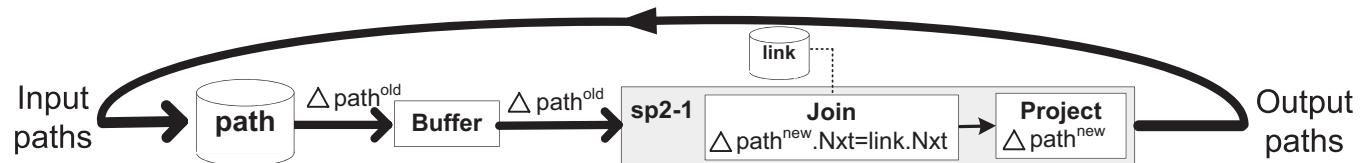
In generating the centralized plan, we utilize the well-known *semi-naïve fixpoint*³ Datalog evaluation mechanism that ensures no redundant evaluations. As a quick review, in semi-naïve (SN) evaluation, input tuples computed in the previous iteration of a recursive rule execution are used as input in the current iteration to compute new tuples. Any new tuples that are generated for the first time in the current iteration, and only these new tuples, are then used as input to the next iteration. This is repeated until a fixpoint is achieved (i.e., no new tuples are produced).

The SN rewritten rule for rule `sp2` is shown below:

```
sp2-1 Δpathnew (@Src,@Dest,Path,Cost) :-
    link(@Src,Nxt,Cost1),
    Δpathold(@Nxt,Dest,Path2,Cost2),
    Cost=Cost1+Cost2,
    Path=f_concatPath(Src,Path2).
```

Figure 2 shows the dataflow realization for a centralized implementation of rule `sp2-1` using the conventions of *P2*.²⁴

Figure 2. Rule strand for a centralized implementation of rule sp2-1 in P2. Output paths that are generated from the strand are “wrapped back” as input into the same strand.



The *P2* system uses an execution model inspired by database query engines and the Click modular router,¹⁴ which consists of elements that are connected together to implement a variety of network and flow control components. In addition, *P2* elements include database operators (such as joins, aggregation, selections, and projects) that are directly generated from the rules.

We will briefly explain how the SN evaluation is achieved in *P2*. Each SN rule is implemented as a *rule strand*. Each strand consists of a number of relational operators for selections, projections, joins, and aggregations. The example strand receives new `delta_path_old` tuples generated in the previous iteration to generate new paths (`delta_path_new`), which are then inserted into the path table (with duplicate elimination) for further processing in the next iteration.

In Algorithm 1, we show the pseudocode for a centralized implementation of multiple SN rule strands where each rule has the form:

$$\Delta p_j^{new} \leftarrow p_1^{old}, \dots, p_{k-1}^{old}, \Delta p_k^{old}, p_{k+1}, \dots, p_n, b_1, b_2, \dots, b_m.$$

p_1, \dots, p_n are recursive predicates and b_1, \dots, b_m are base predicates. Δp_k^{old} refers to p_k tuples generated for the first time in the previous iteration. p_k^{old} refers to all p_k tuples generated before the previous iteration. These rules are logically equivalent to rules of the form:

$$\Delta p_j^{new} \leftarrow p_1, \dots, p_{k-1}, \Delta p_k^{old}, p_{k+1}, \dots, p_n, b_1, b_2, \dots, b_m.$$

The earlier rules have the advantage of avoiding redundant inferences within each iteration.

Algorithm 1 Semi-naïve (SN) Evaluation in *P2*

```

while ∃B_k.size > 0
    ∀B_k where B_k.size > 0, Δp_k^{old} ← B_k.flush()
    execute all rule strands
    foreach recursive predicate p_j
        p_j^{old} ← p_j^{old} ∪ Δp_j^{old}
        B_j ← Δp_j^{new} - p_j^{old}
        p_j ← p_j^{old} ∪ B_j
        Δp_j^{new} ← φ
    
```

In the algorithm, B_k denotes the buffer for p_k tuples generated in the previous iteration (Δp_k^{old}). Initially, p_k , p_k^{old} , Δp_k^{old} , and Δp_k^{new} are empty. As a base case, we execute all the rules to generate the initial p_k tuples, which are inserted into the corresponding B_k buffers. Each subsequent iteration of the while loop consists of flushing all existing Δp_k^{old} tuples from B_k and executing all rule strands to generate Δp_k^{new} tuples, which are used to update p_k^{old} , B_k , and p_k accordingly. Note that only new p_k tuples generated in the current iteration are inserted into B_k for use in the next iteration. Fixpoint is reached when all buffers are empty.

3.2. Distributed plan generation

In the distributed implementation of the path-vector program, nonlocal rules whose body predicates have different location specifiers cannot be executed at a single node,

since the tuples that must be joined are situated at different nodes in the network. A *rule localization* rewrite step ensures that all tuples to be joined are at the same node. This allows a rule body to be locally computable.

Consider the rule `sp2` from the shortest-path program, where the link and path predicates have different location specifiers. These two predicates are joined by a common `@Nxt` address field. Figure 3 shows the corresponding logical query plan depicting the distributed join. The clouds represent an “exchange”-like operator¹¹ that forwards tuples from one network node to another; clouds are labeled with the `link` attribute that determines the tuple's recipient. The first cloud (`link.Nxt`) sends `link` tuples to the neighbor nodes indicated by their destination address fields, in order to join with matching `path` tuples stored by their source address fields. The second cloud (`path.Src`) transmits for further processing new path tuples computed from the join, setting the recipient according to the source address field.

Based on the above distributed join, rule `sp2` can be rewritten into the following two rules. Note that all predicates in the body of `sp2a` have the same location specifiers; the same is true of `sp2b`.

```

sp2a linkD(@Nxt,Src,Cost) :- link(@Src,Nxt,Cost).
sp2b path(@Src,Dest,Nxt,Path,Cost) :- linkD(@Nxt,Src,Cost1),
                                         path(@Nxt,Dest,Path2,Cost2), Cost=Cost1+Cost2,
                                         Path = f_concatPath(Src,Path2).
    
```

The rewrite is achievable because the `link` and `path` predicates, although at different locations, share a common join address field. The details of the rewrite algorithm and associated proofs are described in a longer article.²⁰

Returning to our example, after rule localization we perform the SN rewrite, and then generate the rule strands shown in Figure 4. Unlike the centralized strand in Figure 2, there are now three rule strands. The extra two strands (`sp2a@Src` and `sp2b-2@Nxt`) are used as follows. Rule strand `sp2a@`

Figure 3. Logical query plan for rule `sp2`.

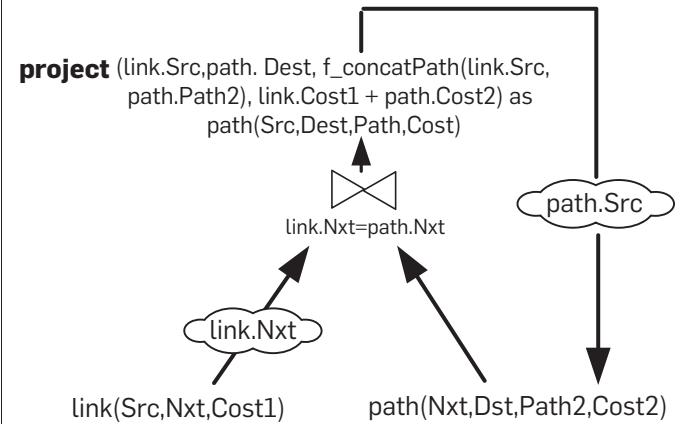
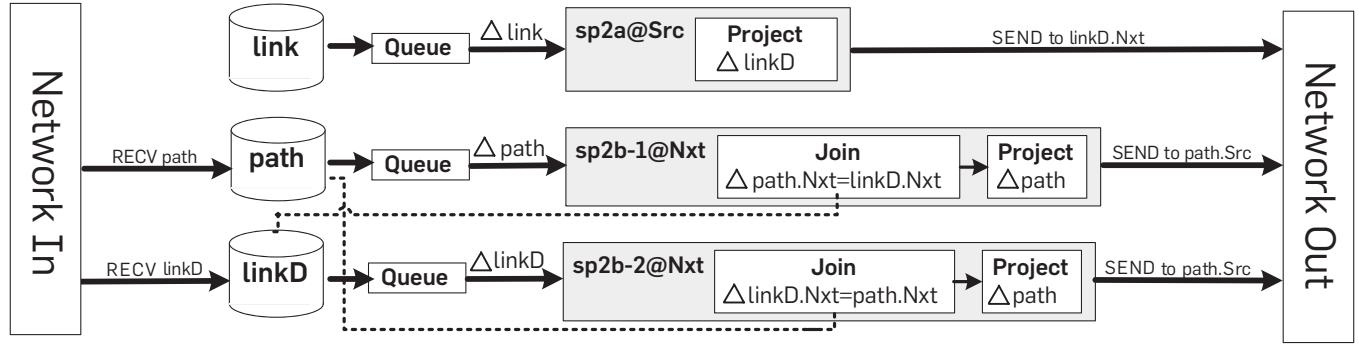


Figure 4. Rule strands for the distributed version of sp2 after localization in P2.

src sends all existing links to the destination address field as linkD tuples. Rule strand sp2b-2@Nxt takes the new linkD tuples it received via the network and performs a join operation with the local path table to generate new paths.

3.3. Relaxing semi-naïve evaluation

In our distributed implementation, the execution of rule strands can depend on tuples arriving via the network, and can also result in new tuples being sent over the network. Traditional SN evaluation completely evaluates all rules on a given set of facts, i.e., completes the *iteration*, before considering any new facts. In a distributed execution environment where messages can be delayed or lost, the completion of an iteration in the traditional sense can only be detected by a consensus computation across multiple nodes, which is expensive; further, the requirement that many nodes complete the iteration together (a “barrier synchronization” in parallel computing terminology) limits parallelism significantly by restricting the rate of progress to that of the slowest node.

We address this by making the notion of iteration local to a node. New facts might be generated through local rule execution, or might be received from another node while a local iteration is in progress. We proposed and proved correct a variation of SN iteration called *pipelined semi-naïve* (PSN) to handle this situation.²⁰ PSN extends SN to work in an asynchronous distributed setting. PSN relaxes SN evaluation to the extreme of processing each tuple as it is received. This provides opportunities for additional optimizations on a per-tuple basis. New tuples that are generated from the SN rules, as well as tuples received from other nodes, are used immediately to compute new tuples without waiting for the current (local) iteration to complete.

Algorithm 2 Pipelined Semi-naïve (PSN) Evaluation

```

while  $\exists Q_k \cdot \text{size} > 0$ 
   $t_k^{old,i} \leftarrow Q_k \cdot \text{dequeueTuple}()$ 
  foreach rule strand execution
     $\Delta p_j^{new,i+1} :=$ 
       $p_1, \dots, p_{k-1}, t_k^{old,i}, p_{k+1}, \dots, p_n, b_1, b_2, \dots, b_m$ 
    foreach  $t_j^{new,i+1} \in \Delta p_j^{new,i+1}$ 
      if  $t_j^{new,i+1} \notin p_j$ 
        then  $p_j \leftarrow p_j \cup t_j^{new,i+1}$ 
         $Q_j \cdot \text{enqueueTuple}(t_j^{new,i+1})$ 
  
```

Algorithm 2 shows the pseudocode for PSN. Each tuple, denoted t , has a superscript (*old/new*, i) where i is its corresponding iteration number in SN evaluation. Each processing step in PSN consists of dequeuing a tuple $t_k^{old,i}$ from Q_k and then using it as input into all corresponding rule strands. Each resulting $t_j^{new,i+1}$ tuple is pipelined, stored in its respective p_j table (if a copy is not already there), and enqueued into Q_j for further processing. Note that in a distributed implementation, Q_j can be a queue on another node, and the node that receives the new tuple can immediately process the tuple after the enqueue into Q_j . For example, the dataflow in Figure 4 is based on a distributed implementation of PSN, where incoming path and linkD tuples received via the network are stored locally, and enqueued for processing in the corresponding rule strands.

To fully pipeline evaluation, we have also removed the distinctions between p_j^{old} and p_j in the rules. Instead, a timestamp (or monotonically increasing sequence number) is added to each tuple at arrival, and the join operator matches each tuple only with tuples that have the same or older timestamp. This allows processing of tuples immediately upon arrival, and is natural for network message handling. This represents an alternative “book-keeping” strategy to the rewriting used in SN to ensure no repeated inferences. Note that the timestamp only needs to be assigned locally, since all the rules are localized.

We have proven elsewhere²⁰ that PSN generates the same results as SN and does not repeat any inferences, as long as the *NDlog* program is monotonic and messages between two network nodes are delivered in FIFO order.

3.4. Incremental maintenance

In practice, most network protocols are executed over a long period of time, and the protocol incrementally updates and repairs routing tables as the underlying network changes (link failures, node departures, etc.). To better map into practical networking scenarios, one key distinction that differentiates the execution of *NDlog* from earlier work in Datalog is our support for continuous rule execution and result materialization, where all tuples derived from *NDlog* rules are materialized and incrementally updated as the underlying network changes. As in network protocols, such incremental maintenance is required both for timely updates and for avoiding the overhead of recomputing all routing tables “from scratch” whenever there are changes

to the underlying network. In the presence of insertions and deletions to base tuples, our original incremental view maintenance implementation utilizes the count algorithm¹³ that ensures only tuples that are no longer derivable are deleted. This has subsequently been improved¹⁸ via the use of a compact form of data provenance encoded using binary decision diagrams shipped with each derived tuple.

In general, updates could occur very frequently, at a period that is shorter than the expected time for a typical query to reach a fixpoint. In that case, query results can never fully reflect the state of the network. We focus our analysis instead on a *bursty* model. In this weaker, but still fairly realistic model, updates are allowed to happen during query processing. However, we make the assumption that after a burst of updates, the network eventually *quiesces* (does not change) for a time long enough to allow all the queries in the system to reach a fixpoint. Unlike the continuous model, the bursty model is amenable to simpler analysis; our results on that model provide some intuition as to the behavior in the continuous update model as well.

We have proven²⁰ that in the presence of reliable, in-order delivery of messages, link-restricted *NDlog* rules under the bursty model achieve a variant of the typical distributed systems notion of *eventual consistency*, where the eventual state of the quiescent system corresponds to what would be achieved by rerunning the queries from scratch in that state.

4. USE CASES

In the past 3 years, since the introduction of declarative networking and the release of *P2*, several applications have been developed. We describe two of the original use cases that motivated our work and drove several of our language and system designs: *safe extensible routers* and *overlay network development*. We will briefly mention new applications in Section 5.

4.1. Declarative routing

The Internet's core routing infrastructure, while arguably robust and efficient, has proven to be difficult to evolve to accommodate the needs of new applications. Prior research on this problem has included new hard-coded routing protocols on the one hand, and fully extensible Active Networks³¹ on the other. *Declarative routing*²¹ explores a new point in this design space that aims to strike a better balance between the extensibility and robustness of a routing infrastructure.

With declarative routing, a routing protocol is implemented by writing a simple query in *NDlog*, which is then executed in a distributed fashion at the nodes that receive the query. Declarative routing can be viewed as a restrictive instantiation of Active Networks for the control plane, which aims to balance the concerns of expressiveness, performance and security, properties which are needed for an extensible routing infrastructure to succeed.

Security is a key concern with any extensible system particularly when it relates to nontermination and the consumption of resources. *NDlog* is amenable to static analysis due to its connections to Datalog. In terms of query execution, *pure Datalog* (without any negation, aggregation, or

function symbols) has polynomial time and space complexities in the size of the input. This property provides a natural bound on the resource consumption. However, many extensions of Datalog (including *NDlog*) augment the core language in various ways, invalidating its polynomial complexity.

Fortunately, static analysis tests have been developed to check for the termination of an augmented Datalog query on a given input.¹⁵ In a nutshell, these tests identify recursive definitions in the query rules, and check whether these definitions terminate. Examples of recursive definitions that terminate are ones that evaluate monotonically increasing (decreasing) predicates whose values are upper (lower) bounded. Moreover, the declarative framework is amenable to other verification techniques, including theorem proving,³² model checking,²⁵ and runtime verification.²⁸

NDlog can express a variety of well-known routing protocols (e.g., distance vector, path vector, dynamic source routing, link state, multicast) in a compact and clean fashion, typically in a handful of lines of program code. Moreover, higher-level routing concepts (e.g., QoS constraints) can be achieved via simple modifications to these queries. Finally, writing the queries in *NDlog* illustrates surprising relationships between protocols. For example, we have shown that *distance vector* and *dynamic source routing* protocols differ only in a simple, traditional query optimization decision: the order in which a query's predicates are evaluated.

To limit query computation to the relevant portion of the network, we use a query rewrite technique, called *magic sets rewriting*.⁴ Rather than reviewing the Magic Sets optimization here, we illustrate its use in an example. Consider the situation where instead of computing all-pairs shortest paths, we are only in computing the shortest paths from a selected group of source nodes (*magicSrc*) to selected destination nodes (*magicDst*). By modifying rules *sp1-sp4* from the path-vector program, the following computes only paths limited to sources/destinations in the *magicSrc*/*magicDst* tables, respectively.

```

sp1-sd pathDst (@Dest, Src, Path, Cost) :- magicSrc(@Src),
    link(@Src, Dest, Cost), Path=f_init(Src, Dst).
sp2-sd pathDst (@Dest, Src, Path, Cost) :-
    pathDst (@Nxt, Src, Path1, Cost1), link(@Nxt, Dest, Cost2),
    Cost=Cost1+Cost2, Path=f_concatPath(Path1, Dest).
sp3-sd spCost (@Dest, Src, min<Cost>) :- magicDst(@Dest),
    pathDst (@Dest, Src, Path, Cost).
sp4-sd shortestPath (@Dest, Src, Path, Cost) :-
    spCost (@Dest, Src, Cost), pathDst (@Dest, Src, Path, Cost).
Query shortestPath (@Src, Dest, Path, Cost).

```

Our evaluation results²¹ based on running declarative routing protocols on the PlanetLab²⁶ global testbed and in a local cluster show that when all nodes issue the same query, the query execution has similar scalability properties as the traditional distance vector and path-vector protocols. For example, the convergence latency for the path-vector program is proportional to the network diameter, and converges in the same time as the path-vector protocol. Second,

the per-node communication overhead increases linearly with the number of nodes. This suggests that our approach does not introduce any fundamental overheads. Moreover, when there are few nodes issuing the same query, query optimization and work-sharing techniques can significantly reduce the communication overhead.

One promising direction stems from our surprising observation on the synergies between query optimization and network routing: a wired protocol (distance-vector protocol) can be translated to a wireless protocol (dynamic source routing) by applying the standard database optimizations of magic sets rewrite and predicate reordering. More complex applications of query optimization have begun to pay dividends in research, synthesizing new hybrid protocols from traditional building blocks.^{5, 17} Given the proliferation of new routing protocols and a diversity of new network architecture proposals, the connection between query optimizations and network routing suggests that query optimizations may help us inform new routing protocol designs and allow the hybridization of protocols within the network.

4.2. Declarative overlays

In declarative routing, we demonstrated the flexibility and compactness of *NDlog* for specifying a variety of routing protocols. In practice, most distributed systems are much more complex than simple routing protocols; in addition to routing, they typically also perform application-level message forwarding and handle the formation and maintenance of a network as well.

In our subsequent work on *declarative overlays*,²¹ we demonstrate the use of the *Overlog* to implement practical application-level *overlay networks*. An overlay network is a virtual network of nodes and logical links that is built on top of an existing network with the purpose of implementing a network service that is not available in the existing network. Examples of overlay networks on today's Internet include commercial content distribution networks,¹ peer-to-peer (P2P) applications for file-sharing¹⁰ and telephony,²⁹ as well as a wide range of experimental prototypes running on PlanetLab.

In declarative overlays, applications submit to *P2* a concise *Overlog* program that describes an overlay network, and the *P2* system executes the program to maintain routing tables, perform neighbor discovery and provide forwarding for the overlay.

Declarative overlay programs are more complex than routing due to the handling of message delivery, acknowledgments, failure detection, and timeouts. These programs also heavily utilize soft-state features in *Overlog* not present in the original *NDlog* language. Despite the increased complexity, we demonstrate that our *NDlog* programs are significantly more compact compared to equivalent C++ implementations. For instance, the Narada⁷ mesh formation and a full-fledged implementation of the Chord distributed hash table³⁰ are implemented in 16 and 48 rules, respectively. In the case of the Chord DHT presented by Loo et al.,¹⁹ there are rules for performing various aspects of Chord, including initial joining of the Chord network, Chord ring maintenance, finger table maintenance, recursive Chord lookups, and failure detection of neighbors.

We note that our Chord implementation is roughly two orders of magnitude less code than the original C++ implementation. This is a quantitative difference that is sufficiently large that it becomes qualitative: in our opinion (and experience), declarative programs that are a few dozen lines of code are markedly easier to understand, debug, and extend than thousands of lines of imperative code. Moreover, we demonstrate^{19, 21} that our declarative overlays achieve the expected high-level properties of their respective overlay networks for both static and dynamic networks. For example, in a static network of up to 500 nodes, the measured hop-count of lookup requests in the Chord network conformed to the theoretical average of $0.5 \times \log_2 N$ hops, and the latency numbers were within the same order of magnitude as published Chord numbers.

5. CONCLUSION

In Jim Gray's Turing Award Lecture,¹² one of his grand challenges was the development of "automatic programming" techniques that would be (a) 1000× easier for people to use, (b) directly compiled into working code, and (c) suitable for general purpose use. Butler Lampson reiterated the first two points in a subsequent invited article, but suggested that they might be more tractable in domain-specific settings.¹⁶

Declarative Networking has gone a long way toward Gray's vision, if only in the domain of network protocol implementation. On multiple occasions we have seen at least two orders of magnitude reduction in code size, with the reduced linecount producing qualitative improvements. In the case of Chord, a multi-thousand-line C++ library was rewritten as a declarative program that fits on a single sheet of paper—a software artifact that can be studied and holistically understood by a programmer in a single sitting.

We have found that a high-level declarative language not only simplifies a programmer's work, but refocuses the programming task on appropriately high-level issues. For example, our work on declarative routing concluded that discussions of routing in wired vs. wireless networks should not result in different protocols, but rather in different compiler optimizations for the same simple declaration, with the potential to be automatically blended into new hybrid strategies as networks become more diverse.^{5, 17} This lifting of abstractions seems well suited to the increasing complexity of modern networking, introducing software malleability by minimizing the affordances for over-engineering solutions to specific settings.

Since we began our work on this topic, there has been increasing evidence that declarative, data-centric programming has much broader applicability. Within the networking domain, we have expanded in multiple directions from our initial work on routing, to encompass low-level network issues at the wireless link layer⁶ to higher-level logic including both overlay networks²¹ and applications like code dissemination, object tracking, and content distribution. Meanwhile, a variety of groups have been using declarative programming ideas in surprising ways in many other domains. We briefly highlight two of our own follow-on efforts.

Secure Distributed Systems: Despite being developed independently by separate communities, logic-based security specifications and declarative networking programs both

extend Datalog in surprisingly similar ways: by supporting the notion of context (location) to identify components (nodes) in distributed systems. The *Secure Network Datalog*³³ language extends *NDlog* with basic security constructs for implementing secure distributed systems, which are further enhanced with type checking and meta-programmability in the *LBTrust*²³ system for supporting various forms of encryption/authentication, delegation, for distributed trust management.

Datacenter Programming: The BOOM² project is exploring the use of declarative languages in the setting of Cloud Computing. Current cloud platforms provide developers with sequential programming models that are a poor match for inherently distributed resources. To illustrate the benefits of declarative programming in a cloud, we used *Overlog* as the basis for a radically simplified and enhanced reimplementation of a standard cloud-based analytics stack: the Hadoop File System (HDFS) and MapReduce infrastructure. Our resulting system is API-compatible with Hadoop, with performance that is equivalent or better. More significantly, the high-level *Overlog* specification of key Hadoop internals enabled a small group of graduate students to quickly add sophisticated distributed features to the system that are not in Hadoop: hot standby master nodes supported by MultiPaxos consensus, scaleout of (quorums of) master nodes via data partitioning, and implementations of new scheduling protocols and query processing strategies.

In addition to these two bodies of work, others have successfully adopted concepts from declarative networking, in the areas of mobility-based overlays, adaptively hybridized mobile ad-hoc networks, overlay network composition, sensor networking, fault-tolerant protocols, network configuration, replicated filesystems, distributed machine learning algorithms, and robotics. Outside the realm of networking and distributed systems, there has been an increasing use of declarative languages—many rooted in Datalog—to a wide range of problems including natural language processing, compiler analysis, security, and computer games. We maintain a list of related declarative languages and research projects at <http://declarativity.net/related>.

For the moment, these various efforts represent individual instances of Lampson's domain-specific approach to Gray's automatic programming challenge. In the coming years, it will be interesting to assess whether these solutions prove fruitful, and whether it is feasible to go after Gray's challenge directly: to deliver an attractive general-purpose declarative programming environment that radically simplifies wide range of tasks. □

References

1. Akamai. Akamai Content Distribution Network. 2006. <http://www.akamai.com>.
2. Alvaro, P., Condie, T., Conway, N., Elmeleegy, K., Hellerstein, J.M., Sears, R.C. BOOM: Data-centric programming in the datacenter. Technical Report UCB/EECS-2009-98, EECS Department, University of California, Berkeley, Jul 2009.
3. Balbin I., Ramamohanarao, K. A generalization of the differential approach to recursive query evaluation. *J. Logic Prog.* 4, 3 (1987), 259–262.
4. Banilhon, F., Maier, D., Sagiv, Y., Ullman, J. Magic sets and other strange ways to implement logic programs. In *Proceedings of ACM SIGMOD International Conference on Management of Data* (1986).
5. Chu, D., Hellerstein, J. Automating rendezvous and proxy selection in sensor networks. In *Eighth International Conference on Information Processing in Sensor Networks (IPSN)* (2009).
6. Chu, D.C., Popa, L., Tavakoli, A., Hellerstein, J.M., Levis, P., Shenker, S., Stoica, I. The design and implementation of a declarative sensor network system. In *5th ACM Conference on Embedded Networked Sensor Systems (SenSys)* (2007).
7. Chu, Y.-H., Rao, S.G., Zhang, H. A case for end system multicast. In *Proceedings of ACM SIGMETRICS* (2000), 1–12.
8. Clark, D.D. The design philosophy of the DARPA internet protocols. In *Proceedings of ACM SIGCOMM Conference on Data Communication* (Stanford, CA, 1988), ACM, 106–114.
9. Condie, T., Chu, D., Hellerstein, J.M., Maniatis, P. Evita raced: metacomilation for declarative networks. In *Proceedings of VLDB Conference* (2008).
10. Gnutella. <http://www.gnutella.com>.
11. Graefe, G. Encapsulation of parallelism in the volcano query processing system. In *Proceedings of ACM SIGMOD International Conference on Management of Data* (1990).
12. Gray, J. What next? A few remaining problems in information technology. SIGMOD Conference 1999. ACM Turing Award Lecture, Video. *ACM SIGMOD Digital Symposium Collection* 2, 2 (2000).
13. Gupta, A., Mumick, I.S., Subrahmanian, V.S. Maintaining views incrementally. In *Proceedings of ACM SIGMOD International Conference on Management of Data* (1993).
14. Kohler, E., Morris, R., Chen, B., Jannotti, J., Kaashoek, M.F. The click modular router. *ACM Trans. Comp. Sys.* 18, 3 (2000), 263–297.
15. Krishnamurthy, R., Ramakrishnan, R., Shmueli, O. A framework for testing safety and effective computability. *J. Comp. Sys. Sci.* 52, 1 (1996), 100–124.
16. Lampson, B. Getting computers to understand. *J. ACM* 50, 1 (2003), 70–72.
17. Liu, C., Correa, R., Li, X., Basu, P., Loo, B.T., Mao, Y. Declarative Policy-based adaptive MANET routing. In *17th IEEE International Conference on Network Protocols (ICNP)* (2009).
18. Liu, M., Taylor, N., Zhou, W., Ives, Z., Loo, B.T. Recursive computation of regions and connectivity in networks. In *Proceedings of IEEE Conference on Data Engineering (ICDE)* (2009).
19. Loo, B.T. The Design and Implementation of Declarative Networks (Ph.D. Dissertation). Technical Report UCB/EECS-2006-177, UC Berkeley (2006).
20. Loo, B.T., Condie, T., Garofalakis, M., Gay, D.E., Hellerstein, J.M., Maniatis, P., Ramakrishnan, R., Roscoe, T., Stoica, I. Declarative networking: language, execution and optimization. In *Proceedings of ACM SIGMOD International Conference on Management of Data* (2006).
21. Loo, B.T., Condie, T., Hellerstein, J.M., Maniatis, P., Roscoe, T., Stoica, I. Implementing declarative overlays. In *Proceedings of ACM Symposium on Operating Systems Principles* (2005).
22. Loo, B.T., Hellerstein, J.M., Stoica, I., Ramakrishnan, R. Declarative routing: extensible routing with declarative queries. In *Proceedings of ACM SIGCOMM Conference on Data Communication* (2005).
23. Marczak, W.R., Zook, D., Zhou, W., Aref, M., Loo, B.T. Declarative reconfigurable trust management. In *Proceedings of Conference on Innovative Data Systems Research (CIDR)* (2009).
24. P2: Declarative Networking System. <http://p2.cs.berkeley.edu>.
25. Perez, J.N., Rybalchenko, A., Singh, A. Cardinality abstraction for declarative networking applications. In *Proceedings of Computer Aided Verification (CAV)* (2009).
26. PlanetLab. Global testbed. 2006. <http://www.planet-lab.org/>.
27. Ramakrishnan, R., Ullman, J.D. A survey of research on deductive database systems. *J. Logic Prog.* 23, 2 (1993), 125–149.
28. Singh, A., Maniatis, P., Roscoe, T., Druschel, P. Distributed monitoring and forensics in overlay networks. In *Proceedings of Eurosyst* (2006).
29. Skype. Skype P2P Telephony. 2006. <http://www.skype.com>.
30. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H. Chord: a scalable P2P lookup service for internet applications. In *SIGCOMM* (2001).
31. Tennenhouse, D.L., Smith, J.M., Sincoskie, W.D., Wetherall, D.J., Minden, G.J. A survey of active network research. *IEEE Commun. Mag.* 35, 1 (1997), 80–86.
32. Wang, A., Basu, P., Loo, B.T., Sokolsky, O. Towards declarative network verification. In *11th International Symposium on Practical Aspects of Declarative Languages (PADL)* (2009).
33. Zhou, W., Mao, Y., Loo, B.T., Abadi, M. Unified declarative platform for secure networked information systems. In *Proceedings of IEEE Conference on Data Engineering (ICDE)* (2009).

Boon Thau Loo (boonloo@cis.upenn.edu)
University of Pennsylvania, Philadelphia, PA.

Tyson Condie (tcondie@cs.berkeley.edu)
University of California, Berkeley, CA.

Minos Garofalakis (minos@softnet.tuc.gr)
Technical University of Crete, Greece.

David E. Gay (david.e.gay@intel.com)
Intel Research, Berkeley, CA.

Joseph M. Hellerstein (hellerstein@cs.berkeley.edu)
University of California, Berkeley, CA.

Petros Maniatis (petros.maniatis@intel.com)
Intel Research, Berkeley, CA.

Raghuram Ramakrishnan (ramakris@yahoo-inc.com)
Yahoo! Research, Silicon Valley.

Timothy Roscoe (troscoe@inf.ethz.ch)
ETH Zurich, Switzerland.

Ion Stoica (istoica@cs.berkeley.edu)
University of California, Berkeley, CA.

Technical Perspective

Machine Learning for Complex Predictions

By John Shawe-Taylor

INTEREST IN MACHINE learning can be traced back to the early days of computer science. Alan Turing himself conjectured that some form of automatic learning would be required to endow a computer with artificial intelligence. The development of machine learning, however, has not always lived up to these ambitious beginnings.

The first flood of interest in machine learning was generated by Rosenblatt's perceptron.⁴ The perceptron is a simple thresholded linear classifier that can be trained from an online sequence of examples. Intriguingly, from a theoretical point of view, the number of mistakes during training can be bounded in terms of intuitive geometric properties that measure a problem's difficulty.³ One such property is the margin, or the distance of the most "borderline" example to the perceptron's decision boundary. First highlighted by the mistake bound in perceptrons, the margin would come to play an important role in the subsequent development of machine learning; however, its importance in Rosenblatt's time was not fully realized. Interest in perceptrons waned after a critical study by Minsky and Papert that catalogued their limitations.²

Machine learning developed in other directions during the 1970s–1980s with such innovations as decision trees, rule-learning methods for expert systems, and self-organizing maps. However, in the late 1980s, there was a resurgence of interest in architectures that used perceptrons as basic building blocks. In particular, the limitations highlighted by Minsky and Papert were overcome by multilayer networks in which perceptron-like nodes appeared as simple computing elements. These so-called neural networks were trained by a biologically inspired backpropagation algorithm⁵ that performed gradient descent on error-based cost functions. This line of work not only raised hopes of creating machines that were able to learn, but also understanding

the basic mechanisms behind biological learning. After a period of intense activity, however, history seemed to repeat itself with early enthusiasm overreaching actual accomplishments.

The study of support vector machines (SVMs) initiated a new approach to machine learning that focused more on statistical foundations⁷ and less on biological plausibility. SVMs are trained by minimizing a convex cost function that measures the margin of correct classification. This is the same margin that measures the mistake complexity in perceptrons, but in SVMs the optimization is justified by a rigorous statistical analysis. This approach to binary classification has proven effective in a wide range of applications from text classification ("Is this article related to my search query?") to bioinformatics ("Do these microarray profiles indicate cancerous cells?"). Indeed, a 1999 paper on SVMs by Thorsten Joachims¹ recently won the field's award for having the most significant and lasting impact.

SVMs were originally formulated for problems in binary classification where the goal is simply to distinguish objects in two different categories. In the following paper, the authors significantly extend the applicability of this approach to machine learning. The paper considers problems that cannot easily be reduced to simple classification—that is, where the goal is to predict a more complex object than a single binary outcome. Such problems arise in many applications of machine learning, including search engine ranking and part-of-speech tagging.

For a long time, researchers attempted to solve such problems by using simple-minded reductions to binary classification problems. These reductions failed to exploit any information about the structure of predicted objects. For example, conventional SVMs can be applied to the problem of part-of-speech tagging by considering each word of

a sentence based on the surrounding words. This approach ignores the fact that the tags of nearby words are mutually constraining—what is really needed is a self-consistent sequence of tags. The approach described here incorporates this type of structure through representations similar to those of probabilistic graphical models, such as Markov random fields.⁶ Moreover, the models in this approach are trained by optimizing measures of discriminative performance that are of most interest to practitioners.

The authors describe a principled framework for predicting structured objects with SVMs. They show that the required optimization involves an exponentially large number of constraints in the problem size. The combinatorial explosion reflects the large number of possible misclassifications when predicting structured objects. Remarkably, the authors show that despite the apparently unmanageable number of constraints, an ϵ -accurate solution can be found by a cutting-plane algorithm that only considers $O(1/\epsilon)$ constraints. The algorithm is described here, along with examples of real-world applications in information retrieval and protein sequence alignment. The impressive variety of applications considered in the paper is yet another reminder that the scope of machine learning expands with each new generation of researchers. □

References

1. Joachims, T. Transductive inference for text classification using support vector machines. In *Proceedings of the 16th International Conference on Machine Learning* (Bled, Slovenia, 1999), 200–209.
2. Minsky, M.L., and Papert, S.A. *Perceptrons*. MIT Press, Cambridge, MA, 1969.
3. Novikoff, A.B. On convergence proofs on perceptrons. *Symposium on the Mathematical Theory of Automata*. Polytechnic Institute of Brooklyn (1962), 615–622.
4. Rosenblatt, F. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review* 65, 6 (1958), 386–408.
5. Rumelhart, D.E., Hinton, G.E., and Williams, R.J. Learning representations by backpropagating errors. *Nature* 323 (1986), 533–536.
6. Taskar, B., Guestrin, C., and Koller, D. Maxmargin markov networks. *Advances in Neural Information Processing Systems*. S. Thrun, L. Saul, and B. Schölkopf, Eds. MIT Press, Cambridge, MA, 2004, 25–32.
7. Vapnik, V. *Statistical Learning Theory*. John Wiley, 1998.

John Shawe-Taylor is a professor at University College London, where he is director of the Centre for Computational Statistics and Machine Learning.

Predicting Structured Objects with Support Vector Machines

By Thorsten Joachims, Thomas Hofmann, Yisong Yue, and Chun-Nam Yu

Abstract

Machine Learning today offers a broad repertoire of methods for classification and regression. But what if we need to predict complex objects like trees, orderings, or alignments? Such problems arise naturally in natural language processing, search engines, and bioinformatics. The following explores a generalization of Support Vector Machines (SVMs) for such complex prediction problems.

1. INTRODUCTION

Consider the problem of natural language parsing illustrated in Figure 1 (left). A parser takes as input a natural language sentence, and the desired output is the parse tree decomposing the sentence into its constituents. While modern machine learning methods like Boosting, Bagging, and Support Vector Machines (SVMs) (see e.g., Hastie et al.⁹) have become the methods of choice for other problems in natural language processing (NLP) (e.g. word-sense disambiguation), parsing does not fit into the conventional framework of classification and regression. In parsing, the prediction is not a single yes/no, but a labeled tree. And breaking the tree down into a collection of yes/no predictions is far from straightforward, since it would require modeling a host of interdependencies between individual predictions. So, how can we take, say, an SVM and learn a rule for predicting trees?

Obviously, this question arises not only for learning to predict trees, but similarly for a variety of other structured and complex outputs. *Structured output prediction* is the name for such learning tasks, where one aims at learning a function $h: \mathcal{X} \rightarrow \mathcal{Y}$ mapping inputs $\mathbf{x} \in \mathcal{X}$ to complex and structured outputs $\mathbf{y} \in \mathcal{Y}$. In NLP, structured output prediction tasks range from predicting an equivalence relation in noun-phrase co-reference resolution (see Figure 1, right), to predicting an accurate and well-formed translation in machine translation. But problems of this type are also plentiful beyond NLP. For example, consider the problem of image segmentation (i.e., predicting an equivalence relation \mathbf{y} over a matrix of pixels \mathbf{x}), the problem of protein structure prediction (which we will phrase as an alignment problem in Section 3.2), or the problem of web search (i.e., predicting a diversified document ranking \mathbf{y} given a query \mathbf{x} as explored in Section 3.1). We will see in Section 3.3 that even binary classification becomes a structured prediction task when aiming to optimize multivariate performance measures like the F_1 -Score or *Precision@k*.

In this paper we describe a generalization of SVMs, called *Structural SVMs*,^{14, 26, 27} that can be used to address a large range of structured output prediction tasks. On the one hand, structural SVMs inherit the attractive properties of

regular SVMs, namely a convex training problem, flexibility in the choice of loss function, and the opportunity to learn nonlinear rules via kernels. On the other hand, structural SVMs inherit the expressive power of generative models (e.g., Probabilistic Context-Free Grammars [PCFG] or Markov Random Fields [MRF]). Most importantly, however, structural SVMs are a discriminative learning method that does not require the independence assumptions made by conventional generative methods. Similar to the increase in prediction accuracy one typically sees when switching from a naive Bayes classifier to a classification SVM (e.g., Joachims¹¹), structural SVMs promise such benefits also for structured prediction tasks (e.g., training PCFGs for parsing).

Structural SVMs follow and build upon a line of research on discriminative training for structured output prediction, including generalizations of Neural Nets,¹⁷ Logistic Regression¹⁶, and other conditional likelihood methods (e.g., McCallum et al.¹⁸). A particularly eye-opening paper is Lafferty et al.¹⁶ showing that global training for structured prediction can be formulated as a convex optimization problem. Our work follows this track. More closely, however, we build upon structural Perceptrons⁵ as well as methods for protein threading¹⁹ and extend these to a large-margin formulation with an efficient training algorithm. Independent of our work, Taskar et al.²⁵ arrived at a similar formulation, but with more restrictive conditions on the form of the learning task.

In the following, we explain how structural SVMs work and highlight three applications in search engine ranking, protein structure prediction, and binary classification under nonstandard performance measures.

2. STRUCTURAL SVMs

How can one approach structured output prediction? On an abstract level, a structured prediction task is much like a multiclass learning task. Each possible structure $\mathbf{y} \in \mathcal{Y}$ (e.g., parse tree) corresponds to one class (see Figure 2), and classifying a new example \mathbf{x} amounts to predicting its correct “class.” While the following derivation of structural SVMs starts from multiclass SVMs,⁶ there are four key problems that need to be overcome. All of these problems arise from the huge number $|\mathcal{Y}|$ of classes. In parsing, for example, the number of possible parse trees is exponential in the length of the sentence. And the situation is similar for most other structured output prediction problems.

A previous version of this paper—“Large Margin Methods for Structured and Interdependent Output Variables” by I. Tschantzidis, T. Joachims, T. Hofmann, and Y. Altun —was published in *J. Mach. Learn. Res.* (Sept. 2005).

Figure 1. Examples of structured output prediction tasks: predicting trees in natural language parsing (left), predicting the structure of proteins (middle), and predicting an equivalence relation over noun phrases (right).

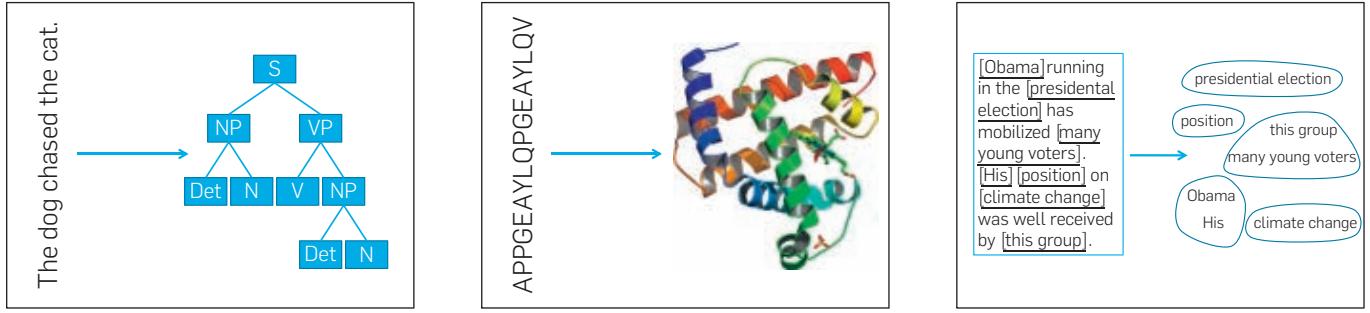
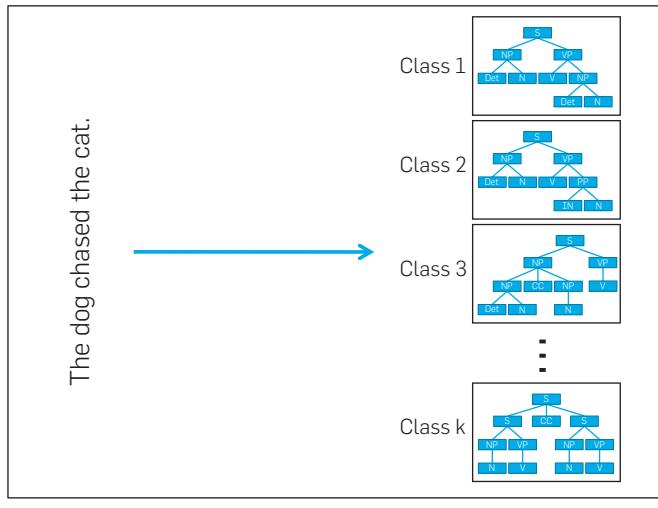


Figure 2. Structured output prediction as a multiclass problem.



The first problem is related to finding a compact representation for large output spaces. If we allowed even just one parameter for each class, we would already have more parameters than we could ever hope to have enough training data for. Second, just making a single prediction on a new example is a computationally challenging problem, since sequentially enumerating all possible classes may be infeasible. Third, we need a more refined notion of what constitutes a prediction error. Clearly, predicting a parse tree that is almost correct should be treated differently from predicting a tree that is completely wrong. And, last but not least, we need efficient training algorithms that have a run-time complexity sublinear in the number of classes.

In the following, we will tackle these problems one by one, starting with the formulation of the structural SVM method.

2.1. Problem 1: Structural SVM formulation

As mentioned above, we start the derivation of the structural SVM from the multiclass SVM.⁶ These multiclass SVMs use one weight vector \mathbf{w}_y for each class y . Each input x now has a score for each class y via $f(x, y) = \mathbf{w}_y \cdot \Phi(x)$. Here $\Phi(x)$ is a vector of binary or numeric features extracted from x . Thus, every feature will have an additively weighted influence in

the modeled compatibility between inputs x and classes y . To classify x , the prediction rule $h(x)$ then simply chooses the highest-scoring class

$$h(x) = \operatorname{argmax}_{y \in \mathcal{Y}} f(x, y) \quad (1)$$

as the predicted output. This will result in the correct prediction y for input x provided the weights $\mathbf{w} = (\mathbf{w}_1, \dots, \mathbf{w}_k)$ have been chosen such that the inequalities $f(x, \bar{y}) < f(x, y)$ hold for all incorrect outputs $\bar{y} \neq y$.

For a given training sample $(x_1, y_1), \dots, (x_n, y_n)$, this leads directly to a (hard-) margin formulation of the learning problem by requiring a fixed margin ($= 1$) separation of all training examples, while using the norm of \mathbf{w} as a regularizer:

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2, \text{ s.t. } f(x_i, y_i) - f(x_i, \bar{y}) \geq 1 \quad (\forall i, \bar{y} \neq y_i) \quad (2)$$

For a k -class problem, the optimization problem has a total of $n(k - 1)$ inequalities that are all linear in \mathbf{w} , since one can expand $f(x_i, y_i) - f(x_i, \bar{y}) = (\mathbf{w}_{y_i} - \mathbf{w}_{\bar{y}}) \cdot \Phi(x_i)$. Hence, it is a convex quadratic program.

The first challenge in using (2) for structured outputs is that, while there is generalization across inputs x , there is *no generalization across outputs*. This is due to having a separate weight vector \mathbf{w}_y for each class y . Furthermore, since the number of possible outputs can become very large (or infinite), naively reducing structured output prediction to multiclass classification leads to an undesirable blowup in the overall number of parameters.

The key idea in overcoming these problems is to extract features from input-output pairs using a so-called *joint feature map* $\Psi(x, y)$ instead of $\Phi(x)$. This yields compatibility functions with contributions from combined properties of inputs and outputs. These joint features will allow us to generalize across outputs and to define meaningful scores even for outputs that were never actually observed in the training data. At the same time, since we will define compatibility functions via $f(x, y) = \mathbf{w} \cdot \Psi(x, y)$, the number of parameters will simply equal the number of features extracted via Ψ , which may not depend on $|\mathcal{Y}|$. One can then use the formulation in (2) with the more flexible definition of f via Ψ to arrive at the following (hard-margin) optimization problem for structural SVMs.

$$\begin{aligned} \min_{\mathbf{w}} \quad & \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{s.t.} \quad & \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w} \cdot \Psi(\mathbf{x}_i, \bar{\mathbf{y}}) \geq 1 \quad (\forall i, \bar{\mathbf{y}} \neq \mathbf{y}_i) \end{aligned} \quad (3)$$

In words, find a weight vector \mathbf{w} of an input-output compatibility function f that is linear in some joint feature map Ψ so that on each training example it scores the correct output higher by a fixed margin than every alternative output, while having low complexity (i.e., small norm $\|\mathbf{w}\|$). Note that the number of linear constraints is still $n(|\mathcal{Y}| - 1)$, but we will suggest ways to cope with this efficiently in Section 2.4.

The design of the features Ψ is problem-specific, and it is the strength of the developed methods to allow for a great deal of flexibility in how to choose it. In the parsing example above, the features extracted via Ψ may consist of counts of how many times each production rule of the underlying grammar has been applied in the derivation described by a pair (\mathbf{x}, \mathbf{y}) . For other applications, the features can be derived from graphical models as proposed in Lafferty et al. and Taskar et al.,^{16,25} but more general features can be used as well.

2.2. Problem 2: Efficient prediction

Before even starting to address the efficiency of solving a quadratic programs of the form (3) for large n and $|\mathcal{Y}|$, we need to make sure that we can actually solve the inference problem of computing a prediction $h(\mathbf{x})$. Since the number of possible outputs $|\mathcal{Y}|$ may grow exponentially in the length of the representation of outputs, a brute-force exhaustive search over \mathcal{Y} may not always be feasible. In general, we require some sort of structural matching between the (given) compositional structure of the outputs \mathbf{y} and the (designed) joint feature map Ψ .

For instance, if we can decompose \mathcal{Y} into nonoverlapping independent parts, $\mathcal{Y} = \mathcal{Y}_1 \times \dots \times \mathcal{Y}_m$ and if the feature extraction Ψ respects this decomposition such that no feature combines properties across parts, then we can maximize the compatibility for each part separately and compose the full output by simple concatenation. A significantly more general case can be captured within the framework of Markov networks as explored by Lafferty et al. and Taskar et al.^{16,25} If we represent outputs as a vector of (random) variables $\mathbf{y} = (y_1, \dots, y_m)$, then for a fixed input \mathbf{x} , we can think of $\Psi(\mathbf{x}, \mathbf{y})$ as sufficient statistics in a conditional exponential model of $P(\mathbf{y}|\mathbf{x})$. Maximizing $f(\mathbf{x}, \mathbf{y})$ then corresponds to finding the most probable output, $\hat{\mathbf{y}} = \operatorname{argmax}_{\mathbf{y}} P(\mathbf{y}|\mathbf{x})$. Modeling the statistical dependencies between output variables via a dependency graph, one can use efficient inference methods such as the junction tree algorithm for prediction. This assumes the clique structure of the dependency graph induced by a particular choice of Ψ is tractable (e.g., the state space of the largest cliques remains sufficiently small). Other efficient algorithms, for instance, based on dynamic programming or minimal graph cuts, may be suitable for other prediction problems.

In our example of natural language parsing, the suggested feature map will lead to a compatibility function in which parse trees are scored by associating a weight with each production rule and by simply combining all weights additively. Therefore, a prediction can be computed efficiently using the CKY-Parsing algorithm (cf. Tsoucharidis et al.²⁷).

2.3. Problem 3: Inconsistent training data

So far we have tacitly assumed that the optimization problem in Equation 3 has a solution, i.e., there exists a weight vector that simultaneously fulfills all margin constraints. In practice this may not be the case, either because the training data is inconsistent or because our model class is not powerful enough. If we allow for mistakes, though, we must be able to quantify the degree of mismatch between a prediction and the correct output, since usually different incorrect predictions vary in quality. This is exactly the role played by a *loss function*, formally $\Delta: \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$, where $\Delta(\mathbf{y}, \bar{\mathbf{y}})$ is the loss (or cost) for predicting $\bar{\mathbf{y}}$, when the correct output is \mathbf{y} . Ideally we are interested in minimizing the expected loss of a structured output classifier, yet, as is common in machine learning, one often uses the empirical or training loss $\frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, h(\mathbf{x}_i))$ as a proxy for the (inaccessible) expected loss. Like the choice of Ψ , defining Δ is problem-specific. If we predict a set or sequence of labels, a natural choice for Δ may be the number of incorrect labels (a.k.a. Hamming loss). In the above parsing problem, a quality measure like F_1 may be the preferred way to quantify the similarity between two labeled trees in terms of common constituents (and then one may set $\Delta \equiv 1 - F_1$).

Now we will utilize the idea of loss functions to refine our initial problem formulation. Several approaches have been suggested in the literature, all of which are variations of the so-called soft-margin SVM: instead of strictly enforcing constraints, one allows for violations, yet penalizes them in the overall objective function. A convenient approach is to introduce slack variables that capture the actual violation,

$$f(\mathbf{x}_i, \mathbf{y}_i) - f(\mathbf{x}_i, \bar{\mathbf{y}}) \geq 1 - \xi_i \quad (\forall i, \bar{\mathbf{y}} \neq \mathbf{y}_i) \quad (4)$$

so that by choosing $\xi_i > 0$, a smaller (even negative) separation margin is possible. All that remains to be done then is to define a penalty for the margin violations. A popular choice is to use $\frac{1}{n} \sum_{i=1}^n \max_{\bar{\mathbf{y}} \neq \mathbf{y}_i} \{\Delta(\mathbf{y}_i, \bar{\mathbf{y}}) \xi_i\}$ thereby penalizing violations more heavily for constraints associated with outputs $\bar{\mathbf{y}}$ that have a high loss with regard to the observed \mathbf{y}_i . Technically, one can convert this back into a quadratic program as follows:

$$\begin{aligned} \min_{\mathbf{w}, \xi_i \geq 0} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{n} \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w} \cdot \Psi(\mathbf{x}_i, \bar{\mathbf{y}}) \geq 1 - \frac{\xi_i}{\Delta(\mathbf{y}_i, \bar{\mathbf{y}})} \quad (\forall i, \bar{\mathbf{y}} \neq \mathbf{y}_i) \end{aligned} \quad (5)$$

Here $C > 0$ is a constant that trades-off constraint violations with the geometric margin (effectively given by $1/\|\mathbf{w}\|$). This has been called the slack rescaling formulation. As an alternative, one can also rescale the margin to upper-bound the training loss as first suggested in Taskar et al.²⁵ and discussed in Tsoucharidis et al.²⁷ This leads to the following quadratic program

$$\begin{aligned} \min_{\mathbf{w}, \xi_i \geq 0} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{n} \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w} \cdot \Psi(\mathbf{x}_i, \bar{\mathbf{y}}) \geq \Delta(\mathbf{y}_i, \bar{\mathbf{y}}) - \xi_i \quad (\forall i, \bar{\mathbf{y}} \neq \mathbf{y}_i) \end{aligned} \quad (6)$$

Since it is slightly simpler, we will focus on this margin-rescaling formulation in the following.

2.4. Problem 4: Efficient training

Last, but not least, we need a training algorithm that finds the optimal \mathbf{w} solving the quadratic program in (6). Since there is a constraint for every incorrect label \bar{y} , we cannot enumerate all constraints and simply give the optimization problem to a standard QP solver. Instead, we propose to use the cutting-plane Algorithm 1 (or a similar algorithm for slack-rescaling). The key idea is to iteratively construct a working set of constraints \mathcal{W} that is equivalent to the full set of constraints in (6) up to a specified precision ε . Starting with an empty \mathcal{W} and $\mathbf{w} = \mathbf{0}$, Algorithm 1 iterates through the training examples. For each example, the argmax in Line 5 finds the most violated constraint of the quadratic program in (6). If this constraint is violated by more than ε (Line 6), it is added to the working set \mathcal{W} in Line 7 and a new \mathbf{w} is computed by solving the quadratic program over the new \mathcal{W} (Line 8). The algorithm stops and returns the current \mathbf{w} if \mathcal{W} did not change between iterations.

It is obvious that for any desired ε , the algorithm only terminates when it has found an ε -accurate solution, since it verifies in Line 5 that none of the constraints of the quadratic program in (6) is violated by more than ε . But how long does it take to terminate? It can be shown²⁷ that Algorithm 1 always terminates in a polynomial number of iterations that is independent of the cardinality of the output space \mathcal{Y} . In fact, a refined version of Algorithm 1^{13,14} always terminates after adding at most $O(C\varepsilon^{-1})$ constraints to \mathcal{W} (typically $|\mathcal{W}| \ll 1000$). Note that the number of constraints is not only independent of $|\mathcal{Y}|$, but also independent of the number of training examples n , which makes it an attractive training algorithm even for conventional SVMs.¹³

While the number of iterations is small, the argmax in Line 5 might be expensive to compute. In general, this is true, but note that this argmax is closely related to the argmax for computing a prediction $h(\mathbf{x})$. It is therefore called the “loss-augmented” inference problem, and often the prediction algorithm can be adapted to efficiently solve the loss-augmented inference problem as well.

Note that Algorithm 1 is rather generic and refers to the output structure only through a narrow interface. In particular, to apply the algorithm to a new structured prediction problem, one merely needs to supply implementations of $\Psi(\mathbf{x}, \mathbf{y})$, $\Delta(\mathbf{y}, \bar{\mathbf{y}})$, and $\text{argmax}_{\hat{\mathbf{y}} \in \mathcal{Y}} \{\Delta(\mathbf{y}, \hat{\mathbf{y}}) + \mathbf{w} \cdot \Psi(\mathbf{x}, \hat{\mathbf{y}})\}$. This

Algorithm 1 for training structural SVMs (margin-rescaling).

```

1: Input:  $S = ((\mathbf{x}_1, \mathbf{y}_1), \dots, (\mathbf{x}_n, \mathbf{y}_n))$ ,  $C, \varepsilon$ 
2:  $\mathcal{W} \leftarrow \emptyset$ ,  $\mathbf{w} = \mathbf{0}$ ,  $\xi_i \leftarrow 0$  for all  $i = 1, \dots, n$ 
3: repeat
4:   for  $i=1, \dots, n$  do
5:      $\hat{\mathbf{y}} \leftarrow \text{argmax}_{\hat{\mathbf{y}} \in \mathcal{Y}} \{\Delta(\mathbf{y}_i, \hat{\mathbf{y}}) + \mathbf{w} \cdot \Psi(\mathbf{x}_i, \hat{\mathbf{y}})\}$ 
6:     if  $\mathbf{w} \cdot [\Psi(\mathbf{x}_i, \mathbf{y}_i) - \Psi(\mathbf{x}_i, \hat{\mathbf{y}})] < \Delta(\mathbf{y}_i, \hat{\mathbf{y}}) - \xi_i - \varepsilon$  then
7:        $\mathcal{W} \leftarrow \mathcal{W} \cup \{\mathbf{w} \cdot [\Psi(\mathbf{x}_i, \mathbf{y}_i) - \Psi(\mathbf{x}_i, \hat{\mathbf{y}})] \geq \Delta(\mathbf{y}_i, \hat{\mathbf{y}}) - \xi_i\}$ 
8:        $(\mathbf{w}, \xi) \leftarrow \text{argmin}_{\substack{\mathbf{w} \\ w, \xi \geq 0}} \frac{1}{2} \mathbf{w} \cdot \mathbf{w} + \frac{C}{n} \sum_{i=1}^n \xi_i$  s.t.  $\mathcal{W}$ 
9:     end if
10:   end for
11: until  $\mathcal{W}$  has not changed during iteration
12: return( $\mathbf{w}, \xi$ )

```

makes the algorithm general and easily transferable to new applications. In particular, all applications discussed in the following used the general SVM^{struct} implementation of Algorithm 1 and supplied these three functions via an API.

2.5. Related approaches

The structural SVM method is closely related to other approaches, namely Conditional Random Fields¹⁶ and Maximum Margin Markov Networks (M3N).²⁵ The crucial link to probabilistic methods is to interpret the joint feature map as sufficient statistics in a conditional exponential family model. This means, we define a conditional probability of outputs given inputs by

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp [\mathbf{w} \cdot \Psi(\mathbf{x}, \mathbf{y})] \quad (7)$$

where $Z(\mathbf{x})$ is a normalization constant. Basically Equation (7) is a generalization of the well-known logistic regression where $\Psi(\mathbf{x}, \mathbf{y}) = \mathbf{y}\Phi(\mathbf{x})$ for $\mathbf{y} \in \{-1, 1\}$. The compatibility function used in structural SVMs directly governs this conditional probability. The probabilistic semantics lends itself to statistical inference techniques like penalized (conditional) maximum likelihood, where one maximizes $\sum_{i=1}^n \log P(\mathbf{y}_i | \mathbf{x}_i) - \lambda \|\mathbf{w}\|^2$. Unlike in Algorithm 1, however, here one usually must compute the expected sufficient statistics $\sum_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) \Psi(\mathbf{x}, \mathbf{y})$, for instance, in order to compute a gradient direction on the conditional log-likelihood. There are additional algorithmic challenges involved with learning CRFs (cf. Sha and Pereira²²) and one of the key advantages of structural SVMs is that it allows an efficient dual formulation. This enables the use of kernels instead of explicit feature representations as well as a sparse expansion of the solution (cf. Hofmann et al.¹⁰). A simpler, but usually less accurate learning algorithm that can be generalized to the structured output setting is the perceptron algorithm, as first suggested in Collins.⁵

The M3N approach is also taking the probabilistic view as its starting point, but instead of maximizing a likelihood-based criterion, aims at solving Equation 6. More specifically, M3Ns exploit the clique-based representation of $P(\mathbf{y}|\mathbf{x})$ over a dependency graph to efficiently reparametrize the dual problem of (6), effectively replacing the margin constraints over pairs of outputs by constraints involving functions of clique configurations. This is an interesting alternative to our cutting-plane algorithm, but has a somewhat more limited range of applicability.

3. APPLICATIONS

As outlined above, one needs to design and supply the following four functions when applying a structural SVM to a new problem:

$$\Psi(\mathbf{x}, \mathbf{y}) \quad (8)$$

$$\Delta(\mathbf{y}, \bar{\mathbf{y}}) \quad (9)$$

$$\text{argmax}_{\hat{\mathbf{y}} \in \mathcal{Y}} \{\mathbf{w} \cdot \Psi(\mathbf{x}, \hat{\mathbf{y}})\} \quad (10)$$

$$\text{argmax}_{\hat{\mathbf{y}} \in \mathcal{Y}} \{\Delta(\mathbf{y}, \hat{\mathbf{y}}) + \mathbf{w} \cdot \Psi(\mathbf{x}, \hat{\mathbf{y}})\} \quad (11)$$

The following three sections illustrate how this can be done for learning retrieval functions that provide diversified results, for aligning amino-acid sequences into

homologous protein structures, and for optimizing non-standard performance measures in binary classification.

3.1. Optimizing diversity in search engines

State of the art retrieval systems commonly use machine learning to optimize their ranking functions. While effective to some extent, conventional learning methods score each document independently and, therefore, cannot account for information diversity (e.g., not presenting multiple redundant results). Indeed, several recent studies in information retrieval emphasized the need for diversity (e.g., Zhai et al. and Swaminathan et al.^{24, 32}). In particular, they stressed modeling interdocument dependencies, which is fundamentally a structured prediction problem. Given a dataset of queries and documents labeled according to information diversity, we used structural SVMs to learn a general model of how to diversify.³¹

What is an example (\mathbf{x}, \mathbf{y}) in this learning problem? For some query, let $\mathbf{x} = \{x_1, \dots, x_n\}$ be the candidate documents that the system needs to rank. Our ground truth labeling for \mathbf{x} is a set of subtopics $\mathbf{T} = \{T_1, \dots, T_n\}$, where T_i denotes the subtopics covered by document x_i . The prediction goal is to find a subset $\mathbf{y} \subset \mathbf{x}$ of size K (e.g., $K = 10$ for search) maximizing subtopic coverage, and therefore maximizing the information presented to the user. We define our loss function $\Delta(\mathbf{T}, \mathbf{y})$ to be the (weighted) fraction of subtopics not covered by \mathbf{y} (more weight for popular subtopics).^a

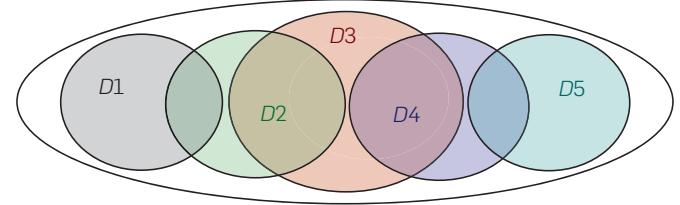
Even if the ground truth subtopics were known, computing the best \mathbf{y} can be difficult. Since documents overlap in subtopics (i.e., $\exists i, j : T_i \cap T_j \neq \emptyset$), this is a budgeted maximum coverage problem. The standard greedy method achieves a $(1 - 1/e)$ -approximation bound,¹⁵ and typically yields good solutions. The greedy method also naturally produces a ranking of the top documents.

Figure 3 depicts an abstract visualization of our prediction problem. The shaded regions represent candidate documents \mathbf{x} of a query, and the area covered by each region is the “information” (represented as subtopics \mathbf{T}) covered by that document. If \mathbf{T} was known, we could use a greedy method to find a solution with high subtopic diversity. For $K = 3$, the optimal solution in Figure 3 is $\mathbf{y} = \{D1, D3, D5\}$.

In general, however, the subtopics of new queries are unknown. One can think of subtopic labels as a manual partitioning of the total information of a query. We do, however, have access to an “automatic” representation of information diversity: the words contained in the documents. Intuitively, covering more (distinct) words should result in covering more subtopics. Since some words are more informative than others, a weighting scheme is required.

Following the approach of Essential Pages,²⁴ we measure word importance primarily using relative word frequencies within \mathbf{x} . For example, the word “computer” is generally informative, but conveys almost no information for the query “ACM” since it likely appears in most of the candidate documents. We learn a word weighting function using labeled training data, whereas Swaminathan et al.²⁴ uses a predefined function.

Figure 3. Different documents (shaded regions) cover different information (subtopics) of a query. Here, the set $\{D2, D3, D4\}$ contains the three documents that individually cover the most information, but $\{D1, D3, D5\}$ collectively covers more information.



We now present a simple example of the joint feature representation Ψ . Let $\phi(v, \mathbf{x})$ denote the feature vector describing the frequency of a word v amongst documents in \mathbf{x} . For example, we can construct $\phi(v, \mathbf{x})$ as

$$\phi(v, \mathbf{x}) = \begin{pmatrix} 1[v \text{ appears in at least } 15\% \text{ of } \mathbf{x}] \\ 1[v \text{ appears in at least } 35\% \text{ of } \mathbf{x}] \\ \dots \end{pmatrix}.$$

Let $V(\mathbf{y})$ denote the union of words contained in the documents of the predicted subset \mathbf{y} . We can write Ψ as

$$\Psi(\mathbf{x}, \mathbf{y}) = \sum_{v \in V(\mathbf{y})} \phi(v, \mathbf{x}).$$

Given a model vector \mathbf{w} , the benefit of covering word v in \mathbf{x} is $\mathbf{w} \cdot \phi(v, \mathbf{x})$, and is realized when a document in \mathbf{y} contains v , i.e., $v \in V(\mathbf{y})$. Because documents overlap in words, this is also a budgeted maximum coverage problem. Thus both prediction (Equation 10) and loss-augmented inference (Equation 11) can be solved effectively via the greedy method. Despite finding an approximate most violated constraint, we can still bound the precision of our solution.⁸ Practical applications require more sophisticated Ψ and we refer to Yue and Joachims³¹ for more details.

Experiments: We tested the effectiveness of our method using the TREC 6–8 Interactive Track queries. Relevant documents are labeled using subtopics. For example, query 392 asked human judges to identify applications of robotics in the world today, and they identified 36 subtopics among the results such as nanorobots and using robots for space missions. Additional details can be found in Yue and Joachims.³¹

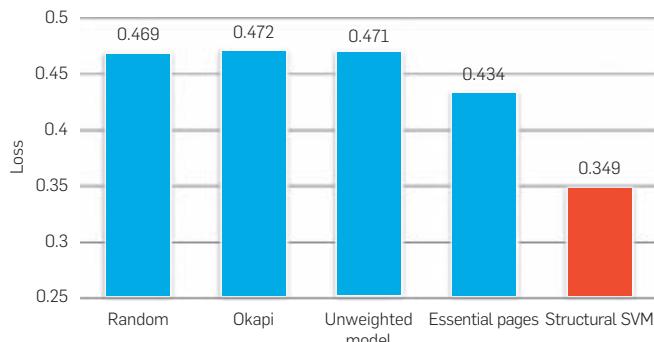
We compared against Okapi,²¹ Essential Pages,²⁴ random selection and unweighted word selection (all words have equal benefit). Okapi is a conventional retrieval function which does not account for diversity. Figure 4 shows the loss on the test set for retrieving $K = 5$ documents. The gains of the structural SVM over Essential Pages are 95% significant, and only the structural SVM and Essential Pages outperformed random.

This approach can accommodate other diversity criteria, such as diversity of clicks gathered from clickthrough logs. We can also accommodate very rich feature spaces based on, e.g., anchor text, URLs, and titles.

Abstractly, we are learning a mapping between two representations of the same set covering problem. One representation defines solution quality using manually

^a The loss function (Equation 9) can be defined using any ground truth format, not necessarily the same format as the output space.

Figure 4. Subtopic loss comparison when retrieving five documents. Structural SVM performance is superior with 95% confidence (using signed rank test).



labeled subtopics. The second representation learns a word weighting function, with goal of having the representations agree on the best solution. This setting is very general and can be applied to other domains beyond subtopic retrieval.

3.2. Predicting protein alignments

Proteins are sequences of 20 different amino acids joined together by peptide bonds. They fold into various stable structures under normal cell environments. A central question in biology is to understand how proteins fold, as their structures relate to their functions.

One of the more successful methods for predicting protein structure from an amino acid sequence is homology modeling. It approximates the structure of an unknown protein by comparing it against a database of proteins with experimentally determined structures. An important intermediate step is to align the unknown protein sequence with known protein sequences in the database, and this is a difficult problem when the sequences have diverged too far (e.g., less than 20% sequence identity).

To align protein sequences accurately for homology modeling, we need to have a good substitution cost matrix as input to the Smith–Waterman algorithm (a dynamic programming algorithm). A common approach is to learn the substitution matrix from a set of known good alignments between evolutionarily related proteins.

Structural SVMs are particularly suitable for learning the substitution matrix, since they allow incorporating all available information (e.g., secondary structures, solvent accessibility, and other physiochemical properties) in a principled way. When predicting the alignment $\mathbf{y} = (y_1, y_2, \dots)$ for two given protein sequences $\mathbf{x} = (s_a, s_b)$, each sequence location is described by a vector of features. The discriminative approach of structural SVMs makes it easy to incorporate these extra features without having to make unjustified independence assumptions as in generative models. As explained below, this enables learning a “richer” substitution function $w \cdot \Phi(\mathbf{x}, y_i)$ instead of a fixed substitution matrix.

The Smith–Waterman algorithm uses a linear function for scoring alignments, which allows us to decompose the

joint feature vector into a sum of feature vectors for individual alignment operations (match, insertion, or deletion):

$$w \cdot \Psi(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{\text{length}(\mathbf{y})} w \cdot \Phi(\mathbf{x}, y_i)$$

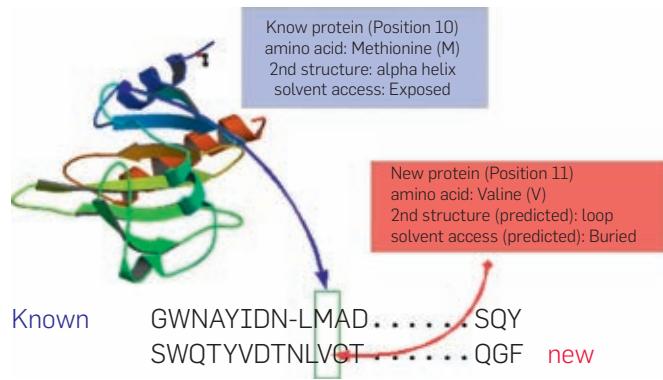
$\Phi(\mathbf{x}, y_i)$ is the feature vector for the i th alignment operation in the alignment \mathbf{y} of the sequence pair \mathbf{x} . Below we describe several alignment models represented by Φ , focusing on the scoring of matching two amino acids (see Figure 5 for an illustration of the features used):

- (i) *Simple*: we only consider the substitution cost of single features, e.g., the cost of matching a position with amino acid “M” with another position in a loop region.
- (ii) *Anova2*: we consider pairwise interaction of features, e.g., the cost of matching a position with amino acid “M” in an alpha helix with another position with amino acid “V” in a loop region.
- (iii) *Tensor*: we consider three-way interaction of features among amino acid, secondary structure, and solvent accessibility.
- (iv) *Window*: on top of three alignment models above, we add neighborhood features using a sliding window, which takes into account the hydrophobicity and secondary structure in the neighborhood of a position.
- (iv) *Profile*: on top of the alignment model *Window*, we add PSI-BLAST profile scores as features.

As the loss function $\Delta(\mathbf{y}, \bar{\mathbf{y}})$, we use Q_4 -loss. It is the percentage of matched amino acids in the correct alignment \mathbf{y} that are aligned more than four positions apart in the predicted alignment $\bar{\mathbf{y}}$. The linearity of the Q_4 -loss allows us to use the Smith–Waterman algorithm also for solving the loss-augmented inference problem during training. We refer to Yu et al.²⁹ for more details.

Experiments: We tested our algorithm with the training and validation sets developed in Qiu and Elber,²⁰ which contain 4542 and 4587 pairwise alignments of evolutionarily-related proteins with high structural similarities. For the test set we selected 4185 structures deposited in the Protein Data Bank from June 2005 to June 2006, leading to 29345 test pairs.

Figure 5. Aligning a new protein sequence with a known protein structure. Features at the aligned positions are used to construct substitution matrices under different alignment models.



For all sequence pairs x in the training, validation, and test sets both structures are known, and we use the CE structural alignment program²³ to generate “correct” alignments y .

The red bars in Figure 6 shows the Q_4 -scores (i.e., 100 minus Q_4 -loss) of the five alignment models described above. By carefully introducing features and considering their interactions, we can build highly complex alignment models (with hundreds of thousands of parameters) with very good alignment performance. Note that a conventional substitution matrix has only $20^2 = 400$ parameters.

SSALN²⁰ (blue bar) uses a generative alignment model for parameter estimation, and is trained using the same training set and features as the structural SVM algorithm. The structural SVM model *Window* substantially outperforms SSALN on Q_4 -score. Incorporating profile information makes the SVM model *Profile* perform even better, illustrating the benefit of being able to easily add additional features in discriminative training. The result from BLAST (green bar) is provided as a baseline.

To get a plausible upper bound for further improvements, we check in how far the CE alignments we used as ground truth agree with the structural alignments computed by TM-Align.³³ TM-Align gives a Q_4 -score of 85.45 when using the CE alignments as ground truth to compare against. This provides a reasonable upper bound on the alignment accuracy we might achieve on this noisy dataset. However, it also shows significant room for further research and improvement from our current alignment models.

3.3. Binary classification with unbalanced classes

Even binary classification problems can become structured output problems in some cases. Consider, for example, the case of learning a binary text classification rule with the classes “machine learning” and “other topics.” Like most text classification tasks, it is highly unbalanced, and we might only have, say, 1% machine-learning documents in our collection. In such a situation, prediction error typically becomes meaningless as a performance measure, since the default classifier that always predicts “other topics” already has a great error rate that is hard to beat. To overcome this problem, the

Figure 6. Q_4 -score of various Structural SVM alignment models compared to two baseline models. The Structural SVM using Window or Profile features significantly outperforms the SSALN baseline. The number in brackets is the number of features of the corresponding alignment model.



Information Retrieval (IR) community has designed other performance measures, like the F_1 -Score and the Precision/Recall Breakeven Point (PRBEP) (see e.g., Baeza-Yates and Ribeiro-Neto¹), that are meaningful even with unbalanced classes.

What does this mean for learning? Instead of optimizing some variant of error rate during training, which is what conventional SVMs and virtually all other learning algorithms do, it seems like a natural choice to have the learning algorithm directly optimize, for example, the F_1 -Score (i.e., the harmonic mean of precision and recall). This is the point where our binary classification task needs to become a structured output problem, since the F_1 -Score (as well as many other IR measures) is not a function of individual examples (like error rate), but a function of a set of examples. In particular, we arrive at the structured output problem of predicting an array of labels $\mathbf{y} = (y_1, \dots, y_n)$, $y_i \in \{-1, +1\}$, for an array of feature vectors $\mathbf{x} = (x_1, \dots, x_n)$, $x_i \in \mathcal{R}^N$. Each possible array of labels $\bar{\mathbf{y}}$ now has an associated F_1 -Score $F_1(\mathbf{y}, \bar{\mathbf{y}})$ w.r.t. the true labeling \mathbf{y} , and optimizing F_1 -Score on the training set becomes a well-defined problem.

The problem of predicting an array of binary labels \mathbf{y} for an array of input vectors \mathbf{x} optimizing a loss function $\Delta(\mathbf{y}, \bar{\mathbf{y}})$ fits neatly into the structural SVM framework. Again, we only need to define $\Psi(\mathbf{x}, \mathbf{y})$ and $\Delta(\mathbf{y}, \bar{\mathbf{y}})$ appropriately, and then find algorithms for the two argmax problems. The choice of loss function is obvious: when optimizing F_1 -Score, which ranges from 0 (worst) to 1 (best), we will use

$$\Delta(\mathbf{y}, \bar{\mathbf{y}}) = 1 - F_1(\mathbf{y}, \bar{\mathbf{y}}).$$

For the joint feature mapping, using

$$\Psi(\mathbf{x}, \mathbf{y}) = \frac{1}{n} \sum_{i=1}^n y_i x_i$$

can be shown to be a natural choice, since it makes the structural SVM equivalent to a conventional SVM if error rate is used as the loss $\Delta(\mathbf{y}, \bar{\mathbf{y}})$. It also makes computing a prediction very efficient with

$$\mathbf{h}(\mathbf{x}) = \text{argmax}_{\mathbf{y}} \{ \mathbf{w} \cdot \Psi(\mathbf{x}, \mathbf{y}) \} = (\text{sign}(\mathbf{w} \cdot \mathbf{x}_1), \dots, \text{sign}(\mathbf{w} \cdot \mathbf{x}_n)).$$

Computing the loss-augmented argmax necessary for training is a bit more complicated and depends on the particular loss function used, but it can be done in time at most $O(n^2)$ for any loss function that can be computed from the contingency table of prediction errors.¹² SVM^{perf} is an implementation of the method for various performance measures.

Experiments: Table 1 shows the prediction performance of the structural SVM on four benchmark datasets, described in more detail in Joachims.¹² In particular, it compares the structural SVM that directly optimizes the measure it is evaluated on (here F_1 , PRBEP, and ROCArea) to a conventional SVM that accounts for unbalanced classes with a linear cost model. For both methods, parameters were selected to optimize the evaluation measure via cross validation. In most cases, the structural SVM outperforms the conventional SVM, and it never does substantially worse.

Beyond these performance gains, the structural SVM approach has an attractive simplicity to it—direct optimization

Table 1. Comparing a conventional SVM to a structural SVM that directly optimizes the performance measure.

Dataset	Method	F_1	PRBEP	ROCArea
REUTERS	STRUCT SVM	62.0	68.2	99.1
	SVM	56.1	65.7	98.6
ARXIV	STRUCT SVM	56.8	58.4	92.8
	SVM	49.6	57.9	92.7
OPTDIGITS	STRUCT SVM	92.5	92.7	99.4
	SVM	91.5	91.5	99.4
COVERTYPE	STRUCT SVM	73.8	72.1	94.6
	SVM	73.9	71.0	94.1

of the desired performance measure instead of using proxy measures during training (e.g., different linear cost models). However, there are still several open question before making this process fully automatic; for example, understanding the interaction between Δ and Ψ as recently explored in Chakrabarti et al. and Chapelle et al.^{3,4}

4. CONCLUSION AND OUTLOOK

In summary, structural SVMs are flexible and efficient methods for structured output prediction with a wide range of potential applications, most of which are completely or largely unexplored. Other explored applications include hierarchical classification,² clustering,⁷ and optimizing average precision.³⁰ Due to the universality of the cutting-plane training algorithm, only relatively small API changes are required for any new application. *SVM^{struct}* is an implementation of the algorithm with APIs in C and Python, and it is available at svmlight.joachims.org.

Beyond applications, there are several areas for research in further developing the method. One such area is training structural SVMs with Kernels. While the cutting-plane method can be extended to use Kernels, it becomes quite slow and more efficient methods are needed.²⁸ Another area results from that fact that solving the two argmax problems exactly is intractable for many application problems. However, for many such problems there exist methods that compute approximate solutions. An interesting question is how the approximation quality affects the quality of the structural SVM solution.⁸

This work was supported in part through NSF Awards IIS-0412894 and IIS-0713483, NIH Grants IS10RR020889 and GM67823, a gift from Yahoo!, and by Google. □

6. Crammer, K., Singer, Y. On the algorithmic implementation of multiclass kernel-based vector machines. *J. Mach. Learn. Res. (JMLR)* 2 (2001), 265–292.
7. Finley, T., Joachims, T. Supervised clustering with support vector machines. In *International Conference on Machine Learning (ICML)* (2005), 217–224.
8. Finley, T., Joachims, T. Training structural SVMs when exact inference is intractable. In *International Conference on Machine Learning (ICML)* (2008), 304–311.
9. Hastie, T., Tibshirani, R., Friedman, J. *The Elements of Statistical Learning*. Springer (2001).
10. Hofmann, T., Schölkopf, B., Smola A.J. Kernel methods in machine learning. *Ann. Stat.* 36, 3 (2008), 1171–1220.
11. Joachims, T. *Learning to Classify Text Using Support Vector Machines – Methods, Theory, and Algorithms*. Kluwer/Springer (2002).
12. Joachims, T. A support vector method for multivariate performance measures. In *International Conference on Machine Learning (ICML)* (2005), 377–384.
13. Joachims, T. Training linear SVMs in linear time. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)* (2006), 217–226.
14. Joachims, T., Finley, T., Yu, C.-N. Cutting-plane training of structural svms. *Machine Learning Journal* (2009) DOI 10.1007/S10994-009-5108-8
15. Khuller, S., Moss, A., Naor, J. The budgeted maximum coverage problem. *Inform. Process. Lett.* 70, 1 (1997), 39–45.
16. Lafferty, J., McCallum, A., Pereira, F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)* (2001).
17. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324, November.
18. McCallum, A., Freitag, D., Pereira, F. Maximum entropy Markov models for information extraction and segmentation. In *International Conference on Machine Learning (ICML)* (2000), 591–598.
19. Meller, J., Elber, R. Linear programming optimization and a double statistical filter for protein threading protocols. *Proteins Struct. Funct. Genet.* 45 (2001), 241–261.
20. Qiu, J., Elber, R. SSALN: an alignment algorithm using structure-dependent substitution matrices and gap penalties learned from structurally aligned protein pairs. *Proteins* 62 (2006), 881–91.
21. Robertson, S., Walker, S., Jones, S., Hancock-Beaulieu, M., Gatford, M. Okapi at TREC-3. In *Proceedings of TREC-3* (1994).
22. Sha, F., Pereira, F. Shallow parsing with conditional random fields. In *NAACL'03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology* (Morristown, NJ, USA, 2003), 134–141. Association for Computational Linguistics.
23. Shindyalov, I.N., Bourne, P.E. Protein structure alignment by incremental combinatorial extension(CE) of the optimal path. *Protein Eng.* 11 (1998), 739–747.
24. Swaminathan, A., Mathew, C., Kirovski, D. Essential pages. Technical Report MSR-TR-2008-015, Microsoft Research (2008).
25. Taskar, B., Guestrin, C., Koller, D. Maximum-margin markov networks. In *Advances in Neural Information Processing Systems (NIPS)* (2003).
26. Tschantaridis, I., Hofmann, T., Joachims, T., Altun, Y. Support vector machine learning for interdependent and structured output spaces. In *International Conference on Machine Learning (ICML)* (2004), 104–112.
27. Tschantaridis, I., Joachims, T., Hofmann, T., Altun, Y. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res. (JMLR)*, 6 (September 2005), 1453–1484.
28. Yu, C.-N.J., Joachims, T. Training structural svms with kernels using sampled cuts. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)* (2008), 794–802.
29. Yu, C.-N.J., Joachims, T., Elber, R., Pillardy, J. Support vector training of protein alignment models. *J. Comput. Biol.* 15, 7 (September 2008), 867–880.
30. Yue, Y., Finley, T., Radlinski, F., Joachims, T. A support vector method for optimizing average precision. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)* (2007), 271–278.
31. Yue, Y., Joachims, T. Predicting diverse subsets using structural SVMs. In *International Conference on Machine Learning (ICML)* (2008), 271–278.
32. Zhai, C., Cohen, W.W., Lafferty, J. Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval (SIGIR)* (2003).
33. Zhang, Y., Skolnick, J. TM-align: A protein structure alignment algorithm based on TM-score. *Nucleic Acids Res.* 33 (2005), 2302–2309.

References

1. Baeza-Yates, R., Ribeiro-Neto, B. *Modern Information Retrieval*. Addison-Wesley-Longman, Harlow, UK (May 1999).
2. Cai, L., Hofmann, T. Hierarchical document categorization with support vector machines. In *Proceedings of the ACM Conference on Information and Knowledge Management (CIKM)* (2004).
3. Chakrabarti, S., Khanna, R., Sawant, U., Bhattacharyya, C. Structured learning for non-smooth ranking losses. In *ACM Conference on Knowledge Discovery and Data Mining (KDD)* (2008).
4. Chapelle, O., Le,Q., Smola, A. Large margin optimization of ranking measures. In *NIPS Workshop on Machine Learning for Web Search* (2007).
5. Collins, M. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Empirical Methods in Natural Language Processing (EMNLP)* (2002), 1–8.

Thorsten Joachims
Department of Computer Science, Cornell University, Ithaca, NY.

Thomas Hofmann
Google Inc., Zürich, Switzerland.

Yisong Yue,
Department of Computer Science, Cornell University, Ithaca, NY.

Chun-Nam Yu
Department of Computer Science Cornell University, Ithaca, NY.



acmqueue has now moved completely online!

acmqueue is guided and written by distinguished and widely known industry experts. The newly expanded site also offers more content and unique features such as *planetqueue* blogs by *queue* authors who "unlock" important content from the ACM Digital Library and provide commentary; **videos**; downloadable **audio**; **roundtable discussions**; plus unique *acmqueue* **case studies**.

acmqueue provides a critical perspective on current and emerging technologies by bridging the worlds of journalism and peer review journals. Its distinguished Editorial Board of experts makes sure that *acmqueue*'s high quality content dives deep into the technical challenges and critical questions software engineers should be thinking about.

The screenshot shows the acmqueue homepage with a navigation bar at the top for HOME, AUDIOCASTS, VIDEOS, BLOG, and PLANET QUEUE. On the left, there's a sidebar with links for ARCHIVE, COLUMNS, DISCUSSIONS, BROWSE TOPICS, and OTHER ACM LINKS. The main content area features a "Welcome to acmqueue" section with a brief introduction, followed by "Latest Queue Content" which includes several blog posts with titles like "Securing in the Browser" and "Cybermen? Where the Cloud Turns Gray". To the right, there's a "QUEUE AUTHOR BLOG ROLL" with thumbnails and names of authors like James Bonding, Jeff Barr, Bruce Schneier, Gregor Letzky, Cary Disten, Tom Blagoev, David Ascher, and James Hamilton.

Visit today!

<http://queue.acm.org/>

CAREERS

Carleton College

Computer Science Department

Tenure-Track Faculty Position

Carleton College invites applications for a tenure-track position in computer science, in any area of specialization, to begin September 1, 2010. More details on this position can be found at

cs.carleton.edu/hiring.

Carleton is a highly selective liberal arts college with outstanding, enthusiastic students. We seek an equally enthusiastic computer scientist committed to excellence in teaching, curriculum design, ongoing research, and undergraduate research advising. To apply, send cover letter, CV, brief statements describing teaching philosophy and research agenda, graduate transcript, and three reference letters, at least one of which addresses your teaching, to cssearch@carleton.edu. Electronic submissions (pdf) are preferred, or write to: Tenure-Track Search Committee, Computer Science Department, Carleton College, One North College Street, Northfield, MN 55057. Applications completed by January 1, 2010 will receive full consideration.

Carleton College is an affirmative action/equal opportunity employer. We are committed to developing our faculty to better reflect the diversity of our student body and American society. Women and members of minority groups are strongly encouraged to apply.

Duke University

Department of Computer Science

The Department of Computer Science at Duke University invites applications and nominations for faculty positions at an assistant professor level, to begin August 2010. We are interested in strong candidates in all active research areas of computer science, both core and interdisciplinary areas, including algorithms, artificial intelligence, computational economics, computer architecture, computer vision, database systems, distributed systems, machine learning, networking, security, and theory.

The department is committed to increasing the diversity of its faculty, and we strongly encourage applications from women and minority candidates.

A successful candidate must have a solid disciplinary foundation and demonstrate promise of outstanding scholarship in every respect, including research and teaching. Please refer to www.cs.duke.edu for information about the department and to www.provost.duke.edu/faculty/ for information about the advantages that Duke offers to faculty.

Applications should be submitted online at www.cs.duke.edu/facsearch. A Ph.D. in computer science or related area is required. To guarantee full consideration, applications and letters of reference should be received by January 3, 2010.

Durham, Chapel Hill, and the Research Triangle of North Carolina are vibrant, diverse, and

thriving communities, frequently ranked among the best places in the country to live and work. Duke and the many other universities in the area offer a wealth of education and employment opportunities for spouses and families.

Duke University is an affirmative action, equal opportunity employer.

Illinois Institute of Technology

Department of Computer Science

Applications are invited for a tenure-track assistant professor position in Computer Science beginning Fall 2010. Highly qualified candidates from all areas of computing will be considered. Applicants from experimental and systems areas are especially encouraged. Excellence in research, teaching and obtaining external funding is expected.

The Department offers B.S., M.S., and Ph.D. degrees in Computer Science and has research strengths in distributed systems, information retrieval, computer networking, intelligent information systems and algorithms. The Department has strong connections to Fermilab and Argonne National Laboratories and local industry.

IIT, located within 10 minutes of downtown Chicago, has entered an exciting new era of prominence in science, technology, and engineering and has hired several top administrators to help build on the university's reputation, excellence in research and education, and diversity. IIT is an Equal Opportunity Affirmative Action Employer.

Evaluation will start December 1, 2009 and continue until position is filled. Applicants should send a detailed curriculum vita, a statement of research and teaching interests, and the names and email addresses of at least four references to:

Computer Science Faculty Search Committee
Department of Computer Science
Illinois Institute of Technology
10 W. 31st Street
Chicago, IL 60616
Phone: 312-567-5152
Email: search@cs.iit.edu
<http://www.iit.edu/cs1/cs>

Max Planck Institute for Software Systems (MPI-SWS)

Tenure-track openings

Applications are invited for tenure-track and tenured faculty positions in all areas related to the design, analysis and engineering of software systems, including programming languages, formal methods, security, distributed, networked and embedded systems, databases and information systems, and human-computer interaction. A doctoral degree in computer science or related areas and an outstanding research record are required. Successful candidates are expected to build a team and pursue a highly visible research

agenda, both independently and in collaboration with other groups. Senior candidates must have demonstrated leadership abilities and recognized international stature.

MPI-SWS, founded in 2005, is part of a network of eighty Max Planck Institutes, Germany's premier basic research facilities. MPIs have an established record of world-class, foundational research in the fields of medicine, biology, chemistry, physics, technology and humanities. Since 1948, MPI researchers have won 17 Nobel prizes. MPI-SWS aspires to meet the highest standards of excellence and international recognition with its research in software systems.

To this end, the institute offers a unique environment that combines the best aspects of a university department and a research laboratory:

- a) Faculty receive generous base funding to build and lead a team of graduate students and postdocs. They have full academic freedom and publish their research results freely.
- b) Faculty have the opportunity to supervise doctoral theses, teach graduate and undergraduate courses, and have the flexibility to incorporate teaching into their research agenda.
- c) Faculty are provided with outstanding technical and administrative support facilities as well as internationally competitive compensation packages.

Funds have been committed to grow the institute to a strength of 17 tenured and tenure-track faculty, and about 100 doctoral and post-doctoral positions. Additional growth through outside funding is expected. We maintain an open, international and diverse work environment and seek applications from outstanding researchers regardless of national origin or citizenship. The working language is English; knowledge of the German language is not required for a successful career at the institute.

The institute is located in Kaiserslautern and Saarbruecken, in the tri-border area of Germany, France and Luxembourg. The area offers a high standard of living, beautiful surroundings and easy access to major metropolitan areas in the center of Europe, as well as a stimulating, competitive and collaborative work environment. In immediate proximity are the MPI for Informatics, Saarland University, the Technical University of Kaiserslautern, the German Center for Artificial Intelligence (DFKI), and the Fraunhofer Institutes for Experimental Software Engineering and for Industrial Mathematics.

Qualified candidates should apply online at <http://www.mpi-sws.org/application>. The review of applications will begin on January 4, 2010, and applicants are strongly encouraged to apply by that date; however, applications will continue to be accepted through January 2010.

The institute is committed to increasing the representation of minorities, women and individuals with physical disabilities in Computer Science. We particularly encourage such individuals to apply.

Montclair State University

Assistant/Associate Professor

The Department of Computer Science invites applications for a tenure track position in Information Technology. The Department's 14 faculty members support the BS in Computer Science with an ABET CAC accredited track, the BS in Information Technology, the BS/MS and MS in Computer Science. The position's teaching focus is information technology major courses. Additional teaching includes a variety of computer science courses at all levels to ethnically diverse students. The position requires the development and maintenance of an active research program with deep student involvement. The ability to work as a member of interdisciplinary teams as the Department develops and modifies computing undergraduate and graduate programs with a planned doctoral program in computational science is essential.

Candidates must have a Ph.D. in Computer Science or a related discipline before August 15, 2010. Candidates must have good communication skills. All faculty members are expected to have an ongoing research program, to actively foster student learning, to be involved in professional activities, and to develop external funding support for their scholarship.

Salary and range is dependent on qualifications.

Starting date: September 1, 2010

Send hardcopy that includes cover letter, C.V., at least three professional references, statement of research interests, teaching philosophy with experience, and professional goals to:

Search Committee - V - F23
Department of Computer Science
1 Normal Avenue
Montclair State University
Montclair, NJ 07043

Screening begins immediately and continues until the position is filled.

Montclair State University is located 14 miles west of New York City on a beautiful 200-acre suburban campus. Additional information can be found on the MSU website at <http://www.montclair.edu>

Rensselaer Polytechnic Institute

Tenured/Tenured Track Faculty Games Simulation Arts and Science

The School of Humanities, Arts and Social Sciences at Rensselaer Polytechnic Institute in Troy, NY seeks to hire three faculty for its Games and Simulation Arts and Science (GSAS) degree program. Rank for these tenured and tenure-track positions is open. Duties include teaching in the area of GSAS; relevant research or other visible work; and service to the department, Institute and profession.

Department affiliation may be with Art, Cog Sci, Communications, Economics, Science and Technology Studies, or Computer Science. We are seeking faculty with broad visions who may not fit into the traditional departmental structure.

Tenured and Tenure Track Faculty Openings Qualifications:

An appropriate terminal degree (MS, MFA, or PhD or foreign equivalent), professional activity and visibility in the game industry and/or history of academic research and teaching in areas related to Games and Simulations broadly defined. Degree

must be conferred by the start of the appointment. Related disciplines could be Fine Arts, Cognitive Science, Communications, Economics, Science and Technology Studies, or Computer Science.

How to Apply: Send a resume and cover letter describing your professional interests and qualifications, portfolio or selected research publications, and three letters of recommendation. Work samples may be in the form of DVDs, CDs, websites, books, articles, or other appropriate media (for return please include a SASE). Screening will begin immediately and will continue until the positions are filled.

JoAnn Drost - HASS
Rensselaer Polytechnic Institute
Sage Laboratory
110 8th Street
Troy, NY 12180-3590
email: drostj@rpi.edu

St. Norbert College

Assistant Professor

Tenure-track faculty position in Computer Science department starting August 2010. Responsibilities include teaching core courses and leading student research and projects. SNC, a liberal arts college, seeks candidates committed to excellence in teaching who will contribute to their profession and to the mission of the College. PhD in CS or related field required. Details at www.snc.edu/hr/positions.

Swarthmore College

Computer Science Department

Applications are invited for a two-year Visiting Assistant Professor position beginning August 2010.

Swarthmore College is a small, selective liberal arts college located in a suburb of Philadelphia. The Computer Science Department offers majors and minors in computer science at the undergraduate level. Applicants must have teaching experience and should be comfortable teaching a wide range of courses at the introductory and intermediate level. We are particularly interested in candidates who specialize in theory and algorithms or in systems areas, however, we will consider candidates from all areas of CS. A Ph.D. in CS by or near the time of appointment is preferred (ABD is required). We expect to begin interviewing in early February 2010.

See <http://cs.swarthmore.edu/jobs> for application submission information and more details about the position. Swarthmore College is an equal opportunity employer. Applications from women and members of minority groups are encouraged. Applications will be accepted until the position is filled.

The Hong Kong Polytechnic University

Department of Computing

The Department invites applications for Professors in Database and Information Systems / AI and Knowledge Engineering / Computer System and Theory (Algorithms, OS, Computer Language, etc). The appointees will be required to provide leadership in all aspects of academic activities, develop established or new research areas in the Department, take responsibility for

the development of teaching programmes, and strengthen the international network of the Department and the University. Applicants should have a PhD degree in Computer Science and be conversant in other related disciplines, outstanding abilities with good administrative experience as an academic leader, and excellent track record in research and high quality publications. Please visit the website at <http://www.comp.polyu.edu.hk> for more information about the Department. Salary offered will be commensurate with qualifications and experience. Initial appointments will be made on a fixed-term gratuity-bearing contract. Re-engagement thereafter is subject to mutual agreement. Remuneration package will be highly competitive. Applicants should state their current and expected salary in the application. Please submit your application via email to hrstaff@polyu.edu.hk. Application forms can be downloaded from <http://www.polyu.edu.hk/hro/job.htm>.

Deadline for application is 17 February 2010. Details of the University's Personal Information Collection Statement for recruitment can be found at <http://www.polyu.edu.hk/hro/jobpics.htm>.

The Hong Kong University of Science and Technology

Department of Computer Science and Engineering Faculty Positions

The Department of Computer Science and Engineering is one of the largest departments in the School of Engineering. The Department currently has 40 faculty members recruited from major universities and research institutions around the world, with about 1000 students (including 600 undergraduate and 170 postgraduate students). The medium of instruction is English. More information on the Department can be found at <http://www.cse.ust.hk>.

The Department will have at least two tenure-track faculty openings at Assistant Professor/Associate Professor/Professor levels for the 2010-2011 academic year. We are looking for faculty candidates with interests in multidisciplinary research areas related to computational science and engineering such as bioinformatics and financial engineering. Strong candidates in core computer science and engineering research areas will also be considered. Applicants at Assistant Professor level should have an earned PhD degree and demonstrated potential in teaching and research.

Salary is highly competitive and will be commensurate with qualifications and experience. Fringe benefits include medical/dental benefits and annual leave. Housing will also be provided where applicable. For appointment at Assistant Professor/Associate Professor level, initial appointment will normally be on a three-year contract, renewable subject to mutual agreement. A gratuity will be payable upon satisfactory completion of contract.

Applications should be sent through e-mail including a cover letter, curriculum vitae (including the names and contact information of at least three referees), a research statement and a teaching statement (all in PDF format) to csrecruit@cse.ust.hk. Priority will be given to applications received by **28 February 2010**. Applicants will be promptly acknowledged through e-mail upon receiving the electronic application material.

(Information provided by applicants will be used for recruitment and other employment-related purposes.)

University at Buffalo, The State University of New York

Faculty Position in Computer Science and Engineering

The CSE Department invites excellent candidates in all core areas of Computer science and Engineering, especially experimental and systems areas, to apply for an opening at the assistant professor level.

The department is affiliated with successful centers devoted to biometrics, bioinformatics, biomedical computing, cognitive science, document analysis and recognition, high performance computing, and information assurance.

Candidates are expected to have a Ph.D. in Computer Science/Engineering or related field by August 2010, with an excellent publication record and potential for developing a strong funded research program.

Applications should be submitted by December 31, 2009 electronically via <http://recruit.cse.buffalo.edu/>

The University at Buffalo is an Equal Opportunity Employer/Recruiter.

University of Hartford (West Hartford, Connecticut)

Assistant Professor

The Departments of Computer Science and Multimedia Web Design and Development at the University of Hartford (West Hartford, Connecticut) seek applications for a joint assistant professor tenure-track position. Candidates must have an earned doctorate (ABD will be considered) in a field that addresses theoretical and applied issues in computer science and information technology. Review of applications will begin November 1, 2009. For more information, please visit <http://www.cs.hartford.edu/employment.pdf>

University of Houston

Department of Computer Science Chair

The Department of Computer Science at the University of Houston (www.cs.uh.edu) is looking for a new Department Chair. The department is a highly dynamic place on an ascending trajectory with 256 undergraduates, 310 graduate students out of which 88 are PhD students. The University of Houston is located in one of the most vibrant metropolitan areas in the nation. Currently, the Department has 22 tenure-track faculty members, slated to expand to 30 faculty members within the next four years. In FY2009, the department received over \$6 M in competitive research funding from federal, state, and corporate sources. All our recently recruited faculty members have federal support for their research, and three are recipients of the prestigious CAREER Award from the National Science Foundation.

The Department has strong research programs in Computer Systems (high performance computing, networks, real-time systems, security), Data Analysis (information retrieval, data mining, machine learn-

ing) and Computational Life Sciences (biomedical image analysis, bioinformatics, biometrics, graphics). The Department's research is the epitome of innovation, mixing advances in core computer science areas with pace-setting multi-disciplinary programs in computational medicine, biology, and psychology. The combination of fundamental research and innovations has led to numerous local, national and international collaborations, the strongest of which are with the Texas Medical Center.

The University of Houston, one of the largest in the nation with over 36,000 students, is located in one of the most vibrant metropolitan areas. Houston, the 4th largest U.S. city, is the epicenter of the energy industry, features the largest medical center in the world, and hosts the Johnson Space Center. The Department's research laboratories have joint programs with laboratories from the local medical schools and hospitals, NASA, and the high-tech industry.

The ideal Department Chair candidate should be an established leader in his/her field and widely known in the computer science community and beyond. S/he should have proven managerial and marketing skills running another department or a major lab. S/he should also have excellent people skills and be privy to the academic, fund-raising, and publicity system's inner workings in the United States. The Chair's designated mission would be to further accelerate the department's ascendancy to top ranking positions.

Qualified applicants need to apply on-line at <http://www.cs.uh.edu/chair-search>. A CV, at least six recommendation letters, and a vision statement are required. In the vision statement the applicants should clearly describe their vision for the growth

of the department and how their track record will support the University mission. The deadline for submission of all documents (including recommendation letters) is January 31, 2010. However, screening of applications and interviews will be ongoing and applicants are encouraged to apply as soon as possible. Interested applicants may further inquire with the Chair of the Search Committee, Prof. Pavlidis at ipavlidis@uh.edu, 713-743-0101.

The University of Houston is an Equal Opportunity/Affirmative Action institution. Minorities, women, veterans and persons with disabilities are encouraged to apply.

University of Notre Dame

The Department of Computer Science and Engineering

Assistant or Associate Professor

The Department of Computer Science and Engineering at the University of Notre Dame invites applications for positions at the rank of Assistant or Associate Professor. Exceptional candidates in all areas of specialization will be considered, and the area of bioinformatics is an especially high priority for us this year.

The Department offers a PhD degree as well as accredited undergraduate Computer Science and Computer Engineering degrees. There are approximately seventy-five students in the PhD program and over one hundred majors in the undergraduate programs. Our faculty are engaged in cutting-edge and highly visible research in algorithms, bioinformatics and computational biology, computer architecture and nanotechnology, computer security, data mining /



VCU
V i r g i n i a C o m m o n w e a l t h
U n i v e r s i t y

FACULTY POSITIONS

School of Engineering

Computer Science

The Computer Science Department at Virginia Commonwealth University (VCU) invites applications for two tenure-track positions at the rank of Assistant/Associate Professor level. Candidates must have a Ph.D. and demonstrate the ability to establish, or continue an active, externally funded research program. All specializations will be considered with particular interest in parallel/cloud computing, information security, networks and software engineering. The CS Department's research is strongly connected to VCU's Schools of Medicine and School of the Life Sciences with applications in medical informatics and bioinformatics. VCU, the largest university in Virginia, is an urban Carnegie Research I Extensive institution ranked in the top 100 universities in federal R&D expenditures, with a richly diverse community and commitment to multicultural opportunities.

Review of applications will begin on November 1, 2009 and will continue until the position is filled. Candidates are to submit applications electronically to: cnssearch@vcu.edu as a single pdf file that includes (in order) a cover letter, resume, teaching statement, research statement, and the names and e-mail addresses of three references.

*Virginia Commonwealth University is an Equal Opportunity/Affirmative Action employer.
Women, minorities and persons with disabilities are strongly encouraged to apply.*

machine learning, computer vision / image analysis, and networks / systems. The Gates Foundation recently awarded a \$20 million grant to Biology and CSE faculty in the bioinformatics area, and the Semiconductor Research Corporation (SRC) together with the state of Indiana and the city of South Bend recently announced the Midwest Institute for Nanoelectronics Discovery, a research consortium led by Notre Dame, has received \$25 million in new funding.

The University of Notre Dame is a private, Catholic university with a doctoral research extensive Carnegie classification, and it is consistently ranked in USN&WR as a top-twenty national university. The South Bend area has a vibrant and diverse economy with affordable housing and excellent school systems, and is within easy driving distance of Chicago and Lake Michigan.

Screening of applications is on-going. Applicants should send (pdf format preferred) a CV, statement of teaching and research interests, and contact information for three professional references to: facultysearch@cse.nd.edu

The University of Notre Dame is an Equal Opportunity, Affirmative Action Employer.

University of Oregon Department of Computer and Information Science Faculty Position

The Computer and Information Science (CIS) Department at the University of Oregon seeks applicants for one or more full-time, tenure-track faculty positions beginning fall, 2010, at the rank of Assistant Professor. The University of Oregon is

an AAU research university located in Eugene, two hours south of Portland, and within one hour's drive of both the Pacific Ocean and the snow-capped Cascade Mountains.

The CIS Department is part of the College of Arts and Sciences and is housed within the Lorry Lokey Science Complex. The department offers B.S., M.S. and Ph.D. degrees. More information about the department, its programs and faculty can be found at <http://www.cs.uoregon.edu>, or by contacting the search committee at facultysearch@cs.uoregon.edu.

We offer a stimulating, friendly environment for collaborative research both within the department and with other departments on campus. Faculty in the department are affiliated with the Cognitive and Decision Sciences Institute, the Computational Science Institute, and the Neuro-Informatics Center.

Computer science is a rapidly evolving academic discipline. The department accordingly seeks to hire faculty in established areas as well as emerging directions in computer science. Applicants interested in interdisciplinary research are encouraged to apply. Applicants must have a Ph.D. in computer science or closely related field, a demonstrated record of excellence in research, and a strong commitment to teaching. A successful candidate will be expected to conduct a vigorous research program and to teach at both the undergraduate and graduate levels.

Applications will be accepted electronically through the department's web site (only). Application information can be found at <http://www.cs.uoregon.edu/Employment/>. Review of applications will begin January 4, 2010 and continue until the position is filled. Please address any questions to faculty.search@cs.uoregon.edu.

COMPUTER SCIENCE Cornell University

Multiple faculty positions are available at Cornell's Department of Computer Science. Candidates are invited to apply at all levels including tenured, tenure-track, or lecturer.

We are interested in applications from any area of computer science, including artificial intelligence, computational biology, databases, game design, graphics, machine learning, networking, programming languages, robotics, security, scientific computing, systems, and theory of computation.

To ensure full consideration, applications should be received by December 15, 2009, but will be accepted until all positions are filled.

Applicants should submit a curriculum vita, brief statements of research and teaching interests through the web at <http://www.cs.cornell.edu/apply> and arrange to have at least three references uploaded on the Web.

Cornell University, located in Ithaca, New York, is an inclusive, dynamic, and innovative Ivy League university and New York's land-grant institution. Its staff, faculty, and students impart an uncommon sense of larger purpose and contribute creative ideas and best practices to further the university's mission of teaching, research, and outreach.



Cornell University
Cornell University is an Affirmative Action/Equal Opportunity Employer and Educator.



Assistant Professor

The University of Kansas Electrical Engineering and Computer Science Department seeks to hire one tenure-track assistant professor to support its computer systems design research area. We are seeking to fill research and teaching needs in digital system design, multi-core architectures, systems-on-chip, electronic system-level design, computer system (hardware and software) synthesis, FPGA synthesis, and high-assurance systems. The successful applicant will have an earned Ph.D. in computer engineering, computer science, or related area. Faculty members in our department are expected to develop nationally recognized research programs while supporting our undergraduate and graduate teaching missions. The appointment will be effective as negotiated. Apply at <https://jobs.ku.edu> and search position no. 00004009. A complete application includes a letter of application, curriculum vita, and the names and addresses of three references. Applications will be reviewed beginning January 15, 2010 and will be accepted until the position is filled.

EO/AA

til the position is filled. Please address any questions to facultysearch@cs.uoregon.edu.

The University of Oregon is an equal opportunity/affirmative action institution committed to cultural diversity and is compliant with the Americans with Disabilities Act. We are committed to creating a more inclusive and diverse institution and seek candidates with demonstrated potential to contribute positively to its diverse community.

University of Pennsylvania Faculty

The University of Pennsylvania invites applicants for tenure-track appointments in computer science to start July 1, 2010. Tenured appointments will also be considered.

The Department of Computer and Information Science seeks individuals with exceptional promise for, or a proven record of, research achievement who will excel in teaching undergraduate and graduate courses and take a position of international leadership in defining their field of study. While exceptional candidates in all areas of core computer science may apply, of particular interest this year are candidates in who are working on the foundations of Market and Social Systems Engineering - the formalization, analysis, optimization, and realization of systems that increasingly integrate engineering, computational, and economic systems and methods. Candidates should have a vision and interest in defining the research and educational frontiers of this rapidly growing field.

The University of Pennsylvania is an Equal

Smartphone Programming in the Classroom

The Centre for **Mobile Education and Research (CMER) Academic Kit** provides the materials you need to integrate mobile devices into the CS/IT/Engineering curricula. The kit includes:

- A 12-week course on mobile application development with Java ME and BlackBerry API.
- A 6-week course on mobile Web-based application development.
- Five plug-in teaching modules to supplement courses on software engineering, information security, game development, web services, and operating systems.

All courses and modules can be taught across the CS/IT/Engineering curricula and most include: lesson slides, labs, tutorials, quizzes, and assignments.



These resources are for academic use only.

<http://cmer.cis.uoguelph.ca>

Opportunity/Affirmative Action Employer.

The Penn CIS Faculty is sensitive to "two-body problems" and would be pleased to assist with opportunities in the Philadelphia region.

For more detailed information regarding this position and application link please visit:

<http://www.cis.upenn.edu/departmental/facultyRecruiting.shtml>

University of Pennsylvania**Lecturer**

The University of Pennsylvania invites applicants for the position of Lecturer in Computer Science to start July 1, 2010. Applicants should hold a graduate degree (preferably a Ph.D.) in Computer Science or Computer Engineering, and have a strong interest in teaching with practical application. Lecturer duties include undergraduate and graduate level courses within the Master of Computer and Information Technology program (www.cis.upenn.edu/grad/mcit/). Of particular interest are applicants with expertise and/or interest in teaching computer hardware and architecture. The position is for one year and is renewable annually up to three years. Successful applicants will find Penn to be a stimulating environment conducive to professional growth in both teaching and research.

The University of Pennsylvania is an Equal Opportunity/Affirmative Action Employer.

The Penn CIS Faculty is sensitive to "two-body problems" and would be pleased to assist with opportunities in the Philadelphia region.

For more detailed information regarding this position and application link please visit:

<http://www.cis.upenn.edu/departmental/facultyRecruiting.shtml>

University of San Francisco**Tenure-track Position**

The Department of Computer Science at the University of San Francisco invites applications for a tenure-track position beginning in August, 2010. While special interest will be given to candidates in bioinformatics, game engineering, systems, and networking, qualified applicants from all areas of Computer Science are encouraged to apply. For full consideration, applications should be submitted by December 1, 2009.

More details, including how to apply, can be found here:

<http://www.cs.usfca.edu/job>

University of South Carolina**Department Chair - Computer Science and Engineering*****The Department of Computer Science and Engineering***

Department Chair - Computer Science and Engineering The Department of Computer Science and Engineering (www.cse.sc.edu) in the College of Engineering and Computing, University of South Carolina, seeks nominations and applications for the position of Department Chair. The Department offers bachelor's degrees in Computer Engineering, Computer Information Systems, and Computer Science, M.S., M.E. and Ph.D. degrees in Computer Science and Engineering, a Master of Software En-

gineering, and a Certificate of Graduate Studies in Information Assurance and Security. This is an active and engaged Department with 21 faculty members, including 20 with current research funding and 8 NSF CAREER award winners. Enrollment is over 300 undergraduate and 90 graduate students, including more than 50 doctoral students.

Applicants must have outstanding leadership and administrative skills, and credentials (including a Ph.D. in computer science, computer engineering, or related field) commensurate with appointment as a full professor with tenure. Nomination letters should include statements regarding the nominee's relevant credentials. Applicants should submit a current resume, a statement of professional interests and vision, and the names, affiliations, and contact information of professional references. Applications will be accepted until the position is filled and should be sent by email to cse-chair-search@cec.sc.edu.

The Department is particularly interested in receiving applications from minorities and women. The University of South Carolina is an affirmative action, equal opportunity employer.

University of Waterloo**David R. Cheriton School of Computer Science
Cheriton Chairs in Software Systems**

Applications are invited for one or two David R. Cheriton Chairs in Software Systems. These are senior positions and include substantial research support and teaching reduction. Candidates with outstanding research records in software systems (very broadly defined) are encouraged to apply. Successful applicants who join the University of Waterloo are expected to be leaders in research, have an active graduate student program and contribute to the overall development of the School. A Ph.D. in Computer Science, or equivalent, is required, with evidence of excellence in teaching and research. Rank and salary will be commensurate with experience, and appointments are expected to commence during the 2010 calendar year. The Chairs are tenured positions.

With over 70 faculty members, the University of Waterloo's David R. Cheriton School of Computer Science is the largest in Canada. It enjoys an excellent reputation in pure and applied research and houses a diverse research program of international stature. Because of its recognized capabilities, the School attracts exceptionally well-qualified students at both undergraduate and graduate levels. In addition, the University has an enlightened intellectual property policy which vests rights in the inventor: this policy has encouraged the creation of many spin-off companies including iAnywhere Solutions Inc., Maplesoft Inc., Open Text Corp and Research in Motion. Please see our website for more information: <http://www.cs.uwaterloo.ca>.

To submit an application, please register at the submission site: <http://www.cs.uwaterloo.ca/faculty-recruiting>. Once registered, instructions will be provided regarding how to submit your application. Applications will be considered as soon as possible after they are complete, and as long as positions are available.

The University of Waterloo encourages applications from all qualified individuals, including women, members of visible minorities, native peoples, and persons with disabilities. All qualified candidates are encouraged to apply; however, Canadian citizens and permanent residents will be given priority.

peoples, and persons with disabilities. All qualified candidates are encouraged to apply; however, Canadian citizens and permanent residents will be given priority.

University of Waterloo**David R. Cheriton School of Computer Science
Faculty Position in Software Engineering**

The University of Waterloo invites applications for a tenure-track or tenured faculty position in the David R. Cheriton School of Computer Science, in the area of Software Engineering. Candidates at all levels of experience are encouraged to apply. Successful applicants who join the University of Waterloo are expected to develop and maintain a productive program of research, attract and develop highly qualified graduate students, provide a stimulating learning environment for undergraduate and graduate students, and contribute to the overall development of the School. A Ph.D. in Computer Science, or equivalent, is required, with evidence of excellence in teaching and research. Rank and salary will be commensurate with experience, and appointments are expected to commence during the 2010 calendar year.

With over 70 faculty members, the University of Waterloo's David R. Cheriton School of Computer Science is the largest in Canada. It enjoys an excellent reputation in pure and applied research and houses a diverse research program of international stature. Because of its recognized capabilities, the School attracts exceptionally well-qualified students at both undergraduate and graduate levels. In addition, the University has an enlightened intellectual property policy which vests rights in the inventor: this policy has encouraged the creation of many spin-off companies including iAnywhere Solutions Inc., Maplesoft Inc., Open Text Corp and Research in Motion. Please see our website for more information: <http://www.cs.uwaterloo.ca>.

To submit an application, please register at the submission site: <http://www.cs.uwaterloo.ca/faculty-recruiting>. Once registered, instructions will be provided regarding how to submit your application. Applications will be considered as soon as possible after they are complete, and as long as positions are available.

The University of Waterloo encourages applications from all qualified individuals, including women, members of visible minorities, native peoples, and persons with disabilities. All qualified candidates are encouraged to apply; however, Canadian citizens and permanent residents will be given priority.

University of Waterloo**David R. Cheriton School of Computer Science
Faculty Positions in Health Informatics**

The University of Waterloo invites applications for one or two tenure-track or tenured faculty positions in the David R. Cheriton School of Computer Science, in the area of Health Informatics. We define health informatics broadly to include medical informatics and biomedical systems. The School plans to start a new graduate degree program in health informatics in September 2010.

Candidates at all levels of experience are en-

couraged to apply. Successful applicants who join the University of Waterloo are expected to develop and maintain a productive program of research, attract and develop highly qualified graduate students, provide a stimulating learning environment for undergraduate and graduate students, and contribute to the overall development of the School. A Ph.D. in Computer Science, or equivalent, is required, with evidence of excellence in teaching and research. Rank and salary will be commensurate with experience, and appointments are expected to commence during the 2010 calendar year.

With over 70 faculty members, the University of Waterloo's David R. Cheriton School of Computer Science is the largest in Canada. It enjoys an excellent reputation in pure and applied research and houses a diverse research program of international stature. Because of its recognized capabilities, the School attracts exceptionally well-qualified students at both undergraduate and graduate levels. In addition, the University has an enlightened intellectual property policy which vests rights in the inventor: this policy has encouraged the creation of many spin-off companies including iAnywhere Solutions Inc., Maplesoft Inc., Open Text Corp and Research in Motion. Please see our website for more information: <http://www.cs.uwaterloo.ca>.

To submit an application, please register at the submission site: <http://www.cs.uwaterloo.ca/faculty-recruiting>. Once registered, instructions will be provided regarding how to submit your application. Applications will be considered as soon as possible after they are complete, and as long as positions are available.

The University of Waterloo encourages applications from all qualified individuals, including women, members of visible minorities, native peoples, and persons with disabilities. All qualified candidates are encouraged to apply; however, Canadian citizens and permanent residents will be given priority.

University of Wisconsin Oshkosh Computer Science

University of Wisconsin Oshkosh seeks one, possibly two, tenure-track Computer Science positions (rank open) beginning September 1, 2010. Ph.D. in Computer Science or a closely related field required. Duties include teaching 9 credits/semester in an ABET-accredited program, conducting research, advising majors. Submit letter of application, curriculum vita, teaching statement, research statement, transcripts (official or photocopy), and 3 current letters of recommendation to: Tom Naps, Chair, Computer Science Department, University of Wisconsin Oshkosh, 800 Algoma Blvd., Oshkosh, WI 54901-8643. Application deadline: November 13, 2009. Employment requires criminal background check. AA/EOE.

Ursinus College Assistant Professor of Computer Science

Ursinus College, a liberal arts college located in the suburbs of Philadelphia, invites applications for a tenure-track position in Computer Science. Please see <http://www.ursinus.edu/NetCommunity/Page.aspx?pid=390> for more information.

Virginia Tech

Department of Computer Science Artificial Intelligence/Machine Learning Senior Position

The Department of Computer Science at Virginia Tech (www.cs.vt.edu) invites applications for a full-time tenured position at the Professor or Associate Professor rank from candidates in artificial intelligence with particular interests in machine learning, knowledge representation, or data mining. Candidates should have an established record of scholarship, leadership, and collaboration in computing and interdisciplinary areas; demonstrated ability to contribute to teaching at the undergraduate and graduate levels in AI and related subjects; sensitivity to issues of diversity in the campus community; and the skills needed to establish and grow a multidisciplinary research group.

CS@VT has over 40 tenure-track research-oriented faculty. PhD production is among the top 30 in the US and annual research expenditures exceed \$6 million. There are rich opportunities in a highly collaborative department with strengths in HCI, HPC, CS education, digital libraries, computational biology and bioinformatics. Active interdisciplinary research also explores Cyber-Arts, digital government, problem-solving environments. Emphases on security and personal health informatics are underway in collaboration with the newly formed VT-Carilion Research Institute associated with the VT-Carilion School of Medicine, opening in Fall 2010.

CS@VT is part of the College of Engineering (www.eng.vt.edu) in a comprehensive research university with more than 26,000 students. The main campus is in Blacksburg, which is consistently ranked among the country's best places to live (<http://www.liveinblacksburg.com/>).

Salary for suitably qualified applicants is competitive and commensurate with experience. Virginia Tech is an Equal Opportunity/Affirmative Action Institution.

Applications must be submitted online to <https://jobs.vt.edu> for posting #090529. Applicant screening will begin January 15, 2010 and continue until the position is filled. Inquiries should be directed to Dennis Kafura, Hiring Committee Chair, kafura@cs.vt.edu.

Washington University in Saint Louis Multiple tenure-track/tenured faculty positions

The Department of Computer Science and Engineering (CSE) and the School of Medicine (WUSM) are jointly searching for multiple tenure-track faculty members with outstanding records of computing research and a serious interest in collaborative research on problems related to biology and/or medicine. Appointments may be made wholly within CSE or jointly with the Departments of Medicine or Pathology & Immunology.

A key initiative in the CSE Department's strategic plan is Integrating Computing and Science. As part of that initiative, we expect to make synergistic hires with a combined research portfolio spanning the range from fundamental computer science/engineering to applied research focused on science or medicine. Specific areas of interest include, but are not limited to:

- ▶ Analysis of complex genetic, genomic, proteomic, and metabolomic datasets;
- ▶ Theory/Algorithms with the potential for biomedical applications;
- ▶ Image analysis or visualization with the potential for biomedical applications;
- ▶ Databases, medical informatics, clinical or public-health informatics;
- ▶ Computer engineering with applications to medicine or the natural sciences;
- ▶ All areas of computational biology and biomedical informatics

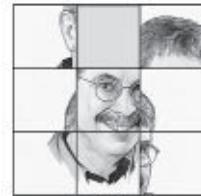
These positions will continue a successful, ongoing strategy of collaborative research between CSE and the School of Medicine, which is consistently ranked among the top 3 medical schools in the United States. CSE currently consists of 24 tenured and tenure-track faculty members, 71 Ph.D. students, and a stellar group of undergraduates with a history of significant research contributions. The Department seeks to build on and complement its strengths in biological sequence analysis, biomedical image analysis, and biomedical applications of novel computing architectures. Exceptional candidates conducting research in other areas of Computer Science are also encouraged to apply.

Washington University is a private university with roughly 6,000 full-time undergraduates and 6,000 graduate students. It has one of the most attractive university campuses anywhere, and is located in a lovely residential neighborhood, adjacent to one of the nation's largest urban parks, in the heart of a vibrant metropolitan area. St. Louis is a wonderful place to live, providing access to a wealth of cultural and entertainment opportunities without the everyday hassles of the largest cities.

We anticipate appointments at the rank of Assistant Professor; however, in the case of exceptionally qualified candidates appointments at any rank may be considered. Applicants must have a Ph.D. in computer science, computer engineering, electrical engineering, biomedical engineering, or a closely related field and a record of excellence in teaching and research appropriate to the appointment level. The selected candidate is expected to build an externally-supported research program, teach and mentor students at the graduate and undergraduate levels, and foster interdisciplinary interactions with colleagues throughout the university. Candidates who would contribute to enhancing diversity at the departmental and university levels are strongly encouraged to apply. Applications from academic couples are welcomed and encouraged.

Qualified applicants should submit a complete application (cover letter, curriculum vita, research statement, teaching statement, and names of at least three references) electronically by following the directions provided at <http://cse.wustl.edu/faculty-recruiting/>. Other communications may be directed to Prof. Michael Brent, Department of Computer Science and Engineering, Campus Box 1045, Washington University, One Brookings Drive, St. Louis, MO 63130-4899.

Applications submitted before January 31, 2010 will receive full consideration. Washington University is an equal opportunity/affirmative action employer.



DOI:10.1145/1592761.1592786

Peter Winkler

Puzzled Covering the Plane

Welcome to three new puzzles. Solutions to the first two will be published next month; the third is (as yet) unsolved. In each, the issue is how your intuition matches up with the mathematics.

1. One hundred identical coins lie flat on a rectangular table in such a way that no more coins can be added without the coins overlapping. (A coin is allowed to extend over the edge as long as its center is on the table.) Now the coins are cleared from the table, and you must prove you can cover the table with 400 of the same coins. (Again overhang is allowed, but this time the coins must overlap.)

The original layout of coins is called a maximal packing (of a rectangle by discs). What is sought is a covering, with the claim that it requires no more than four times as many discs as a maximal packing. One approach might be to try using 100 larger discs, then replace each of them with four of the original size. However, the second part doesn't seem to work. What would you suggest?

2. Using two full sets of parallel lines, you can cover an infinite plane in such a way that every point belongs to exactly two lines (an arrangement called an exact double cover of the plane by lines). For example, you can use all the vertical lines and all the horizontal lines; every point on the plane belongs to one line from each category.

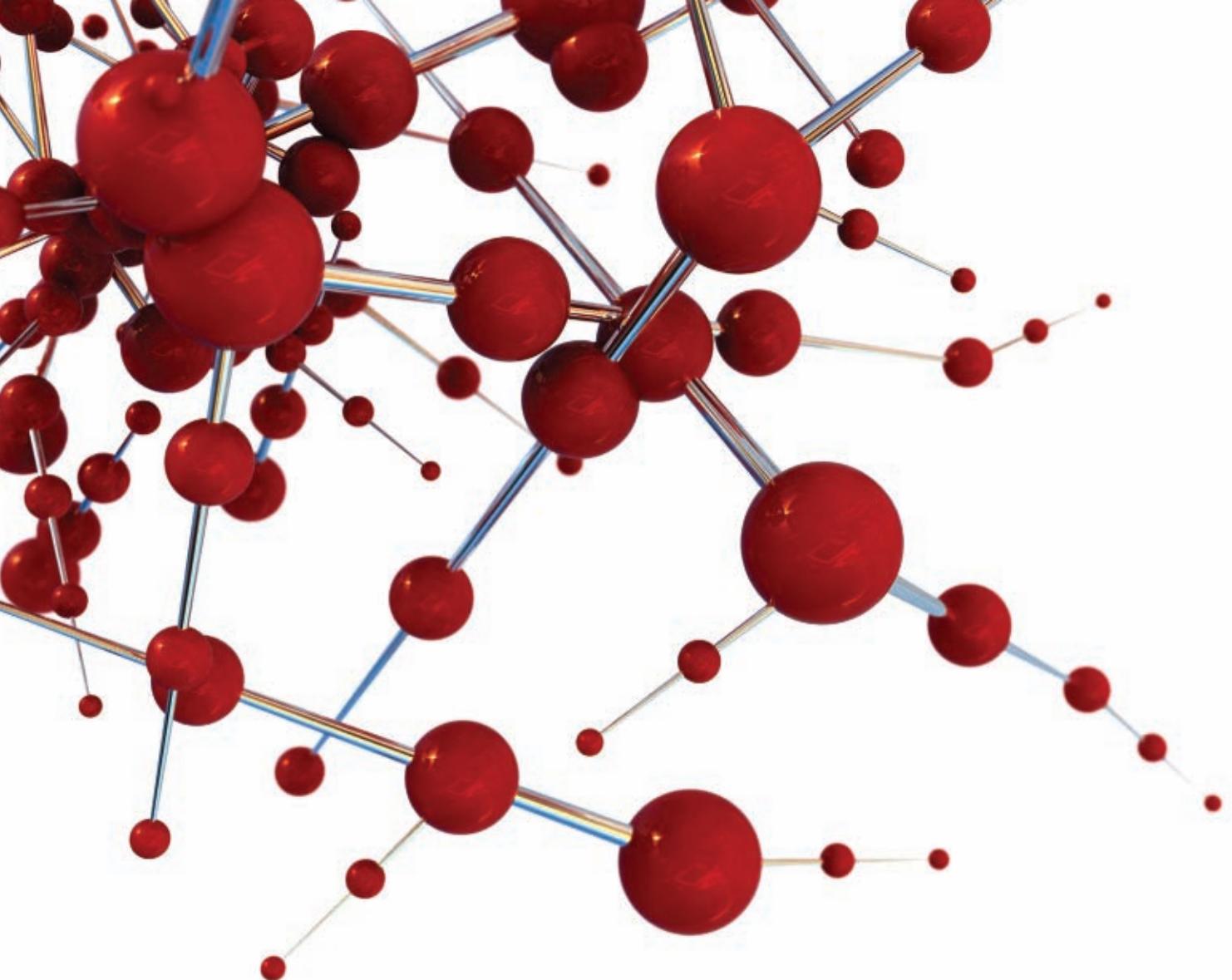
Is there another way to do it? Is there an exact double cover using lines that extend in more than two directions? For example, you could try taking all lines tangential to some fixed circle. This would work outside the circle but would hit each point on the circle only once and miss the inside entirely. Hmm...

3. Suppose you have a collection of open-unit discs covering a plane everywhere at least a thousand discs deep; that is, every point on the infinite plane is covered by at least a thousand discs. Now prove you can color each one red or blue in such a way that by themselves the red discs would cover the plane, and the blue discs would cover the plane. For each covering, each point of the plane must be in one or more discs of that color—surely not too much to ask.

Maybe not, but no one has proved it can be done, even for a billion-fold cover. János Pach of New York University is the originator of (and expert on) this wonderful open problem. In his paper “Covering the Plane with Convex Polygons” (*Discrete and Computational Geometry* 1, 1 (1986), 73–81), he proved that, for any symmetric polygon P , there is a number k such that any k -fold covering of the plane (by translates of P) can be partitioned into two separate covers. However, no such k is known when the polygon is a disc. Personally, I think $k = 4$ ought to be enough. What do you think?

Readers are encouraged to submit prospective puzzles for future columns to puzzled@cacm.acm.org.

Peter Winkler (puzzled@cacm.acm.org) is Professor of Mathematics and of Computer Science and Albert Bradley Third Century Professor in the Sciences at Dartmouth College, Hanover, NH.



**CONNECT WITH OUR
COMMUNITY OF EXPERTS.**

www.reviews.com



Association for
Computing Machinery

Reviews.com

They'll help you find the best new books
and articles in computing.

Computing Reviews is a collaboration between the ACM and Reviews.com.



Think Parallel....

It's not just what we make.
It's what we make possible.

Advancing Technology Curriculum
Driving Software Evolution
Fostering Tomorrow's Innovators

Learn more at: www.intel.com/thinkparallel

