

COMMUNICATIONS

CACM.ACM.ORG OF THE

ACM

10/2015 VOL.58 NO.10

Discovering Genes Involved in Disease and the Mystery of Missing Heritability

Crash Consistency

Concerns Rise
about AI

Seeking Anonymity
in an Internet
Panopticon

What Can Be
Done about
Gender Diversity
in Computing?
A Lot!



SIGGRAPH ASIA 2015 KOBÉ

The 8th ACM SIGGRAPH Conference
and Exhibition on Computer Graphics
and Interactive Techniques in Asia

Register online before
2 October 2015, 15:00 UTC/GMT
& enjoy early bird discounts of up to

20%!

sa2015.siggraph.org/registration

[RE]VOLUTIONARY

CONFERENCE 2 - 5 November 2015

EXHIBITION 3 - 5 November 2015

KOBE CONVENTION CENTER, KOBE, JAPAN

SA2015.SIGGRAPH.ORG

Sponsored by
ACM SIGGRAPH



**Previous
A.M. Turing Award
Recipients**

1966 A.J. Perlis
1967 Maurice Wilkes
1968 R.W. Hamming
1969 Marvin Minsky
1970 J.H. Wilkinson
1971 John McCarthy
1972 E.W. Dijkstra
1973 Charles Bachman
1974 Donald Knuth
1975 Allen Newell
1975 Herbert Simon
1976 Michael Rabin
1976 Dana Scott
1977 John Backus
1978 Robert Floyd
1979 Kenneth Iverson
1980 C.A.R Hoare
1981 Edgar Codd
1982 Stephen Cook
1983 Ken Thompson
1983 Dennis Ritchie
1984 Niklaus Wirth
1985 Richard Karp
1986 John Hopcroft
1986 Robert Tarjan
1987 John Cocke
1988 Ivan Sutherland
1989 William Kahan
1990 Fernando Corbató
1991 Robin Milner
1992 Butler Lampson
1993 Juris Hartmanis
1993 Richard Stearns
1994 Edward Feigenbaum
1994 Raj Reddy
1995 Manuel Blum
1996 Amir Pnueli
1997 Douglas Engelbart
1998 James Gray
1999 Frederick Brooks
2000 Andrew Yao
2001 Ole-Johan Dahl
2001 Kristen Nygaard
2002 Leonard Adleman
2002 Ronald Rivest
2002 Adi Shamir
2003 Alan Kay
2004 Vinton Cerf
2004 Robert Kahn
2005 Peter Naur
2006 Frances E. Allen
2007 Edmund Clarke
2007 E. Allen Emerson
2007 Joseph Sifakis
2008 Barbara Liskov
2009 Charles P. Thacker
2010 Leslie G. Valiant
2011 Judea Pearl
2012 Shafi Goldwasser
2012 Silvio Micali
2013 Leslie Lamport
2014 Michael Stonebraker

ACM A.M. TURING AWARD NOMINATIONS SOLICITED

Nominations are invited for the 2015 ACM A.M. Turing Award. This is ACM's oldest and most prestigious award and is presented annually for major contributions of lasting importance to computing. Although the long-term influences of the nominee's work are taken into consideration, there should be a particular outstanding and trendsetting technical achievement that constitutes the principal claim to the award. The recipient presents an address at an ACM event that will be published in an ACM journal. The award is accompanied by a prize of \$1,000,000. Financial support for the award is provided by Google Inc.

**Nomination information and the online submission form
are available on:**
http://amturing.acm.org/call_for_nominations.cfm

**Additional information on the Turing Laureates
is available on:**
<http://amturing.acm.org/byyear.cfm>

**The deadline for nominations/endorsements is
November 30, 2015.**

**For additional information on ACM's award program
please visit: www.acm.org/awards/**



Association for
Computing Machinery

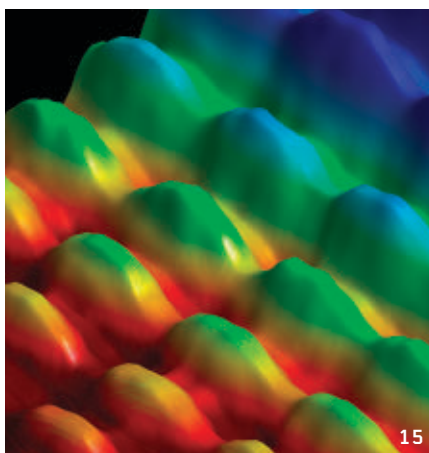
Departments

- 5 **Editor's Letter**
What Can Be Done about Gender Diversity in Computing? A Lot!
By Moshe Y. Vardi
-
- 7 **Cerf's Up**
The Third Heidelberg Laureate Forum
By Vinton G. Cerf
-
- 10 **Letters to the Editor**
Ban 'Naked' Braces!
-
- 12 **BLOG@CACM**
The Morality of Online War; the Fates of Data Analytics, HPC
John Arquilla considers justifications for warfare in the cyber realm, while Daniel Reed looks ahead at big data and exascale computing.
-
- 33 **Calendar**
-
- 98 **Careers**

Last Byte

- 104 **Future Tense**
Processional
Information processing gives spiritual meaning to life, for those who make it their life's work.
By William Sims Bainbridge

News

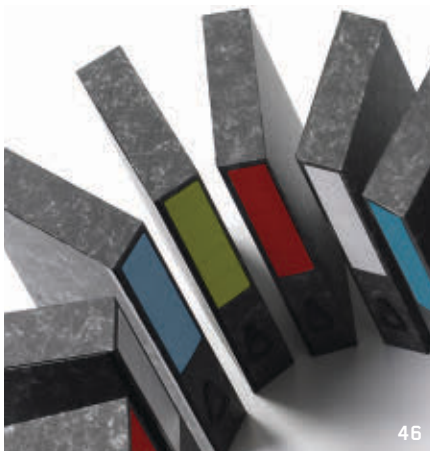


- 15 **Scientists Update Views of Light**
Experiment sheds new light on wave-particle duality.
By Gary Anthes
-
- 18 **Automotive Systems Get Smarter**
Automotive infotainment systems are driving changes to automobiles, and to driver behavior.
By Samuel Greengard
-
- 21 **Cyber Policies on the Rise**
A growing number of companies are taking out cybersecurity insurance policies to protect themselves from the costs of data breaches.
By Keith Kirkpatrick

Viewpoints

- 24 **Inside Risks**
Keys Under Doormats
Mandating insecurity by requiring government access to all data and communications.
By Peter G. Neumann et al.
-
- 27 **Technology Strategy and Management**
In Defense of IBM
The ability to adjust to various technical and business disruptions has been essential to IBM's success during the past century.
By Michael A. Cusumano
-
- 29 **Kode Vicious**
Storming the Cubicle
Acquisitive redux.
By George V. Neville-Neil
-
- 32 **The Business of Software**
Thinking Thoughts
On brains and bytes.
By Phillip G. Armour
-
- 35 **Historical Reflections**
Computing Is History
Reflections on the past to inform the future.
By Thomas J. Misa
-
- 38 **Viewpoint**
Rise of Concerns about AI: Reflections and Directions
Research, leadership, and communication about AI futures.
By Thomas G. Dietterich and Eric J. Horvitz
-
- Watch the authors discuss their work in this exclusive *Communications* video.
<http://cacm.acm.org/videos/rise-of-concerns-about-ai-reflections-and-directions>
-
- 41 **Viewpoint**
Life After MOOCs
Online science education needs a new revolution.
By Phillip Compeau and Pavel A. Pevzner

Practice



46

- 46 **Crash Consistency**
Rethinking the fundamental abstractions of the file system.
By T.S. Pillai, V. Chidambaram, R. Alagappan, S. Al-Kiswani, A.C. Arpaci-Dusseau, and R.H. Arpaci-Dusseau

- 52 **Dismantling the Barriers to Entry**
We have to choose to build a Web that is accessible to everyone.
By Rich Harris

q Articles' development led by **acmqueue**
queue.acm.org

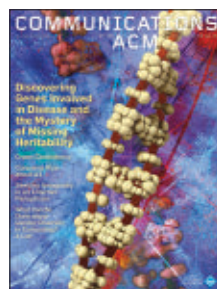
Contributed Articles



70

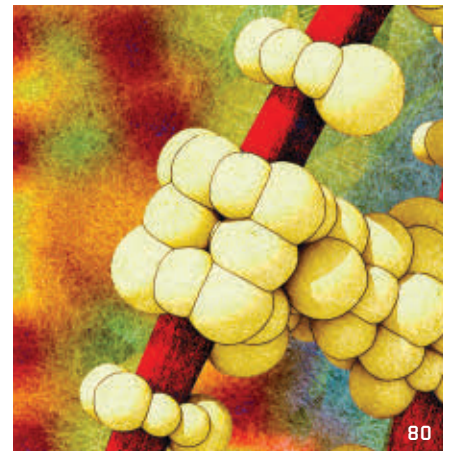
- 58 **Seeking Anonymity in an Internet Panopticon**
The Dissent system aims for a quantifiably secure, collective approach to anonymous communication online.
By Joan Feigenbaum and Bryan Ford

- 70 **Framing Sustainability as a Property of Software Quality**
This framework addresses the environmental dimension of software performance, as applied here by a paper mill and a car-sharing service.
By Patricia Lago, Sedef Akinli Koçak, Ivica Crnkovic, and Birgit Penzenstadler



About the Cover:
Discovering the variants involved in human disease calls on computing scientists to lead the exploration of huge datasets. Eleazar Eskin examines the mystery of missing heritability (p. 80) Cover illustration by Charles Wiese; www.charleswiese.com.

Review Articles



80

- 80 **Discovering Genes Involved in Disease and the Mystery of Missing Heritability**
The challenge of missing heritability offers great contribution options for computer scientists.
By Eleazar Eskin



Watch the author discuss his work in this exclusive *Communications* video.
<http://cacm.acm.org/videos/discovering-genes-involved-in-disease-and-the-mystery-of-missing-heritability>

Research Highlights

- 90 **Technical Perspective**
Not Just a Matrix Laboratory Anymore
By Cleve Moler
- 91 **Computing Numerically with Functions Instead of Numbers**
By Lloyd N. Trefethen



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.

Acting Director and CEO and Deputy Executive Director and COO
Patricia Ryan
Director, Office of Information Systems
Wayne Graves
Director, Office of Financial Services
Darren Ramdin
Director, Office of SIG Services
Donna Cappo
Director, Office of Publications
Bernard Rous
Director, Office of Group Publishing
Scott E. Delman

ACM COUNCIL

President
Alexander L. Wolf
Vice-President
Vicki L. Hanson
Secretary/Treasurer
Erik Altman
Past President
Vinton G. Cerf
Chair, SGB Board
Patrick Madden
Co-Chairs, Publications Board
Jack Davidson and Joseph Konstan
Members-at-Large
Eric Allman; Ricardo Baeza-Yates;
Cherri Pancake; Radia Perlman;
Mary Lou Soffa; Eugene Spafford;
Per Stenström
SGB Council Representatives
Paul Beame; Barbara Boucher Owens

BOARD CHAIRS

Education Board
Mehran Sahami and Jane Chu Prey
Practitioners Board
George Neville-Neil

REGIONAL COUNCIL CHAIRS

ACM Europe Council
Fabrizio Gagliardi
ACM India Council
Srinivas Padmanabhuni
ACM China Council
Jiaquan Sun

PUBLICATIONS BOARD

Co-Chairs
Jack Davidson; Joseph Konstan
Board Members
Ronald F. Boisvert; Nikil Dutt; Roch Guerrin;
Carol Hutchins; Yannis Ioannidis;
Catherine McGeoch; M. Tamer Ozsu;
Mary Lou Soffa

ACM U.S. Public Policy Office

Renee Dopplack, Director
1828 L Street, N.W., Suite 800
Washington, DC 20036 USA
T (202) 659-9711; F (202) 667-1066

Computer Science Teachers Association
Lissa Clayborn, Acting Executive Director

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.

STAFF

DIRECTOR OF GROUP PUBLISHING
Scott E. Delman
cacm-publisher@cacm.acm.org

Executive Editor
Diane Crawford
Managing Editor
Thomas E. Lambert
Senior Editor
Andrew Rosenbloom
Senior Editor/News
Larry Fisher
Web Editor
David Roman
Rights and Permissions
Deborah Cotton

Art Director
Andrij Borys
Associate Art Director
Margaret Gray
Assistant Art Director
Mia Angelica Balaquiot
Designer
Iwona Usakiewicz
Production Manager
Lynn D'Addesio
Director of Media Sales
Jennifer Ruzicka
Publications Assistant
Juliet Chance

Columnists
David Anderson; Phillip G. Armour;
Michael Cusumano; Peter J. Denning;
Mark Guzdial; Thomas Haigh;
Leah Hoffmann; Mari Sako;
Pamela Samuelson; Marshall Van Alstyne

CONTACT POINTS

Copyright permission
permissions@cacm.acm.org
Calendar items
calendar@cacm.acm.org
Change of address
acmhq@cacm.acm.org
Letters to the Editor
letters@cacm.acm.org

WEBSITE

<http://cacm.acm.org>

AUTHOR GUIDELINES

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

ACM ADVERTISING DEPARTMENT

2 Penn Plaza, Suite 701, New York, NY
10121-0701
T (212) 626-0686
F (212) 869-0481

Director of Media Sales
Jennifer Ruzicka
jen.ruzicka@hq.acm.org

Media Kit acmm mediasales@acm.org

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

EDITORIAL BOARD

EDITOR-IN-CHIEF
Moshe Y. Vardi
eic@cacm.acm.org

NEWS

Co-Chairs
William Pulleyblank and Marc Snir
Board Members
Mei Kobayashi; Kurt Mehlhorn;
Michael Mitzenmacher; Rajeev Rastogi

VIEWPOINTS

Co-Chairs
Tim Finin; Susanne E. Hambrusch;
John Leslie King
Board Members
William Aspray; Stefan Bechtold;
Michael L. Best; Judith Bishop;
Stuart I. Feldman; Peter Freeman;
Mark Guzdial; Rachelle Hollander;
Richard Ladner; Carl Landwehr;
Carlos Jose Pereira de Lucena;
Beng Chin Ooi; Loren Terveen;
Marshall Van Alstyne; Jeannette Wing

PRACTICE

Co-Chairs
Stephen Bourne
Board Members
Eric Allman; Terry Coatta; Stuart Feldman;
Benjamin Fried; Pat Hanrahan;
Tom Limoncelli; Kate Matsudaira;
Marshall Kirk McKusick; George Neville-Neil;
Theo Schlossnagle; Jim Waldo

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

CONTRIBUTED ARTICLES

Co-Chairs
Andrew Chien and James Larus
Board Members
William Aiello; Robert Austin; Elisa Bertino;
Gilles Brassard; Kim Bruce; Alan Bundy;
Peter Buneman; Peter Druschel;
Carlo Ghezzi; Carl Gutwin; Gal A. Kaminka;
James Larus; Igor Markov; Gail C. Murphy;
Bernhard Nebel; Lionel M. Ni; Kenton O'Hara;
Sriram Rajamani; Marie-Christine Rousset;
Avi Rubin; Krishan Sabnani;
Ron Shamir; Yoav Shoham; Larry Snyder;
Michael Vitale; Wolfgang Wahlster;
Hannes Werthner; Reinhard Wilhelm

RESEARCH HIGHLIGHTS

Co-Chairs
Azer Bestavros and Gregory Morrisett
Board Members
Martin Abadi; Amr El Abbadi; Sanjeev Arora;
Nina Balcan; Dan Boneh; Andrei Broder;
Doug Burger; Stuart K. Card; Jeff Chase;
Jon Crowcroft; Sandhya Dwaekadas;
Matt Dwyer; Alon Halevy; Norm Jouppi;
Andrew B. Kahng; Henry Kautz; Xavier Leroy;
Steve Marschner; Kobbi Nissim;
Steve Seitz; Guy Steele, Jr.; David Wagner;
Margaret H. Wright

WEB

Chair
James Landay
Board Members
Marti Hearst; Jason I. Hong;
Jeff Johnson; Wendy E. MacKay

ACM Copyright Notice

Copyright © 2015 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) 626-0686.

Single Copies

Single copies of *Communications of the ACM* are available for purchase. Please contact acmhq@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

Printed in the U.S.A.



Association for
Computing Machinery





DOI:10.1145/2816937

Moshe Y. Vardi

What Can Be Done about Gender Diversity in Computing? A Lot!

THE 2015 GRACE HOPPER Celebration of Women in Computing (GHC, for short) will take place October 14–16 in Houston, TX. GHC is an annual conference designed to bring the research and career interests of women in computing to the forefront. It is the world's largest gathering of women in computing. GHC is organized by the Anita Borg Institute for Women in Technology in partnership with ACM. This year's event is expected to bring together more than 12,000—mostly female—computer scientists!

But this impressive number should not be taken to mean all is well on the gender-diversity front. Far from it! According to the most recent Taulbee Survey (covering academic year 2013–2014), conducted by the Computing Research Association in North America, only 14.7% of CS bachelor's degrees went to women. The U.S. Department of Education's data shows the female participation level in computing peaked at about 35% in 1984, more than twice as high as it is today.

The low participation of women in computer science has been, indeed, a matter of concern for many years. The Anita Borg Institute was founded in 1997 “to recruit, retain, and advance women in technology.” (GHC is the Institute's most prominent program.) The National Center for Women & Information Technology, founded in 2004, is another organization that works to increase the meaningful participation of girls and women in computing. And yet, we seem to be regressing rather than progressing on this issue.

The gender-diversity issue received a fair amount of attention over the past year, when several major technology companies released workforce-diversity

data, showing, no surprise, a significant underrepresentation of women in technical jobs. Tech companies point, of course, to the narrow pipeline of women with computing degrees to explain this underrepresentation, but the culture inside some of these companies also seems to be a major factor. In fact, the male-dominated tech culture gave rise to the phrase “brogramming,” a slang term used to refer to computer code produced by “bros” (slang for male friends). A magazine article on the subject, titled: “Brogramming—The Disturbing Rise of Frat Culture in Silicon Valley,” was circulated widely a few years ago.

But amid the deluge of bad news, one can find some points of light. Carnegie Mellon University decided in the late 1990s to take decisive action on gender diversity and was able to increase the percentage of women entering its computer science program to 40%. A similar outcome was recently reported by Harvey Mudd College. The Anita Borg Institute, together with Harvey Mudd College, launched the BRAID Initiative (<http://anitaborg.org/braid-building-recruiting-and-inclusion-for-diversity/>) in 2014 to increase the percentage of women and students of color majoring in computer science in the U.S.

At my own institution, Rice University, we were able to raise the percentage of declared female majors (Rice students declare their major toward the end of the second year of study) from 14% in 2007 to 30% in 2014. What distinguishes Rice from Carnegie Mellon and Harvey Mudd is that computer science at Rice has no control whatsoever of the undergraduate-admission pipeline. To raise the level of participation of women in computer science at Rice required a departmental decision that we cannot simply blame the situa-

tion on the narrow pipeline of female high school graduates with interest in CS. Several measures were adopted:

- Changing CS1 from a course about programming techniques to a course about computational thinking. The latter course is more popular with both male and female students, and also puts students with widely varied high school computing experiences on a more level playing field.

- Creating a club for female computer science students. There are a fair number of female students who desire the camaraderie of an all-women computing group on campus, given that the CS student body is still very much male dominated.

- Having faculty members, male and female, develop mentoring relationships with female students to motivate and encourage them, including offering opportunities for interaction beyond the classroom, for example, undergraduate research opportunities.

- Continually dispel myths about the preparedness and ability of women for technical jobs.

- Last, but not least, sending female students to GHC. Especially given Rice's small size, this allows students to see there are many successful women in the field.

The bottom line is that while the gender-diversity problem is a very challenging one, it is not hopeless. Indeed, the pipeline is narrow, but it can be expanded, one student at a time, one program at a time, one company at a time. Institutional and personal commitments can make a significant difference!

Follow me on Facebook, Google+, and Twitter.

Moshe Y. Vardi, EDITOR-IN-CHIEF

Copyright held by author.



17th International Conference on **Multimodal Interaction**

November 9-13, 2015

<http://icmi.acm.org/2015/>

Seattle, WA, USA

- ✦ Multimodal signal and interaction processing technologies
- ✦ Multimodal models for human-human and human-machine interaction
- ✦ Multimodal data, evaluation and tools
- ✦ Multimodal systems and applications

Keynote Speakers

Samy Bengio, Google, USA

Kerstin Dautenhahn, University of Hertfordshire, UK

Organising Committee

General Chairs

Zhengyou Zhang (*Microsoft Research, USA*)
Phil Cohen (*VoiceBox Technologies, USA*)

Program Chairs

Dan Bohus (*Microsoft Research, USA*)
Radu Horaud (*INRIA Grenoble Rhone-Alpes, France*)
Helen Meng (*Chinese University of Hong Kong, China*)

Workshop Chairs

Jean-Marc Odobez (*IDIAP, Switzerland*)
Hayley Hung (*Technical University of Delft, Netherlands*)

Demo Chairs

Hrvoje Benko (*Microsoft Research, USA*)
Stefan Scherer (*University of Southern California, USA*)

Multimodal Grand Challenge Chairs

Cosmin Munteanu (*University of Toronto, Canada*)
Marcelo Worsley (*Stanford University, USA*)

Doctoral Consortium Chairs

Carlos Busso (*University of Texas at Dallas, USA*)
Vidhyasaharan Sethu (*University of New South Wales, Australia*)

Publication Chair

Lisa Anthony (*University of Florida at Gainesville, USA*)

Publicity Chairs

Xilin Chen (*Chinese Academy of Sciences, China*)
Louis-Philippe Morency (*Carnegie Mellon University, USA*)
Christian Müller (*DFKI GmbH, Germany*)

Sponsorship Chairs

YingLi Tian (*City University of New York, USA*)
Laurence Devillers (*LIMSI, France*)
Fei Wu (*Zhejiang University, China*)

Local Organization Chairs

Qin Cai (*Microsoft Research, USA*)
Zicheng Liu (*Microsoft Research, USA*)

Finance Chair

David McGee (*Adapx, USA*)

Web Chair

Hyunggu Jung (*University of Washington, USA*)

Volunteer Chair

Ankur Agrawal (*University of Washington, USA*)



Vinton G. Cerf

DOI:10.1145/2818988

The Third Heidelberg Laureate Forum

IHAVE JUST returned from the Third Heidelberg Laureate Forum^a and it equaled and perhaps outperformed the previous two. It was also, however, a poignant event because we were reminded of the ephemeral nature of our human lives. The instigator and patron of these conferences, Klaus Tschira, passed away unexpectedly in March 2015. His enthusiasm, curiosity, and capacity for making things happen were greatly missed, but his spirit lives on in the leadership and staff of his foundations. They showed renewed commitment to Klaus' vision, warmth, and generosity in the conduct of this extraordinary gathering.

A new element was introduced this year: a truly inspiring lecture by Nobel Prize winner Stefan W. Hell on the development of super-resolved fluorescence microscopy. Combining stunningly clear, animated, technical slides with his personal story, Stefan told of a compelling and dramatic odyssey toward a brilliant insight into the improved resolution of optical microscopy. Each future Heidelberg Laureate Forum will feature the "Lindau Lecture" by a Nobel Prize winner. The lecture is named after an annual meeting^b of Nobel Prize winners and 600 students that has been held since 1951 in Lindau, Germany. It is now also planned that at each Lindau meeting, there will be a "Heidelberg Lecture" by one of the Heidelberg laureates. This has a personal consequence for me, as I have been invited to make that first lecture in 2016. This is a daunting prospect and I hope I will be up to it!

The lectures were once again thought provoking and stimulated a lot of dis-

cussion. There were many poster sessions and workshops that stirred comparable interactions and, as usual, there was ample time for informal discussion among the students and laureates. For me, the opportunity to explore ideas at meal times and on excursions represented a substantial portion of the value of this annual convocation.

Among the excursions was a new one (for me) to the Speyer Technik Museum^c led by Gerhard Daum. The museum was originally built to house the Russian BURAN spacecraft^d—the counterpart to the U.S. Space Shuttle. Daum, who had been collecting space artifacts since boyhood, brought hundreds of additional artifacts to the museum, including a full-size Lunar Excursion Module in a moon-diorama setting along with the moon rover vehicle and figures in spacesuits. The most surprising artifact was an actual 3.4-billion-year-old moonstone collected during the Apollo 15 mission! The exhibition tells the story of the American, European, and Russian space efforts and includes many original artifacts from each. I spent at least an hour and a half with Daum, whose knowledge of the space programs around the world is encyclopedic in scope and rivaled only by his unbridled enthusiasm for space exploration.

ACM President Alexander Wolf represented ACM ably and eloquently and chaired one of the morning lecture sessions. Many fellow ACM Turing Award recipients were key contributors to the event. Leslie Lamport gave a compelling lecture advocating the use of mathematics in the description of computer systems to aid in their construction and analysis. Manuel Blum brought drama to the stage by demonstrating how he

could brief four volunteers on ways to "compute" passwords at need without memorizing them. All four succeeded! Sir Tony Hoare reminded us the roots of computation and science go back to Aristotle and Euclid and other philosophers who have advanced the state of the art over millennia. Edmund Clarke drew our attention to the importance of being able to say something about the correctness of computations dealing with real, continuous quantities ("hybrid systems"). As we enter into a period in which we depend increasingly on cyber-physical systems, such considerations are vital. Ivan Sutherland demonstrated by construction that asynchronous computing is not only feasible but also incredibly fast. Fred Brooks offered a personal history of computing by sharing his experiences with some of the giants in our field—it was as if the pages of a history book opened up. Butler Lampson reminded us there are principles for good system design: STEADY AID: <goals> simple, timely, efficient, adaptable, dependable, yummy and <methods> approximate, increment, iterate, indirect, divide (and conquer). Leonard Adleman led us through a fascinating exploration of Riemannian Surfaces and their properties in algebraic number theory. Peter Naur explored a synapse-state theory of the mind and its associative properties. Andy Yao drew attention to the growing potential of quantum computation. Leslie Valiant pondered when two mathematical functions are the same and used the concept of holographic transformations applied to computational complexity. Surprisingly, Valiant's talk reignited my personal interest in the graph equivalence problem and I spent several hours exploring this with some students over dinner.

I am looking forward to Heidelberg and Lindau in 2016. 

Vinton G. Cerf is vice president and Chief Internet Evangelist at Google. He served as ACM president from 2012–2014.

Copyright held by author.

a <http://www.heidelberg-laureate-forum.org/>

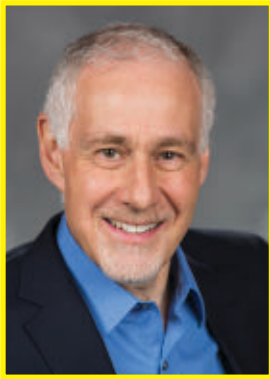
b <http://www.lindau-nobel.org/>

c <http://speyer.technik-museum.de/en/>

d <http://bit.ly/1NJicZd>

ACM

ON A MISSION TO SOLVE TOMORROW.



Dear Colleague,

Computing professionals like you are driving innovations and transforming technology across continents, changing the way we live and work. We applaud your success.

We believe in constantly redefining what computing can and should do, as online social networks actively reshape relationships among community stakeholders. We keep inventing to push computing technology forward in this rapidly evolving environment.

For over 50 years, ACM has helped computing professionals to be their most creative, connect to peers, and see what's next. We are creating a climate in which fresh ideas are generated and put into play.

Enhance your professional career with these exclusive ACM Member benefits:

- Subscription to ACM's flagship publication ***Communications of the ACM***
- Online books, courses, and webinars through the **ACM Learning Center**
- Local Chapters, Special Interest Groups, and conferences all over the world
- Savings on peer-driven specialty magazines and research journals
- The opportunity to subscribe to the **ACM Digital Library**, the world's largest and most respected computing resource

We're more than computational theorists, database engineers, UX mavens, coders and developers. Be a part of the dynamic changes that are transforming our world. Join ACM and dare to be the best computing professional you can be. Help us shape the future of computing.

Sincerely,

A stylized, handwritten signature in black ink, appearing to read 'Alex Wolf'.

Alexander Wolf
President
Association for Computing Machinery



Association for
Computing Machinery

Advancing Computing as a Science & Profession

SHAPE THE FUTURE OF COMPUTING.

JOIN ACM TODAY.

ACM is the world's largest computing society, offering benefits and resources that can advance your career and enrich your knowledge. We dare to be the best we can be, believing what we do is a force for good, and in joining together to shape the future of computing.

SELECT ONE MEMBERSHIP OPTION

ACM PROFESSIONAL MEMBERSHIP:

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

ACM STUDENT MEMBERSHIP:

- ☐ Student Membership: \$19 USD
- ☐ Student Membership plus ACM Digital Library: \$42 USD
- ☐ Student Membership plus Print *CACM* Magazine: \$42 USD
- ☐ Student Membership with ACM Digital Library plus
Print *CACM* Magazine: \$62 USD

- ☐ **Join ACM-W:** ACM-W supports, celebrates, and advocates internationally for the full engagement of women in all aspects of the computing field. Available at no additional cost.

Priority Code: CAPP

Payment Information

Name _____

ACM Member # _____

Mailing Address _____

City/State/Province _____

ZIP/Postal Code/Country _____

Email _____

Payment must accompany application. If paying by check or money order, make payable to ACM, Inc., in U.S. dollars or equivalent in foreign currency.

☐ AMEX ☐ VISA/MasterCard ☐ Check/money order

Total Amount Due _____

Credit Card # _____

Exp. Date _____

Signature _____

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

Return completed application to:
ACM General Post Office
P.O. Box 30777
New York, NY 10087-0777

Prices include surface delivery charge. Expedited Air Service, which is a partial air freight delivery service, is available outside North America. Contact ACM for more information.

Satisfaction Guaranteed!

BE CREATIVE. STAY CONNECTED. KEEP INVENTING.



Association for
Computing Machinery

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

Hours: 8:30AM - 4:30PM (US EST)
Fax: 212-944-1318

acmhelp@acm.org
acm.org/join/CAPP

Call for Nominations for ACM General Election

The ACM Nominating Committee is preparing to nominate candidates for the officers of ACM: **President, Vice-President, Secretary/Treasurer; and five Members at Large.**

Suggestions for candidates are solicited. Names should be sent by **November 5, 2015** to the Nominating Committee Chair, c/o Pat Ryan, Chief Operating Officer, ACM, 2 Penn Plaza, Suite 701, New York, NY 10121-0701, USA.

With each recommendation, please include background information and names of individuals the Nominating Committee can contact for additional information if necessary.

Vinton G. Cerf is the Chair of the Nominating Committee, and the members are Michel Beaudouin-Lafon, Jennifer Chayes, P.J. Narayanan, and Douglas Terry.



Ban 'Naked' Braces!

ONE FINE BUSINESS afternoon early in 1990, when we still used wires and microwave towers to make phone calls, and almost all long-distance calls went through big AT&T switches, one of the 100 or so 4ESS switches that handled U.S. long-distance traffic at the time hit a glitch and executed some untested recovery code. The switch went down briefly. No biggie, since traffic automatically took other routes, but in the process the initial switch that hit the glitch dragged its neighboring switches down, and the process cascaded across the country, as all the switches that handled long-distance traffic began to repeatedly crash and auto-recover. The result was that hardly any public telephone customer in the U.S. could make a long-distance phone call that afternoon, along with millions of dollars of time-sensitive business lost.

AT&T tried to contain the damage by rebooting the misbehaving switches, but as soon as a switch was brought back up, a neighboring switch would tell it to go down. The engineers at AT&T's R&D arm, Bell Labs, who wrote the switch programs, were called in, and, by the end of the day, network normality was restored by reducing the network message load.

An investigation was launched immediately, and after digging through a few hundred lines of code, word-of-mouth within Bell Labs was that the culprit was a closing brace (}) that terminated a selection construct—but the wrong one. The lawyers at Bell Labs quickly claimed such a lapse of human frailty could never be avoided entirely, and so dodged any potential lawsuits.

The lawyers were right; the intrinsic nature of software is such that the total absence of bugs is never guaranteed. But the simple practice of tagging all closing braces (or end in some languages) with a brief comment that indicates which construct they are closing would go far toward eliminat-

ing such an error; for example, instead of just writing '}' all by its naked self, write `///for, or ///if, or whatever.`

Tagging construct terminators can be done without changing existing compilers, and since such construct terminators usually appear on a line of code by themselves, the structure of the code is not affected. All this does is make the code easier to understand and helps prevent bugs like the one just described. This practice is especially helpful when code must be moved about, which happens often. In addition, if coders want to go one step further in making their code understandable, a brief comment can be added after the tag, like this

```
///for all transactions over a
thousand dollars
```

This would also eliminate the usefulness of putting the opening brace on a line by itself where it would be separated, from a syntactic viewpoint, from the construct it is punctuating, while creating an almost blank line that could better serve to separate logically distinct parts of a program.

I thus propose adoption of this practice by all software engineers and coders forthwith, as well as taught to all beginners from the get-go.

A. Frank Ackerman, Butte, MT

Surprisingly Deep Roots of Word Processor Interface Design

The Research Highlight "Soylent: A Word Processor with a Crowd Inside" by Michael Bernstein et al. (Aug. 2015) reminded me how long software developers have been pursuing such basic concepts as reducing redundancy and improving readability in computer-generated text. Soylent recruits volunteer humans via the Web, through a novel form of crowdsourcing, to accomplish what has long been a goal for natural language processing—improving readability and reducing redundancy in computer-produced text. Early work on auto-

mated abstracting, as in Betty Mathis et al.'s 1973 article "Improvement of Automatic Abstracts by the Use of Structural Analysis" in the *Journal of the American Society for Information Science*, demonstrated an algorithm that improved readability. Mathis et al. cited 18 even earlier works, including those covering algorithms showing how to shorten abstracts by removing redundant and/or unnecessary phrases. Their earliest citation was to a 1958 paper by IBM's Hans Peter Luhn "The Automatic Creation of Literature Abstracts" in the *IBM Journal of Research and Development*, demonstrating the deep roots of automated text generation.

Charles H. Davis, Bloomington, IN

CS Quantity Is Not CS Quality

Moshe Y. Vardi's Editor's Letter "Incentivizing Quality and Impact in Computing Research" (May 2015) was the first public acknowledgment I have seen of the problem of how to quantify quality in computer science research, as well as in applied computer science; that is, numbers alone do not determine quality. The belief in quantity-quality equivalence appears to have so permeated the computer science culture it is not uncommon to use quality numbers to cover real problems in research and software development. An instance I can cite from my own experience is the number of regression tests performed in software development despite the outcry from developers that most such tests add no value and in fact hinder development. I can only hope the realization of the problem of covering inferior research and practice with inflated numbers of published papers and software projects completed trickles down to the trenches of software development worldwide.

Raghavendra Rao Loka, Palo Alto, CA

Liability in Software License Agreements

Vinton G. Cerf's "Cerf's Up" column "But Officer, I was Only Programming at 100 Lines Per Hour!" (July 2013) asked for readers' views on how to address current software quality/reliability issues before legislative or

regulatory measures are enacted. The lion's share of the "persistent lack of software quality" problem lies not with software "professionals" but with business managers at software companies rushing to ship software well before it is ready for public consumption. There are few direct negative consequences for such decisions and far too many positive consequences, including the business mantra "First to market wins regardless of product quality."

I still see nothing to alter this bleak landscape until society as a whole becomes so fed up with the sad state of software it finally enacts laws making it illegal for software vendors to disclaim liability in their license agreements. Such drastic measures would have immediate consequences: Most vendors would go out of business rather than face the legal and financial music of their past transgressions; the price of software would instantly jump by a factor of 5 to 50; development and delivery schedules would expand; software prices would vary by customer, reflecting the liability risk posed by the customer; and, as always, lawyers would continue to win, even as their clients lose.

Many software developers would lose their jobs, but those among them able to design, structure, and implement software in a reliable manner would be in demand and earn much higher salaries, especially if the title "professional" meant they were personally liable for any possible failure of software they approved. However, much of the higher salary would go to cover "professional insurance" premiums.

In many jurisdictions, those in the licensed construction professions have the power and legal authority to deny their signatures when appropriate, halting construction until the related flaw is corrected, and management cannot legally circumvent the process.

How many software professionals wield such power over their own products? Until they have the authority, the primary problem for flawed software products will continue to reside outside the technical field of software development and computer science.

One hopes there would be a legal exception from liability for software that is free and/or open source. Freedom from liability could actually be an

incredible stimulus for the free/open-source software market.

David Warme, Annandale, VA

Whose Calendar?

In Leah Hoffmann's interview with Michael Stonebraker "The Path to Clean Data" (June 2015), Stonebraker said, "Turned out, the standard said to implement the Julian calendar, so that if you have two dates, and you subtract them, then the answer is Julian calendar subtraction." I surmise this was a lapsus linguae, and he must have meant the Gregorian calendar used throughout the former British Empire since 1752.

Marko Petkovšek, Ljubljana, Slovenia

Author's Response

I thank Petkovšek for the clarification. The two calendars are, in fact, different, and I meant the Gregorian calendar.

Michael Stonebraker, Cambridge, MA

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

© 2015 ACM 0001-0782/15/10 \$15.00

Coming Next Month in COMMUNICATIONS

Information Cartography

Why Do People Post Benevolent and Malicious Comments?

Rolling 'Moneyball' with Sentiment Analysis

Inductive Programming Meets the Real World

Fail at Scale

Componentizing the Web

Plus the latest news about algorithmic authors, solving the cocktail party problem, and employee-tracking technology.

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 will publish selected posts or excerpts.



Follow us on Twitter at <http://twitter.com/blogCACM>

DOI:10.1145/2811284

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

The Morality of Online War; the Fates of Data Analytics, HPC

John Arquilla considers justifications for warfare in the cyber realm, while Daniel Reed looks ahead at big data and exascale computing.



John Arquilla
"The Ethics of Cyberwar"

<http://bit.ly/1LFEU2g>

July 2, 2015

All over the world, there is a growing sense conflict is spreading from the physical realm to the virtual domain. The 2007 cyberattacks on Estonia, the military use of cyberwar techniques in the 2008 Russo-Georgian War, and the "cybotage" committed against Iran's nuclear program by the Stuxnet (<http://bit.ly/1KMCIo0>) worm are salient signs of a growing trend. These likely form the tip of an iceberg, as cyber attacks and counterattacks can be observed in many other places. It is high time, as this new mode of conflict diffuses in breadth and deepens in intensity, to think through the ethics of cyberwar.

Under what conditions should one engage in cyberwar? How should such a conflict be waged? These questions speak to the classical division in ethical thought about warfare that addresses the matter of going from peace to war justly, then ponders how to fight one's

battles honorably. In terms of going to war justly, there are three commonly held principles: *Right purpose*, which refers mostly to acting in self-defense; *Due authority* seeks authorization from a national or supranational body; and *Last resort*, which is self-explanatory. Ideas of fighting justly cluster around *Noncombatant immunity*, a focus on military vs. civilian targets, and *Proportionality*, avoiding excessive force.

Right purpose has always been a fraught element of just-war theory and practice. As Napoleon once said, "I had to conquer Europe to defend France." Many military adventures follow similar logic, justifying acts of aggression as preemptive or preventive defensive actions. Stuxnet would fall in the ethically dodgy area of prevention, and one can see how cyber attack may move nations in the direction of preemptive and preventive action. Not good.

Due authority, until the Information Age, was confined to nations, coalitions, or even transnational bodies like the United Nations. NATO made choices to intervene militarily in Kosovo in

1999, and in recent years in Libya. The U.N. authorized action to repel invading North Korean forces in 1950; and so on. This category includes and allows ethical choices to go to war made by individual nations—even when that choice might have been made in error (like the U.S.-led war against Iraq in 2003, whose justification was the mistaken belief Saddam Hussein had, or soon would have, weapons of mass destruction). In cyberwar, "due authority" suffers because armies, navies, and air forces are not necessary; just malicious software and skilled hackers. "Authority" loses meaning in a world where aggressive networks, or even highly adept individuals, can wage cyberwar.

Last resort typically has referred to a requirement to pursue diplomatic efforts until it is clear they will not resolve a given crisis. This aspect of just-war theory has also proved a bit nebulous, as sometimes war is resorted to because one or another party to a dispute just gets tired of negotiating. The July Crisis of 1914 that led to World War I falls in this category. The Japanese-American talks in 1941 were frustrating enough to Tokyo that the choice was made to attack Pearl Harbor before diplomatic talks ended. When it comes to cyberwar, its fundamentally covert, deniable nature may mean it will be used *during* negotiations—clearly the case with Stuxnet.

Noncombatant immunity is the principle to avoid deliberate targeting of civilians. Over the past century, it has been outflanked by technologies that allow the innocent to be struck directly, without prior need to defeat armed forces.

es protecting them. World War II saw deliberate burning of many cities—and nuclear attacks on civilians in Japan as soon as the atomic bomb became available. During the Korean War, virtually every building in Pyongyang was flattened, and a greater weight of bombs fell on North Vietnam in “the American War” than were dropped on Hitler’s Germany. How will this principle play out in an era of cyberwar? With far less lethal harm done to noncombatants, but no doubt with great economic costs inflicted upon the innocent.

Proportionality has proved less difficult to parse over the past century or so. By and large, nuclear-armed nations have refrained from using ultimate weapons in wars against others not so armed. Korea stayed a conventional conflict; Vietnam, too, even though the outcomes of both for the nuclear-armed U.S. were, in the former case an uneasy draw, in the latter an outright defeat. In cyberwar, the principle of proportionality may play out more in the type of action taken, rather than in the degree of intensity of the action. A cyber counterattack in retaliation for a prior cyber attack generally will fall under the proportionality rubric. When might a cyber attack be answered with a physically destructive military action? The U.S. and Russia have both elucidated policies suggesting they might respond to a “sufficiently serious” cyber attack by other-than-cyber means.

Classical ideas about waging war remain relevant to strategic and policy discourses on cyberwar. Yet, it is clear conflict in and from the virtual domain should impel us to think in new ways about these principles. In terms of whether to go to war, the prospects may prove troubling, as cyber capabilities may encourage preemptive action and erode the notion of “war” as a tool of last resort. When it comes to strictures against targeting civilians (so often violated in traditional war), cyberwar may provide a means of causing disruption without killing many (perhaps not any) civilians. Yet there are other problems, as when non-state actors outflank the “authority” principle, and when nations might employ disproportionate physical force in response to virtual attack.

In 1899, when advances in weapons technologies made leaders wary of the costs and dangers of war, a conference (<http://bit.ly/1KMCJZg>) was held at The

Hague to codify the ethics and laws of armed conflict, followed by another meeting on the same subject in 1907. Perhaps it is time to go to The Hague again, as a new realm of virtual conflict has emerged. Even if we cannot live up to ethical ideals that might be agreed upon in such a gathering, it is imperative the world community should make the effort. Now.



Daniel A. Reed
“Exascale Computing
and Big Data:
Time to Reunite”

<http://bit.ly/1SQ0X8w>
 June 25, 2015

In other contexts, I have written about cultural and technical divergence of the data analytics (also known as machine learning and big data) and high-performance computing (big iron) communities. I have called them “twins separated at both” (in <http://bit.ly/1M186kd> and <http://bit.ly/1IUkOSF>). They share technical DNA and innate behaviors despite superficial differences. After all, they were once united by their use of BSD UNIX and SUN workstations for software development.

Both have built scalable infrastructures using high-performance, low-cost x86 hardware and a suite of (mostly) open source software tools. Both have addressed ecosystem deficiencies by developing special-purpose software libraries and tools (such as SLURM (<http://bit.ly/1M18i32>) and Zookeeper (<http://bit.ly/1IUl3xl>) for resource management and MPI (<http://bit.ly/1E4Ij41>) and Hadoop (<http://bit.ly/1IHHR1b>) for parallelism), and both have optimized hardware for problem domains (Open Compute (<http://bit.ly/1DlipOT>) for hardware building block standardization, FPGAs (<http://bit.ly/1KMEFRs>) for search and machine learning, and GPU accelerators for computational science).

I have seen this evolution in both the HPC and cloud computing worlds. One reason I went to Microsoft was to bring HPC ideas and applications to cloud computing. At Microsoft, I led a research team (<http://bit.ly/1K179nC>) to explore energy-efficient cloud hardware designs and programming models, and I launched a public-private partnership between Microsoft and the National Science Foundation on cloud applications (<http://bit.ly/1hfZr1V>). Back in aca-

demia, I seek to bring cloud computing ideas to HPC.

Jack Dongarra and I co-authored an article for *Communications* on the twin ecosystems of HPC and big data and the challenges facing both. The article (<http://bit.ly/1If45X0>) examines commonalities and differences, and discusses unresolved issues associated with resilience, programmability, scalability, and post-Dennard hardware futures (<http://bit.ly/1Dlj1E3>). The article makes a plea for hardware and software integration and cultural convergence.

The possibilities for this convergence are legion. The algorithms underlying deep machine learning (<http://bit.ly/1gEXlsr>) would benefit from parallelization and data movement minimization techniques commonly used in HPC applications and libraries. Similarly, approaches to failure tolerance and systemic resilience common in cloud software have broad applicability to high-performance computing. Both domains face growing energy constraints on the maximum size of systems, necessitating shared focus on domain-specific architectural optimizations that maximize operations per joule.

There is increasing overlap of application domains. New scientific instruments and sensors produce unprecedented volumes of observational data, and intelligent *in situ* algorithms are increasingly required to reduce raw data and identify important phenomena in real time. Conversely, client-plus-cloud services are increasingly model-based, with rich physics, image processing, and context that depend on parallel algorithms to meet real-time needs.

The growth of Docker (<http://bit.ly/1IHHLI>) and containerized (<http://bit.ly/1DljqGL>) software management speaks to the need for lightweight, flexible software configuration management for increasingly complex software environments. I hope we can develop a unified hardware/software ecosystem leveraging the strengths of each community; each would benefit from the experiences and insights of the other. It is past time for the twins to have a family reunion. ■

John Arquilla is a professor at the U.S. Naval Postgraduate School. **Daniel A. Reed** is Vice President for Research and Economic Development, University Computational Science and Bioinformatics Chair, and professor of Computer Science, Electrical and Computer Engineering, and Medicine at the University of Iowa.

© 2015 ACM 0001-0782/15/10 \$15.00

VEE 2016

12th ACM SIGPLAN/SIGOPS international conference on

Virtual Execution Environments

Atlanta, GA April 2-3, 2016 with ASPLOS

Authors are invited to submit original papers related to virtualization across all layers of the software stack, from high-level language virtual machines down to the microarchitectural level. VEE 2016 accepts both full-length and short papers.

Abstract deadline: November 23, 2015

Paper deadline: November 30, 2015



Image: Courtesy of Chuck Koehler <https://www.flickr.com/photos/cokak/355135172/>, <https://creativecommons.org/licenses/by/2.0/>

General Chair

Vishakha Gupta-Cledat (Intel Labs)

Program Co-chairs

Donald Porter (Stony Brook University)

Vivek Sarkar (Rice University)



in cooperation with



<http://conf.researchr.org/home/vee-2016>

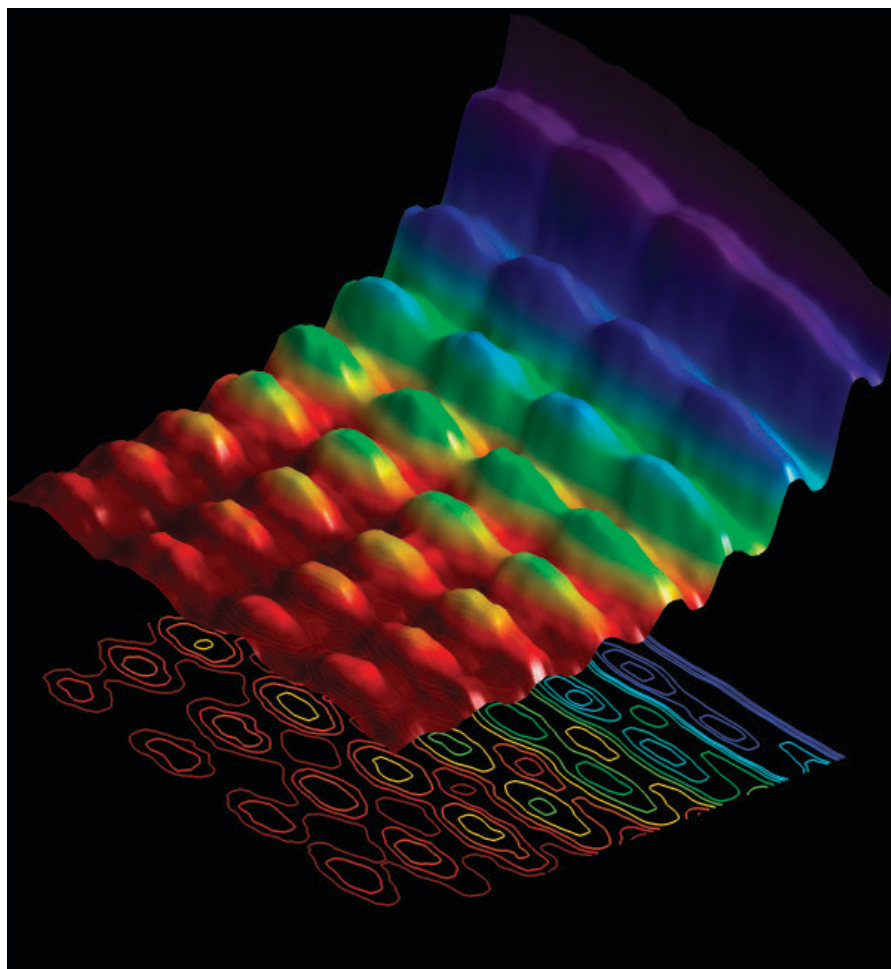
Scientists Update Views of Light

Experiment sheds new light on wave-particle duality.

THE DEBATE ABOUT whether light consists of waves or particles dates back to the 17th century. Early in the 20th century, Albert Einstein, Niels Bohr, and others exploring the world of quantum mechanics said light behaves as both waves and particles. Later experiments clearly showed this “wave-particle duality,” but they were never able to show light as both waves and particles at the same time.

Now, in a triumph of science and engineering at scales measured in nanometers and femtoseconds, international researchers have shown light acting as waves and particles simultaneously and continuously, and they have even produced photographic images of it. The scientists are from École Polytechnique Fédérale de Lausanne (EPFL) in Switzerland, Trinity College in Connecticut, and Lawrence Livermore National Laboratory in California.

The scientists fired intense femtosecond (fs) pulses of ultraviolet light at a tiny (40nm in diameter, 2 microns in length) silver wire, adding energy to charged particles on the wire that trapped the light in a standing wave along the surface of the wire. Then the researchers shot a beam of electrons close to the wire, and the electrons



The first-ever image of light behaving simultaneously as a particle and a wave.

interacted with the photons of light radiating around the wire. These electron-photon interactions either sped up or slowed down the electrons in an exchange of energy packets (quanta) between the particles. These quanta created images of the standing light wave that could be seen by an ultra-fast transmission electron microscope (UTEM), which can make videos at very high spatial resolutions.

After interacting with the photons traveling along the wire, the imaging electrons carry information about the exchange encoded in their spatial and energy distributions, explains EPFL's Fabrizio Carbone, the leader of the research team. These energy- and space-resolved images simultaneously show both the quantization of the light field (particles) and its interference pattern (waves). "For the first time, we can film quantum mechanics—and its paradoxical nature—directly," Carbone says.

The electromagnetic radiation on the nanowire is not light in the conventional sense, but a form of light called "surface plasmon polaritons" (SPP), or simply "plasmons," which exhibit all the properties—both classical and quantum—of light. Light striking a metal wire can produce these plasmonic fields as an electromagnetic wave that is coupled to free electrons in the metal and which travel along

"This is really an experimental tour de force, where you can visualize the beautiful plasmonic waves on these nano-needles."

the metal-air interface. These surface waves have a wavelength much shorter than the light that produces them, and can exist in extremely tiny spaces and move at far sharper angles than ordinary light on an optical fiber.

"This is really an experimental tour de force, where you can visualize the beautiful plasmonic waves on these nano-needles," says Herman Batelaan, a professor of physics at the University of Nebraska at Lincoln. "They use synchronous pulses of light and pulses of free electrons. The light hits the nano-needle, gets the electrons in the needle sloshing back and forth (the plasmonic wave), the pulse of electrons flies by the needle and their motion is affected by the electrons in the needle. The electrons that fly by

are then observed and they tell you what was going on in the needle. By changing the delay between light and free electron pulse, you can make a movie of the plasmonic wave."

The experiment neither contradicts nor extends the known laws of quantum mechanics, Batelaan says, "but this will certainly stimulate the discussion of what is particle-wave duality."

It also will make it easier to visualize that duality, Carbone says. The use of an experimental UTEM imaging system—one of just two femtosecond-resolved UTEMs in the world—is noteworthy because most electron microscopes only take snapshots, not time-resolved images (movies). "We design these kinds of circuits and then we induce these plasmons on them and we follow them as a function of time," he says.

Applications

The plasmons adhere very closely to the surface of the wire, even in complex geometries, making them especially suitable for use in tiny photonic circuits. "You can miniaturize [photonic] circuits in a very confined space using this property of guiding, and this offers an alternative to electronic circuits with faster switching and propagation," Carbone says. "The next step is to use materials other than simple metal, other materials of interest such

Milestones

Computer Science Awards, Appointments

BIOINFORMATICS LEADERS AWARDED DAN DAVID PRIZE

Leaders in bioinformatics recently received the Dan David Prize, a \$1-million award (which they shared) endowed by the Dan David Foundation and based at Tel Aviv University.

The Dan David Prize recognizes interdisciplinary research across traditional boundaries and paradigms in the past (fields that expand knowledge of former times), the present (achievements that shape and enrich society today), and the future (breakthroughs that hold great promise for improvement of our world).

The 2015 laureates for the future time dimension in the field of bioinformatics were

David Haussler, professor of biomolecular engineering and director of the Genomics Institute at the University of California, Santa Cruz; Michael Waterman, professor of biological sciences, computer science, and mathematics at the University of Southern California; and Cyrus Chothia, emeritus scientist at the MRC Laboratory of Molecular Biology in Cambridge, U.K.

The award for Retrieving the Past: Historians and their Sources was shared by historians Peter Brown and Alessandro Portelli, while the prize for the Present: The Information Revolution was presented to Jimmy Wales, cofounder of Wikipedia.

UC BERKELEY PROFESSOR WINS ACADEMY AWARD

University of California, Berkeley computer science professor James O'Brien received an Academy Award for scientific and technical achievement from the Academy of Motion Pictures Arts and Sciences.

O'Brien was recognized for his computer graphics research, which served as the foundation for systems that create fracture and deformation simulations. Software based on his research was used for films such as *Avatar*, *Prometheus*, *Skyfall*, *Harry Potter and the Deathly Hallows*, and *Guardians of the Galaxy*, among others.

O'Brien conducted research on simulations that assisted

in the development of the Kali Destruction System and the Digital Molecular Matter toolkit, systems that formed a way to model scalable and realistic fracture and deformation simulations.

When buildings are destroyed and broken apart in a movie, software based on O'Brien's research is used to determine how each building breaks.

He began his research on destruction simulations for his doctoral thesis at Georgia Institute of Technology's College of Computing, and continued this work when he began teaching at UC Berkeley in 2000.

O'Brien said he always had the film industry in mind when conducting his research.

as graphene or transition metal dichalcogenide monolayers.”

Indeed, SPPs are of great interest in fields such as communications and measurement, in applications including optical data storage, bio-sensing, optical switching, and sub-wavelength lithography. While Carbone’s work does not contribute directly to the science underlying these applications, the ability to both see and control what is going on at such tiny scales in space and time will likely be of interest to product developers and engineers.

“The technique employed enables the coupling of free electrons traveling at two-thirds the speed of light with electromagnetic fields to be spatially imaged on scales below the wavelength of light,” says David Flannigan, a professor of chemistry at the University of Minnesota. He said the technique’s ability to probe essentially any nanostructure geometry “allows for a clearer understanding of deviations from ideal behavior; for example, in the presence of impurities and morphological imperfections that are challenging to quantify and understand via other means. One could envision a number of ways this could be useful for real-world materials, systems, and device architectures.”

The success of the experiment using nanoscale wires and femtosecond time frames will be of interest to developers of tiny integrated circuits, Batelaan agrees. “They have gotten such beautiful control over what happens in the wire, and they can measure it probably better than anybody before.”

Batelaan points out today’s computer processors operate at speeds of a few GHz, “but when they are working in femtoseconds, orders of magnitude faster,” he says, “that could lead to completely new computer architectures.”

The experiment is controlled by 80fs laser pulses that produce 800fs electron pulses along the wire. “The buses linking the circuitry in a computer suffer higher loss if the frequency of the signal traveling in them is higher,” Carbone says. “Ultimately, beyond the GHz range, simple cable radiates like an antenna, thus losing signal when propagating an electromagnetic wave, especially when sharp corners or bends are made. Surface plasmons can circumvent this problem, although they suffer other types of losses in

“The significance of this experiment is that it takes a very different approach to a classical problem, opening a new perspective for its investigation.”

simple metal structures. So the hope is that new materials can support surface plasmons while having small propagation losses.”

The Double-Slit experiment

The wave-particle duality theories of the early 20th century were verified via a classic experiment in which light is projected onto a surface with two slits, which split the beam into two parts. The split beams are then measured, recombined, and measured again. Photon detectors behind each of the two slits show individual photons “choose” with equal probability to go one way or the other, showing light’s particle nature. In addition, the light beams when recombined produce the interference patterns characteristic of waves. The two measurements are performed one after the other, so the particle and wave states of light are not detected simultaneously.

Says Carbone, “The [split-beam] experiments show the paradox of quantum mechanics, and they show light is basically a superposition of both a wave and a particle until one decides to measure it.” The photon detector will say “particle,” but the interferometer will later say “wave.” “So the question was, ‘Is light somehow capable of adapting its behavior depending on the experiment being performed?’”

Until now, no one has performed an experiment that shows both natures of light occurring at the same time, he says. “The significance of this experiment is that it takes a very different approach to a classical problem, opening a new perspective for its investigation.”

Carbone says the experiment does not resolve an issue that arose between Einstein and Bohr: whether a single photon can act as both a wave and a particle at the same time. Carbone’s experiment considers small numbers of photons as a group, some of which behave as particles and some as waves, and its results are consistent with the known laws of quantum mechanics, he says. However, he says his research team is exploring the possibility of looking at the behavior of single electron-photon interactions. If that were to show wave-particle duality at the single photon level, that would violate the known laws of quantum mechanics, he says, but experimental data so far suggests that will not be the case.

Scientists agree the merit of this experiment lies not in new science revealed, but in greater insights about known phenomena and better ways to study them. “If you can see it, you can understand it better,” Carbone says. ■

Further Reading

Kocsis, S., et al.

Observing the average trajectories of single photons in a two-slit interferometer, *Science*, vol. 332, June 3, 2011, pp. 1170–1173 <http://bit.ly/1DEVegd>

Papageorgiou, N., Porchet, O., and Pousaz, L. Two-in-one photography: Light as wave and particle!

École polytechnique fédérale de Lausanne <https://www.youtube.com/watch?v=mlaVHxUSiNk>

Piazza, L., Lummen, T.T.A., Quiñonez, E., Murooka, Y., Reed, B.W., Barwick, B., and Carbone, F.

Simultaneous observation of the quantization and the interference pattern of a plasmonic near-field, *Nature Communications*, March 2, 2015. <http://bit.ly/1aPJ2p>

Piazza, L., Maisel, D.J., LaGrange, T., Reed, B.W., Barwick, B., and Carbone, F.

Design and implementation of a fs-resolved transmission electron microscope based on thermionic gun technology, *Chemical Physics*, Vol. 423, September 2013, pp. 79–84 <http://bit.ly/1yoxfl1>

Zia, R., Brongersma, M.

Surface plasmon polariton analogue to Young’s double-slit experiment, *Nature Nanotechnology* 2, published online: 1 July 2007 <http://bit.ly/1IatOcR>

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

© 2015 ACM 0001-0782/15/10 \$15.00

Automotive Systems Get Smarter

Automotive infotainment systems are driving changes to automobiles, and to driver behavior.

OVER THE LAST quarter-century, automobiles have evolved into increasingly sophisticated—and computerized—machines.

Today, some motor vehicles contain upward of 100 electronic control units with microprocessors that manage everything from steering and braking to navigation, climate control, and entertainment. They also have hundreds of millions of lines of software code. Overseeing the tangle of systems—and integrating buttons, knobs, voice commands and more—has emerged as a growing challenge, particularly as consumers carry smartphones into cars and look to integrate all these systems and controls seamlessly.

“There is a huge challenge associated with providing a driver with the right amount of information at the right time. You don’t want to overwhelm a driver or have someone get to the point where they are distracted or tuning out crucial information,” says Sam Abuelsamid, senior analyst on the Transportation Efficiencies Team at Navigant Research, which closely tracks automobile technologies. In recent years, auto manufacturers have introduced apps, speech recognition, and other systems, but often with limited success. “While these systems have delivered extra features to drivers, they’ve been limited in capabilities and the user interfaces have been relatively clunky,” he notes.

As a result, many consumers have thrown up their hands (but not while driving) and given up on using these systems. Instead, they prefer to tap into their smartphones and the simple, familiar interfaces they provide as the hub for infotainment and other functions. As John Maddox, assistant director of the Michigan Transportation Center at the University of Michigan, puts it:



Automotive infotainment systems provide drivers with a simplified interface to their vehicles.

“Consumers have become enamored by the breadth, variety, and timeliness of information they get on their phone, and they are now expecting this level of information in a vehicle. In some cases, they want the same display and the same choices built into their car.”

The upshot? As automobiles and computing roll forward and distracted driving becomes an ever-greater concern, automakers are looking for ways

“You don’t want to overwhelm a driver or have someone get to the point where they are distracted or tuning out crucial information.”

to integrate all these systems effectively and add advanced technology features, while Apple and Google are introducing infotainment platforms for vehicles. “We are moving past an era where features and capabilities have been thrown into cars, to a new environment that supports a connected lifestyle,” observes Mark Boyadjis, senior analyst and manager of infotainment and Human Machine Interface at automotive research and consulting firm IHS Automotive. “We will see a huge transformation in vehicles over the next few years.”

Beyond the Dashboard

Although GPS-based automobile navigation systems and other advanced technology features have been around since the early 1990s, a new era of automobile infotainment systems began around 2007, when Ford announced the first integrated, in-vehicle communications and entertainment system, SYNC. It allowed motorists to make hands-free phone calls with their cellular phones and to control music and other functions with specialized controls, including voice commands,

activated by tapping a button on the steering wheel. Over the next few years, other automobile makers introduced similar systems, typically built on Microsoft's Embedded Automobile System or BlackBerry's QNX software platform, which is used for critical systems such as air traffic controls, surgical equipment, and nuclear power plants.

Unfortunately, many of these early systems were difficult to use, and some relied on highly specialized and, at times, cryptic voice commands rather than natural language. In fact, J.D. Power reports the number-one disappointment of new car buyers is the voice recognition function. These systems also did not integrate well with iPods and emerging iPhones. Even with a built-in USB connection or Bluetooth connectivity, it was difficult, if not impossible, to view or control a music playlist or see information about a song, for example. In addition, these early systems could not pull contact information directly from a smartphone, making it necessary for a motorist to program in phone numbers and addresses manually.

By 2010, Ford had introduced AppLink and Chevrolet introduced MyLink—and other auto companies, including Audi and Volvo, soon followed suit with tighter integration with iPhones or similar controls accessible from a vehicle's LCD display or, in some cases, from a smartphone app. Yet, as Abuelsamid puts it: "These systems were a step forward, but consumers still found them confusing and clunky. There was a need for a platform that could tie together all the various tools, technologies, and other elements effectively."

In 2013, Apple introduced a new concept: an interface and software driver layer that runs on top of QNX and other real-time vehicle operating systems. Apple's CarPlay, and the subsequent introduction of Google's Android Auto, allow motorists to pair their mobile devices with a vehicle and view a simplified phone interface on the car's display screen, with a limited number of icons. "Anyone that is comfortable with the phone should be immediately comfortable with the interface," Abuelsamid explains.

For automakers, the appeal of CarPlay and Android Auto is that they essentially adapt to whatever vehicle they are

in. This might include a Mercedes with a non-touchscreen system and knob controls on the center console, a Ferrari with a resistive touchscreen interface, or a Volvo with a capacitive touchscreen interface. In every instance, the software translates the relevant hardware signals into a form the phone recognizes. Moreover, these platforms allow manufacturers to move away from proprietary systems and let consumers use either Android or iOS devices in their car—and even to switch between them. "It eliminates a basic problem: every car is different and it's difficult to operate a car you're not familiar with. It introduces a standard interface," Boyadjis says.

Convenience and happier motorists are not the only goals, however. According to the Virginia Tech Transportation Institute's Center for Automotive Safety, 80% of all crashes and 65% of all near-crashes involve a motorist looking away from the forward roadway within three seconds of the event. CarPlay and Android Auto aim to minimize driver distraction. For example, the phone's screen goes dark when the automobile is running, and these systems do not support social media or video. In addition, Android Auto has no "back" or "recents" buttons. Finally, both platforms offer better speech recognition through Siri and Google Now, which off-load processing to the cloud.

Says Jim Buczkowski, Henry Ford technical fellow and director for electrical and electronic systems in Ford's Research and Innovation Center, "A key is understanding what to process onboard and what to process in the cloud. The experience must be seamless and consistent, even when there isn't 100% cloud availability."

Driving Forward

Automotive infotainment systems are only part of the story, however. The J.D. Power 2015 U.S. Tech Choice Study found consumers increasingly seek technology that makes driving safer. Blind-spot detection and collision-avoidance systems, night vision, and other enhanced features ranked highest among desired technologies. Many high-end cars now include these features. Automakers are experimenting with head-up displays that project text and graphics on an area of the windshield. In addition, Texas Instruments is developing a pro-

ACM Member News

USING BIG DATA TO FIX CITIES



Juliana Freire is passionate about using big data analytics to solve real-world problems,

particularly those involving large urban centers like her Rio de Janeiro, Brazil, birthplace and her adopted hometown New York City.

"Data can make people's lives better," says Freire, a professor in the Department of Computer Science and Engineering at New York University (NYU). She has co-authored over 130 technical papers and holds nine U.S. patents. Her research focuses on large-scale data analysis, visualization, and provenance management involving urban, scientific, and Web data.

With her team in the Visualization, Imaging and Data Analysis Center at NYU's School of Engineering, Freire explores spatial temporal data, like energy and electricity consumption and traffic flow. She and the team work with New York City's Taxi and Limousine Commission to analyze real-time streaming data, like information about the 500,000 daily taxi trips that take place in that city. "We utilize predictive analysis to examine 'what-if' scenarios, like the cost-effectiveness of putting in a new subway line or a new bridge between Queens and Manhattan, and the potential impact on traffic patterns," she explains, adding, "We can take action in minutes or hours, instead of weeks or months."

Freire returns to Brazil annually to collaborate with researchers there on urban projects like bus usage in Rio de Janeiro. "They have amazing information about automobile movement because cameras are everywhere," she notes.

A proponent of "democratizing big data," Freire strives to create a virtual online facility "to house a structured urban data analysis search engine that's accessible to everyone," she says.

—Laura DiDio

jection system that uses digital light processing and interpolation methods to produce clear images across a windshield, even in poor weather or at night. The critical factor? “An HUD that displays information or alerts has to work with a quick glance and allow a person’s eyes to remain upward and forward,” Ford’s Buczkowski says.

Today, separate computerized systems in a vehicle typically use dedicated electronic controllers. Future automobiles will begin to combine and connect these systems, including GPS, cameras, radar, lidar, and more, Abuelsamid says. “They will be tied together through a vehicle network that will allow data sharing and introduce new and more advanced capabilities. This is a step toward automated driving systems.” General Motors has announced support for “Super Cruise” control in the 2016 Cadillac CT6; the technology will enable hands-free lane following and automatic braking and speed control during highway driving.

Critical to engineering these next-generation vehicles is embedding robust but highly secure communications systems. Researchers have already demonstrated the ability to hack into vehicles and take control of steering wheels and brakes. Informatics systems pose additional risks.

As a result, some auto manufacturers are now building Ethernet into vehicles in order to tie together all the various onboard systems in a more secure way. In addition, the automotive industry is developing a dedicated short-range wireless communications protocol called 802.11p, and some are

Some automakers are now building Ethernet into vehicles in order to tie together all the various onboard systems in a more secure way.

also building LTE cellular connectivity directly into vehicles. This makes vehicle-to-vehicle and vehicle-to-infrastructure communications possible, along with advanced certificate management and support for enhanced security features, including data encoding and encryption. Ford’s Buczkowski says this could ultimately lead to far more innovative features, including, for example, cars that can “see” around corners by communicating with other vehicles, and using their onboard systems to spot a cyclist or pedestrian. The network might also deliver an alert to the pedestrian through a smartwatch that vibrates or a smartphone that emits an alarm. “Mobility and cloud computing will play important roles in defining future driving experiences,” he says.

These communications capabilities will prove nothing less than transformative, Boyadjis says. Today, a two-year old car seems outdated, “but when you build a platform that allows infotainment sys-

tems and other onboard systems to update over the air, you enter an entirely different realm.” For instance, automaker Tesla has instantly updated more than 30,000 vehicles over the air. “In the future, it will be possible to add features and improve safety for power train, braking systems, steering controls, and other components through real-time software updates.” Adds Buczkowski: “Cars will add new features and address deficiencies or shortfalls based on customer feedback. It will likely be a very similar model as today’s smartphones.”

To be sure, greater technology integration will radically redefine the automobile and the driving experience over the next few years. In a decade, cars and their interiors may not resemble what we drive today. Concludes Abuelsamid: “We may at some point see reprogrammable touch interfaces that allow vehicle consoles and interfaces to appear the same way, regardless of the vehicle. We may see NFC tags that recognize you and adapt the car automatically. When you migrate to a software-based platform, all sorts of ideas become possible.” ■

Further Reading

Gharavi, H., Venkatesh, K., and Petros Ioannou, P. Scanning Advanced Automobile Technology, *Proceedings of The IEEE - PIIIEE*, vol. 95, no. 2, pp. 328-333, 2007, <http://1.usa.gov/1b7sFMO>

Alt, F., Kern, D., Schulte, F., Pfleging, B., Sahami Shirazi, A., and Schmidt, A. Enabling micro-entertainment in vehicles based on context information, *Proceedings of the 2nd International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, 2010. Pages 117-124. <http://dl.acm.org/citation.cfm?id=1969794>

Steinbach, T. Real-time Ethernet for automotive applications: A solution for future in-car networks, *Consumer Electronics - Berlin (ICCE-Berlin)*, 2011 IEEE International Conference, September 6-8, 2011, Pages 216-220. <http://bit.ly/1Efgbxf>

Huang, Y., Qin, G. H., Liu, T., and Wang, X. D. Strategy for Ensuring In-Vehicle Infotainment Security, *Applied Mechanics and Materials*, Vols. 556-562, pp. 5460-5465, May 2014. <http://www.scientific.net/AMM.556-562.5460>

Samuel Greengard is an author and journalist based in West Linn, OR.

© 2015 ACM 0001-0782/15/10 \$15.00



Recently, automaker Tesla remotely updated more than 30,000 vehicles at once.

Cyber Policies on the Rise

A growing number of companies are taking out cybersecurity insurance policies to protect themselves from the costs of data breaches.

THE CYBER ATTACKS carried out against Sony, Target, Home Depot, and J.P. Morgan Chase garnered a great deal of press coverage in 2014, but data breaches, denial-of-service attacks, and other acts of electronic malfeasance are hardly limited to large, multinational corporations. However, it is the high-profile nature of these breaches—as well as the staggering monetary costs associated with several of the attacks—that are driving businesses of all types and sizes to seriously look at purchasing cybersecurity insurance.

Currently, the global market for cybersecurity insurance policies is estimated at around \$1.5 billion in gross written premiums, according to reinsurance giant Aon Benfield. Approximately 50 carriers worldwide write specific cyber insurance policies, and many other carriers write endorsements to existing liability policies. The U.S. accounts for the lion's share of the market—about \$1 billion in premiums spread out across about 35 carriers, according to broker Marsh & McLennan, with Europe accounting for just \$150 million or so in premiums, and the rest of the world accounting for the balance of the policy value.

Due to strong privacy laws that have been enacted over the past decade, it is no surprise the U.S. is the leading market for cyber policies.

"The United States is many years ahead, due to 47 state privacy laws that require companies to disclose data breach incidents," says Christine Marciano, president of Cyber Data-Risk Managers LLC, a Princeton, NJ-based cyber-insurance broker. While notification may only cost a few cents per customer, large companies with millions of customers likely will be looking at



outlays of millions of dollars each time a breach occurs, a cost that could be covered by a cyber insurance policy.

The market for cyber insurance is projected to grow strongly, largely due to regulatory changes being enacted in jurisdictions around the globe. The Data Protection Directive (Directive 95/46/EC), which is being debated by the European Union and is expected to be ratified by 2017, spells out customer privacy and data-breach notification requirements. This type of regulation likely will bolster the cyber insurance market in Europe, which currently accounts for less than 10% of the global cyber insurance premiums written, according to Nigel Pearson, global head of Fidelity at Allianz Global Corporate & Specialty (AGCS), one of the world's largest insurance firms.

Pearson notes that in the U.K., the Information Commissioner (a government-level post established to uphold

information rights in the public interest) can fine companies up to about 500,000 pounds (about \$750,000) for failure to prevent a data breach, but with the EU reforms currently being discussed, the potential fines for data breaches are likely to be significantly higher, portending a greater need for insurance coverage. "Where those fines and penalties are insurable, we'll pay them," Pearson notes.

Marciano agrees, noting that "once the EU Data Protection reform reaches an agreement and is passed, the European cyber insurance market will see many new insurers offering cyber insurance policies, and many companies seeking coverage."

Pearson says the market continues to evolve in Asia as well, as jurisdictions such as Hong Kong and Australia introduce tougher privacy laws. The market for cyber insurance is "certainly evolving in Asia," Pearson says, noting that

“last year Hong Kong, Singapore, [and] Australia all had new data protection legislation. The big question is whether there is a requirement for mandatory notification.”

General Policies Fall Short

One of the key reasons businesses need to consider a cyber insurance policy or endorsement is that general liability coverage only covers losses related to a physical act, such as a person breaking in to an office and stealing files or computers. Cyber policies focus on so-called “intangible losses,” which are often not covered under general business liability policies, Marciano says.

“Many business liability policies that are coming up for renewal now contain clearly defined data breach exclusions, whilst most of the older policies did not clearly define such losses, and in some instances in which a claim arose, such policies were challenged,” Marciano says. “For those companies wanting to ensure they’re covered for cyber and data risk, a standalone cyber insurance policy should be explored and purchased.”

Damage caused by intrusions, attacks, or other losses must be covered by a specific cyber policy that generally covers three main activities or issues related to a cyber attack: liability, business interruption, and the cost of IT notification and forensics, according to Pearson. Furthermore, cyber policies typically offer both first-party coverage (covering the policyholder’s losses) and third-party coverage (covering defense costs and damages and liabilities to customers, partners, and regulatory agencies.)

First-party coverage includes the cost of forensic investigations, which include determining whether a data breach has occurred, containing the breach, and then investigating the cause and scope of the breach. Other coverage elements include the cost of computer and data-loss replacement or restoration costs, and the costs associated with interruption to the business (such as paying for alternative network services, employee overtime, and covering profits lost due to the data breach).

Other first-party costs often covered include the cost of public relations efforts to communicate appropriately to customers, business partners, and the press and general public, to try to pre-

General liability insurance covers losses related to a physical act, such as a person breaking into an office and stealing files or computers. Cyber policies focus on “intangible losses.”

vent and limit lost business. Notification costs, call center costs, and credit monitoring services for victims of the breach are also items that can be covered by cyber policies, and often represent a major portion of the overall cost of the breach, given that many companies have hundreds of thousands, if not millions, of individual customers to contact.

Finally, the cost of financial losses caused directly by electronic theft and fraud can be covered, as can the cost of cyber-extortion, in which criminals take control of a company’s Website or network, and refuse to relinquish control until a ransom is paid.

Third-party coverage will generally cover the cost to hire attorneys, consultants, and expert witnesses to defend a company from civil lawsuits by customers, business partners, and vendors harmed as a result of malware delivered via a compromised network, and shareholders (who may claim the value of their investment has been damaged as a result of the company’s failure to protect itself). Insurance may also be purchased to cover any settlements or judgments entered against the company. Additional third-party coverage can be purchased to cover the costs of regulatory or administrative agency investigations, prosecutions, and fines or penalties, though certain state or country laws may prohibit the coverage of such fines by insurance.

However, identifying the proper coverage levels, as well as securing a fair quote can be extremely challenging,

due to a relatively smaller pool of actuarial data, the evolving nature of cyber attacks or breaches, and the unwillingness of many carriers to share claims data, collectively make it challenging to craft standard cyber policies.

“Within cyber, it’s not unusual to have quotes that vary by multiples—sometimes 100%, 200%, 300% different,” Pearson says. “Companies are seeing the risks in very different ways, and are assessing the risk in very different ways.”

Nevertheless, according to January 2015 testimony before the U.S. Senate Committee on Homeland Security & Government Affairs by Peter J. Beshar, executive vice president and general counsel for the Marsh & McLennan Companies, the average cost for \$1 million of coverage is between \$12,500 and \$15,000 across industry sectors including healthcare; transportation; retail/wholesale; financial institutions; communications, media, and technology; education; and power and utilities.

According to news reports, the attack on Target cost that company \$148 million, along with an investment of \$61 million to implement anti-breach technology in the months after the attack. Meanwhile, Home Depot was expected to pay \$62 million to cover the cost of its attack, including legal fees and overtime for staff.

Before the breach occurred, Target carried at least \$100 million in cyber insurance. Home Depot had \$105 million in cyber insurance at the time of the attack, and Sony, hacked in December, carried a \$60-million policy. These policies helped offset some of the costs of the breaches, but not all, underscoring the need to ensure cyber policies’ coverage levels match the potential losses.

Limitations and Exclusions

However, there are limits to coverage. Cyber insurance does not cover losses due to terrorist acts or acts of war, and according to Marciano, few cyber policies cover physical injuries or damage caused by an attack that started online, but then caused actual physical damage in the real world, important issues businesses must consider when deciding on coverage levels.

“New threats and vulnerabilities are discovered daily, and it is hard to cover

every cyber incident, especially evolving risks we don't yet understand," Marciano says. "Insurers tend to be conservative on evolving risks until they have a better understanding of how to quantify and cover them." As such, individual company limits are determined based on factors such as company size, industry, revenues, services offered, types of data (such as whether personal identifiable information or personal health information is stored by the company), and, ultimately, how much the company can afford to purchase.

Still, understanding how much insurance to carry has been a struggle for many companies, says John Farley, Cyber-Risk Practice Leader for North American insurance brokerage HUB International. "You want to understand what type of data you hold, and what could cause you heartache if it's compromised," he says, noting that certain types of businesses are likely to be deemed to be a higher risk for insurers, and therefore likely will require higher coverage limits. Unsurprisingly, the companies and industries that likely face the largest cyber security threats are those that hold and use sensitive consumer information, including IT companies, financial services companies, retailers, higher education organizations, and healthcare firms, according to Farley.

"Healthcare and retail would be considered higher risk than manufacturing," Farley says, noting that companies that hold personal information, financial data, or health information are more likely to be targets for attackers than those companies that do not have data that can easily be re-sold or used by cyber criminals.

However, carriers and brokers note that practicing good "cyber hygiene" can help lower the cost of purchasing insurance, particularly if a company and its policies, systems, and practices can demonstrate a reduction in cyber risk.

Marciano defines cyber hygiene as "implementing and enforcing data security and privacy policies, procedures, and controls to help minimize potential damages and reduce the chances of a data security breach."

Marciano says processes should be put in place to protect against, monitor, and detect both internal and external threats, as well as to respond and recover from incidents. "Establishing

and enforcing policies and procedures, encrypting sensitive data at rest and in transit, being PCI compliant, adopting a security framework such as the NIST Cybersecurity Framework, and practicing good cyber hygiene can help companies obtain the most favorable cyber insurance premium."

Undergoing a network vulnerability assessment to determine strengths and weaknesses of a firm's IT infrastructure can help companies spot weaknesses before they can be exploited, allowing them to be corrected and then the firms can get coverage based on their tightened defenses.

The most important step a company can take is to ensure specific cyber coverage is already in place, and if not, to speak with a broker or carrier to obtain coverage, even if they believe their industry or business probably is not a target for hackers.

"The response we often get [from clients] is that 'I'm not Home Depot, I'm not Target, I'm not Chase, so the hackers aren't going to be after me,'" says Shawn Bernabeu, a business development manager with HUB International. "The hackers are continually going after smaller, not-so-well-known clients, and the fact of the matter is those smaller clients may not have the financial wherewithal to withstand and emerge from that hack and actually function." **C**

Further Reading

"Code Spaces forced to close its doors after security incident," *CSO*, June 18, 2014, <http://bit.ly/1KdGMg3>

Cyber Claims Examples, London Australia Underwriting, <http://bit.ly/1HxObZv>

Cybersecurity Framework, National Institute of Standards and Technology, <http://www.nist.gov/cyberframework/>

Cybersecurity In Demand, *Nightly Business Report*, March 17, 2015, https://www.youtube.com/watch?v=GS_HPiwhJWQ

Testimony of Peter J. Beshar, executive vice president and general counsel, Marsh & McLennan Companies, before the United States Senate Committee on Homeland Security & Governmental Affairs, Jan. 28, 2015 <http://1.usa.gov/1HcQSKX>

Keith Kirkpatrick is principal of 4K Research & Consulting, LLC, based in Lynbrook, NY.

© 2015 ACM 0001-0782/15/10 \$15.00

Education

ACM, CSTA Launch New Award

ACM and the Computer Science Teachers Association (CSTA) have launched a new award to recognize talented high school students in computer science.

The ACM/CSTA Cutler-Bell Prize in High School Computing program aims to promote computer science, as well as empower aspiring learners to pursue computing challenges outside of the classroom.

Four winners each year will be awarded a \$10,000 prize and cost of travel to the annual ACM/CSTA Cutler-Bell Prize in High School Computing Reception.

The prizes will be funded by a \$1-million endowment established by David Cutler and Gordon Bell. Cutler, Senior Technical Fellow at Microsoft, is a software engineer, designer, and developer of operating systems including Windows NT at Microsoft and RSX-11M, VMS, and VAXELN at Digital Equipment Corp. (DEC). Bell, researcher emeritus at Microsoft Research, is an electrical engineer and an early employee of DEC, where he led the development of VAX.

ACM President Alexander L. Wolf said the new award "touches on several areas central to ACM's mission," including "to foster technological innovation and excellence, in this case, by bringing the excitement of invention to students at a time in their lives when they begin to make decisions about higher education and career possibilities."

Said CSTA Executive Director Mark R. Nelson, "The Cutler-Bell Award celebrates core tenets of computer science education: creativity, innovation, and computational thinking. To encourage more students to pursue careers in computer science, to be America's next pioneers, we need intentional and visible attempts to increase awareness of what is possible. We expect the entries to the competition to set a high bar on what is possible with exposure to computer science in K-12."

The application period for the awards closes Jan. 1; inaugural awards will be announced in February 2016.

Inside Risks Keys Under Doormats

Mandating insecurity by requiring government access to all data and communications.

TWENTY YEARS AGO, law enforcement organizations lobbied to require data and communication services to engineer their products to guarantee law enforcement access to all data. After lengthy debate and vigorous predictions of enforcement channels going dark, these attempts to regulate the emerging Internet were abandoned. In the intervening years, innovation on the Internet flourished, and law enforcement agencies found new and more effective means of accessing vastly larger quantities of data. Today, we are again hearing calls for regulation to mandate the provision of exceptional access mechanisms.

In this column, a group of computer scientists and security experts, many of whom participated in a 1997 study of these same topics, explore the likely effects of imposing extraordinary access mandates. We have found the damage that could be caused by law enforcement exceptional access requirements would be even greater today than it would have been 20 years ago. In the wake of the growing economic and social cost of the fundamental insecurity of today's Internet environment, any proposals that alter the security dy-

The complexity of today's Internet environment means new law enforcement requirements are likely to introduce unanticipated security flaws.

namics online should be approached with caution. Exceptional access would force Internet system developers to reverse forward-secrecy design practices that seek to minimize the impact on user privacy when systems are breached. The complexity of today's Internet environment, with millions of apps and globally connected services, means new law enforcement requirements are likely to introduce unanticipated, hard-to-detect security flaws. Beyond these and other technical vulnerabilities, the prospect of globally

deployed exceptional access systems raises difficult problems about how such an environment would be governed and how to ensure such systems would respect human rights and the rule of law.

Political and law enforcement leaders in the U.S. and the U.K. have called for Internet systems to be redesigned to ensure government access to information—even encrypted information. They argue the growing use of encryption will neutralize their investigative capabilities. They propose data storage and communications systems must be designed for exceptional access by law enforcement agencies. These proposals are unworkable in practice, raise enormous legal and ethical questions, and would undo progress on security at a time when Internet vulnerabilities are causing extreme economic harm.

As computer scientists with extensive security and systems experience, we believe law enforcement has failed to account for the risks inherent in exceptional access systems. Based on our considerable expertise in real-world applications, we know such risks lurk in the technical details. In this column, we examine whether it is technically and operationally feasible to meet



law enforcement's call for exceptional access without causing large-scale security vulnerabilities. We take no issue here with law enforcement's desire to execute lawful surveillance orders when they meet the requirements of human rights and the rule of law. Our strong recommendation is that anyone proposing regulations should first present concrete technical requirements, which industry, academics, and the public can analyze for technical weaknesses and for hidden costs.

Many of this column's authors worked together in 1997 in response to a similar but narrower and better-defined proposal called the Clipper Chip.¹ The Clipper proposal sought to have all strong encryption systems retain a copy of keys necessary to decrypt information with a trusted third party who would turn over keys to law enforcement upon proper legal authorization. We found at that time it was beyond the technical state of the art to build key escrow systems at scale. Governments kept pressing for key escrow, but Internet firms successfully resisted on the grounds of the enormous expense, the governance issues, and the risk. The Clipper Chip was eventually abandoned. A much narrower set of

law-enforcement access requirements has been imposed in the U.S., but only on regulated telecommunications systems. Still, in a small but troubling number of cases, weaknesses related to these requirements have emerged and been exploited by state actors and others. Those problems would have been worse had key escrow been widely deployed. And if all information applications had to be designed and certified for exceptional access, it is doubtful that companies like Facebook and Twitter would even exist. Another important lesson from the 1990s is that the decline in surveillance capacity predicted by law enforcement 20 years ago did not happen. Indeed, in 1992, the FBI's Advanced Telephony Unit warned that within three years Title III wiretaps would be useless: no more than 40% would be intelligible and in the worst case all might be rendered useless.² The world did not "go dark." On the contrary, law enforcement has much better and more effective surveillance capabilities now than it did then.

The goal of this column is to similarly analyze the newly proposed requirement of exceptional access to communications in today's more complex, global information infrastructure. We

find it would pose far more grave security risks, imperil innovation, and raise difficult issues for human rights and international relations.

There are three general problems. First, providing exceptional access to communications would force a U-turn from the best practices now being deployed to make the Internet more secure. These practices include forward secrecy—where decryption keys are deleted immediately after use, so that stealing the encryption key used by a communications server would not compromise earlier or later communications. A related technique, authenticated encryption, uses the same temporary key to guarantee confidentiality and to verify the message has not been forged or tampered with.

Second, building in exceptional access would substantially increase system complexity. Security researchers inside and outside government agree that complexity is the enemy of security—every new feature can interact with others to create vulnerabilities. To achieve widespread exceptional access, new technology features would have to be deployed and tested with literally hundreds of thousands of developers all around the world. This is a far

more complex environment than the electronic surveillance now deployed in telecommunications and Internet access services, which tend to use similar technologies and are more likely to have the resources to manage vulnerabilities that may arise from new features. Features to permit law enforcement exceptional access across a wide range of Internet and mobile computing applications could be particularly problematic because their typical use would be surreptitious—making security testing difficult and less effective.

Third, exceptional access would create concentrated targets that could attract bad actors. Security credentials that unlock the data would have to be retained by the platform provider, law enforcement agencies, or some other trusted third party. If law enforcement's keys guaranteed access to everything, an attacker who gained access to these keys would enjoy the same privilege. Moreover, law enforcement's stated need for rapid access to data would make it impractical to store keys offline or split keys among multiple key holders, as security engineers would normally do with extremely high-value credentials. Recent attacks on the U.S. Government Office of Personnel Management (OPM) show how much harm can arise when many organizations rely on a single institution that itself has security vulnerabilities. In the case of OPM, numerous federal agencies lost sensitive data because OPM had insecure infrastructure. If service providers implement exceptional access requirements incorrectly, the security of all of their users will be at risk.

Our analysis applies not just to systems providing access to encrypted data but also to systems providing access directly to plaintext. For example, law enforcement has called for social networks to allow automated, rapid access to their data. A law enforcement backdoor into a social network is also a vulnerability open to attack and abuse. Indeed, Google's database of surveillance targets was surveilled by Chinese agents who hacked into its systems, presumably for counterintelligence purposes.³

The greatest impediment to exceptional access may be jurisdiction. Building in exceptional access would be risky enough even if only one law enforcement agency in the world had

... legislators should reject out of hand any proposal to return to the failed cryptography control policy of the 1990s.

it. But this is not only a U.S. issue. The U.K. government promises legislation this fall to compel communications service providers, including U.S.-based corporations, to grant access to U.K. law enforcement agencies, and other countries would certainly follow suit. China has already intimated it may require exceptional access. If a British-based developer deploys a messaging application used by citizens of China, must it provide exceptional access to Chinese law enforcement? Which countries have sufficient respect for the rule of law to participate in an international exceptional access framework? How would such determinations be made? How would timely approvals be given for the millions of new products with communications capabilities? And how would this new surveillance ecosystem be funded and supervised? The U.S. and U.K. governments have fought long and hard to keep the governance of the Internet open, in the face of demands from authoritarian countries that it be brought under state control. Does not the push for exceptional access represent a breathtaking policy reversal?

The need to grapple with these legal and policy concerns could move the Internet overnight from its current open and entrepreneurial model to becoming a highly regulated industry. Tackling these questions requires more than our technical expertise as computer scientists, but they must be answered before anyone can embark on the technical design of an exceptional access system. Absent a concrete technical proposal, and without adequate answers to the questions raised in this column, legislators should reject out of hand any proposal to return to the failed cryptography control policy of the 1990s. **C**

References

1. Abelson, H. et al. The risks of key recovery, key escrow, and trusted third-party encryption, 1997; <http://academiccommons.columbia.edu/catalog/ac:127127>.
2. Advanced Telephony Unit, Federal Bureau of Investigation. Telecommunications Overview, slide on Encryption Equipment, 1992; https://www.cs.columbia.edu/~smb/Telecommunications_Overview_1992.pdf.
3. Nakashima, E. "Chinese hackers who breached Google gained access to sensitive data, U.S. officials say." *The Washington Post* (May 20, 2013); <http://wapo.st/1MpTz3n>.

Harold "Hal" Abelson (hal@MIT.edu) is a professor of electrical engineering and computer science at MIT, a fellow of the IEEE, and a founding director of both Creative Commons and the Free Software Foundation.

Ross Anderson (Ross.Anderson@ccl.cam.ac.uk) is Professor of Security Engineering at the University of Cambridge.

Steven M. Bellovin (smb@cs.columbia.edu) is the Percy K. and Vida L.W. Hudson Professor of Computer Science at Columbia University.

Josh Benaloh is Senior Cryptographer at Microsoft Research where his research focuses on verifiable election protocols and related technologies.

Matt Blaze (blaze@cis.upenn.edu) is Associate Professor of Computer and Information Science at the University of Pennsylvania where he directs the Distributed Systems Lab.

Whitfield "Whit" Diffie is an American cryptographer whose 1975 discovery of the concept of public-key cryptography opened up the possibility of secure, Internet-scale communications.

John Gilmore (gnu@eff.org) is an entrepreneur and civil libertarian. He was an early employee of Sun Microsystems, and co-founded Cygnus Solutions, the Electronic Frontier Foundation, the Cypherpunks, and the Internet's alt newsgroups.

Matthew Green (mgreen@cs.jhu.edu) is a research professor at the Johns Hopkins University Information Security Institute. His research focus is on cryptographic techniques for maintaining users' privacy, and on new techniques for deploying secure messaging protocols.

Susan Landau (susan.landau@privacyink.org) is Professor of Cybersecurity Policy at Worcester Polytechnic Institute.

Peter G. Neumann (neumann@csl.sri.com) is Senior Principal Scientist in the Computer Science Lab at SRI International, and moderator of the ACM Risks Forum.

Ronald L. Rivest (rivest@mit.edu) is an MIT Institute Professor, and well known for his co-invention of the RSA public-key cryptosystem, as well for founding RSA Security and Verisign.

Jeffrey I. Schiller (jis@mit.edu) was the Internet Engineering Steering Group Area Director for Security (1994–2003).

Bruce Schneier is a security technologist, author, Fellow at the Berkman Center for Internet and Society at Harvard Law School, and the CTO of Resilient Systems, Inc. He has written a number of books, including *Data and Goliath: The Hidden Battles to Collect Your Data and Control Your World* (Norton, 2015).

Michael A. Specter (specter@mit.edu) is a security researcher and Ph.D. candidate in computer science at MIT's Computer Science and Artificial Intelligence Laboratory.

Daniel J. Weitzner (djweitzner@csail.mit.edu) is Principal Research Scientist at the MIT Computer Science and Artificial Intelligence Lab and Founding Director, MIT Cybersecurity and Internet Policy Research Initiative. From 2011–2012, he was U.S. Deputy Chief Technology Officer in the White House.

The full technical report MIT-CSAIL-TR-2015-026 from which this column has been derived is available at <http://dspace.mit.edu/bitstream/handle/1721.1/97690/MIT-CSAIL-TR-2015-026.pdf>.

Copyright held by authors.



Technology Strategy and Management

In Defense of IBM

The ability to adjust to various technical and business disruptions has been essential to IBM's success during the past century.

IBM'S CURRENT FINANCIAL results have made the news again—relatively good profits but flat or declining revenues for the past five years as well as a stagnant stock price.³⁻⁶ Rather than dismiss this historic company (founded in 1911) as an obsolete tech titan, however, I find myself instead appreciating what IBM has achieved over the past 100 years as well as thinking about what it might do in the future. IBM has struggled to grow but has also demonstrated the ability to navigate through multiple technological and business disruptions. These include mechanical punch-card tabulators to electromechanical calculators and then mainframes, personal computers, complex software programs, and now “cloud-based” services of almost magical sophistication, like the Watson artificial intelligence system that won the 2011 “Jeopardy!” game show.^a

There are many accounts of IBM's history, so I will not attempt to relate all the details here.^{1,b} However, most important to appreciate the modern company takes us back to 1993, when IBM appointed a new CEO, Louis Gerstner, who joined an organization that had just recorded the largest corporate loss in history—nearly \$9 bil-

Should we always judge the value of a company simply on sales growth and profit? Maybe not.

lion. IBM still dominated mainframes but that business was shrinking. The company had successfully launched a personal computer in 1981 but lost control over the new platform business to Microsoft and Intel. Gerstner's predecessor, John Akers, responded by laying off approximately 100,000 employees and devising a plan to split up the company into more than a dozen firms. Instead, IBM's board of directors hired Gerstner, and he decided to keep the company together but change the strategy.^c

IBM's mainframe business faced a major disruption not only from the personal computer, a mass-market product that produced much smaller profit margins. Within a year or so,

Gerstner also had to deal with the Internet and the World Wide Web—another historic disruption that would eventually offer a lot of software and services *for free*. To his credit, Gerstner saw the Internet less as a threat and more as a new opportunity. He understood that large customers faced challenges similar to what he had experienced at RJR Nabisco and American Express—how to combine the new technologies with the old systems. He settled on using professional services—IT consulting around “e-business” as well as system customization, integration, maintenance, and outsourcing—to help large customers pull together hardware and software for mainframes, PCs, and the Internet.

Over the next 20 years, Gerstner and his successors, Sam Palmisano and Virginia Rometty, would continue on this path, adding other skills and new businesses, along with a much more responsive strategy and resource allocation process.² As the accompanying table shows, the structural changes they introduced have been dramatic. Hardware accounted for 49% of revenues in 1993 and only 11% in 2014. Services have grown from 27% to 61%, and software products from 17% to 27%. Annual revenues did stall at approximately \$100 billion over the past several years and even declined in 2014 by \$7 billion. Part of the reason is that, following Gerstner's lead, IBM has continued to

a See “Watson Computer Wins at Jeopardy”; <https://www.youtube.com/watch?v=Puhs2LuO3Zc>.

b See “IBM Centennial Film”; <http://www.youtube.com/watch?v=39jtNUGmd4>.

c Gerstner told his own story in L. Gerstner, *Who Says Elephants Can't Dance: Inside IBM's Historic Turnaround*. Harper Business, 2002.

IBM financial comparison, 1993 and 2013–2014.

	1993	2013	2014
Revenues (\$million)	\$62,716	\$99,751	\$92,793
Profit (before tax)	(\$8,797)	\$19,524	\$18,356
Gross Margin	39%	49%	50%
Employees (year-end)	256,207	431,212	379,592
Revenues/Employee	\$245,000	\$231,000	\$244,000
R&D/Sales	9%	6%	6%
SG&A/Sales	29%	19%	20%
Hardware as % of Revenues	49%	14%	11%
Hardware Gross Margin	32%	36%	40%
Software as % of Revenues	17%	26%	27%
Software Gross Margin	61%	89%	89%
Services as % of Revenues	27%	57%	61%
Services Gross Margin	31%	36%	37%

Note: SG&A refers to Sales, General, and Administrative Expenses.
Source: Calculated from IBM Form 10-K annual reports.

shed commodity businesses—the list now includes PCs, semiconductors, printers, storage equipment, low-end servers, and call centers. Yet the company still managed to generate more than \$18 billion in operating profits in 2014 on sales of under \$93 billion. Moreover, hardware, software, and services are all more profitable today than they were when Akers left the company in 1993.

IBM's biggest structural challenge today is that it has become so dependent on professional services, and these kinds of revenues are difficult to scale and automate. They grow approximately on a one-to-one ratio with headcount increases. In fact, in terms of revenues generated per employee, not adjusted for inflation, IBM employees are no more productive today than they were in 1993 (see the table here). Not surprisingly, IBM's market value (about \$170 billion in May 2015) is far behind Apple (\$750 billion), Microsoft (\$395 billion), Google (\$370 billion), and even Facebook (\$220 billion), and just ahead of Intel (\$160 billion).

Another reason for lagging sales productivity is that technology has become cheaper. Not only do we see this in hardware and software products but in maintenance and services. Software as a service (SaaS) and

cloud computing, as well as overseas development and service centers in low-wage areas such as in India, have reduced the need for lucrative maintenance and other technical services. These trends have brought down the total cost of enterprise computing and have meant less revenues for companies such as IBM.

Critics also point out that IBM has propped up the value of company shares through stock buybacks (\$108 billion worth since 2000) instead of investing in research and development at the level of other enterprise technology companies, or making big transformational acquisitions.⁷ (By comparison, Microsoft, Oracle, Google, and SAP generally spend 13% or 14% of revenues on R&D. Apple, because of its limited consumer product lines and rapid sales growth, only spends about 3% of sales on R&D.) For a company whose business is mainly services, though, IBM still spends a lot on R&D. And big R&D spending has not necessarily helped other companies like Microsoft and Intel grow faster than the enterprise computing market, which is increasing sales slowly compared to hot consumer product segments like smartphones and tablets, or even SaaS for small and medium-size enterprises.

But should we always judge the

value of a company simply on sales growth and profits? Maybe not. We are now moving into an era of exciting opportunities for new types of products and services that blend big data and “intelligent” analytics with massive computing power—precisely the combination of skills and technologies that few firms, other than IBM, possess within the same organization. One potential example of this combination is the application of IBM Watson to problems such as reducing healthcare costs, diagnosing diseases, minimizing pollution, or optimizing energy usage.

Gerstner's main contribution was to keep IBM as one company with a clear purpose—service the data processing needs of large organizations, public and private. Those customers often tackle enormously complex problems of value to business, government, and society. In the 1930s, for example, IBM built the information infrastructure for the U.S. Social Security system. In the 1950s and 1960s, it pioneered anti-missile defense software as well as airline reservation systems. Today, it is tackling new applications for artificial intelligence. IBM has always taken on the biggest information technology problems since its predecessor company first began making mechanical tabulators for census taking more than 100 years ago. I expect it will still be taking on society's most complex data processing and analysis problems 100 years from now. 

References

1. Cusumano, M. IBM: One hundred years of customer solutions. In *The Business of Software*. Free Press, New York, 2004, 97–108.
2. Harreld, J.B., O'Reilly III, C.A., and Tushman, M.L. Dynamic capabilities at IBM. *California Management Review* (Summer 2007), 21–43.
3. Langley, M. Behind Ginni Rometty's plan to reboot IBM. *The Wall Street Journal* (Apr. 20, 2015).
4. Lohr, S. IBM first quarter earnings top Wall Street expectations. *The New York Times* (Apr. 20, 2015).
5. Lohr, S. The nature of the IBM crisis. *The New York Times*, (Oct. 22, 2014).
6. Sommer, J. Apple won't always rule. Just look at IBM. *The New York Times* (Apr. 25, 2015).
7. Sorkin, A.R. The truth hidden by IBM's buybacks. *The New York Times* (Oct. 20, 2014).

Michael A. Cusumano (cusumano@mit.edu) is a professor at the MIT Sloan School of Management and School of Engineering and co-author of *Strategy Rules: Five Timeless Lessons from Bill Gates, Andy Grove, and Steve Jobs* (HarperBusiness, 2015).

Copyright held by author.



Kode Vicious

Storming the Cubicle

Acquisitive redux.

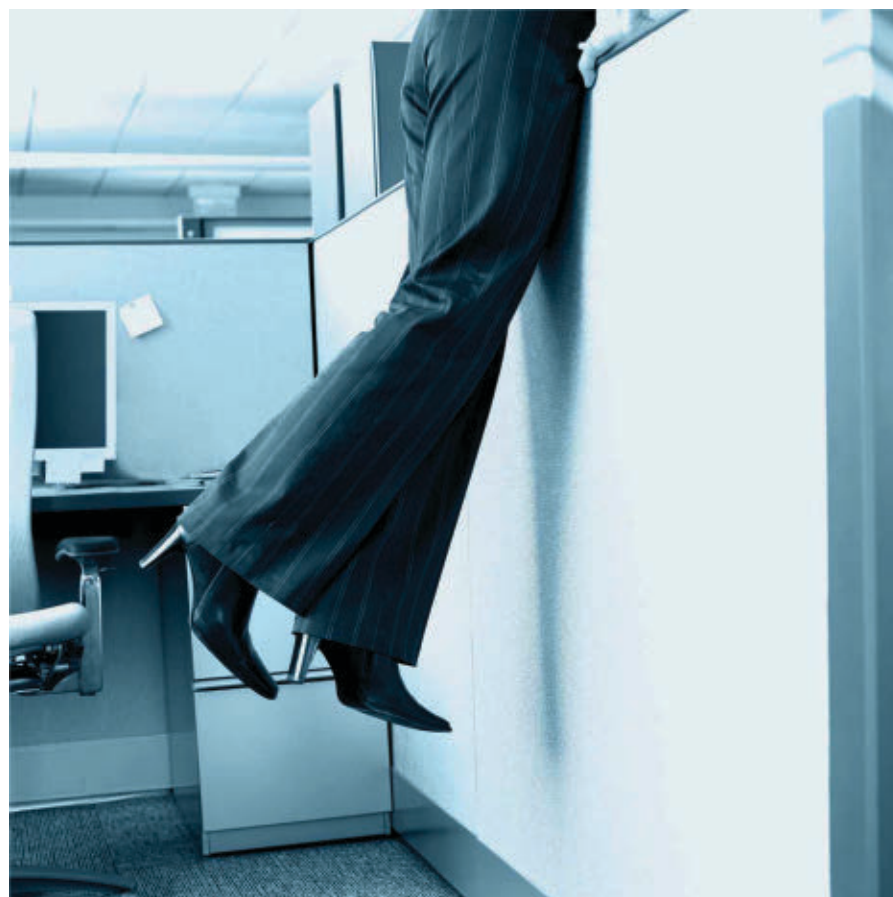
Dear KV,

I just signed on to a new project and started watching commits on the project's GitLab. While many of the commits seem rational, I noticed one of the developers was first committing large chunks of code and then following up by commenting out small bits of the file, with the commit message "Silence warning." No one else seemed to notice or comment on this, so I decided to ask the developer what kinds of warnings were being silenced. The reply was equally obscure—"Oh, it's just the compiler not understanding the code properly." I decided to run a small test of my own, and I checked out a version of the code without the lines commented out, and ran it through the build system. Each and every warning actually made quite a bit of sense. Since I'm new to the project, I didn't want to go storming into this person's cubicle to demand he fix the warnings, but I was also confused by why he might think this was a proper way to work. Do developers often work around warnings or other errors in this way?

Forewarned If Not Forearmed

Dear Forewarned,

Let me commend your restraint in not storming into this person's cubicle and, perhaps, setting it and the developer alight, figuratively speaking of course. I doubt I would have had the same level of restraint without being physically restrained. I am told screaming at developers is a poor way to motivate them, but this kind of behavior



definitely warrants the use of strong words, words I am not, alas, allowed to use here. But I commend to you George Carlin's "Seven Words You Can Never Say on Television"¹ as a good starting point. If you find that too strong you can use my tried-and-true phrase, "What made you think ..." which needs to be said in a way that makes it clear you are quite sure the listener did not, in fact, think at all.

Once upon a time compilers were notoriously poor at finding and flagging warnings and errors. I suspect there are readers old enough to have seen unhelpful messages such as, "Too many errors on one line (make fewer)," as well as remembering compilers where a single missing character would result in pages of error output, all of which was either misleading or wrong.

COMMUNICATIONS APPS

Access the latest issue, past issues, BLOG@CACM, News, and more.



Available for iPad, iPhone, and Android



Available for iOS, Android, and Windows



Association for Computing Machinery

There is a lesson here for both tool writers and tool users. If you write a tool that cries wolf too often then the users of that tool, in the absence of a new and better tool, will simply ignore the warnings and errors you output. Between warnings and errors, the latter are easier to get right, because the tool can, and should, stop processing the input and indicate immediately what the problem was. Communicating the problem is your next challenge. The error message I mentioned here came from a real, for-pay product sold by a company that went on to make quite a lot of money—it was not generated by some toy compiler created by a second-year college student. Looking back through previous Kode Vicious columns you will find plenty of commentary on how to write good log messages, but for tool writers, in particular those who write tools for other engineers, there are a couple of key points to keep in mind.

The first point is to be specific. Say exactly what was wrong with the input you were trying to process. The more specific your message, the easier it is for the user of the tool to address the problem and move on. Given that computer languages are complex beasts, being specific is not always easy, as the input received may have sent your compiler off into some very odd corners of its internal data structures, but you must try to maintain enough state about the compilation process to be able to make the warning or error specific.

The second point is even simpler: tell the consumer exactly where, down to the character in the file if possible, the error occurs. Older compilers thought the line was enough, but if you are looking at a function prototype with five arguments, and one of them is wrong, it is best if your tool says exactly which one is causing the issue, rather than making the rest of us guess. A blind guess on five arguments gives you a 20% chance, and if you think tool users do not have to guess blindly very often, then you are one of those engineers who never have to deal with random bits of other people's code.

If you want a good example of a tool that tries to adhere to the two points I have laid out, I recommend you look at Clang and the LLVM compiler suite.

Their errors and warnings are clearer and better targeted than any I have used thus far. The system is not perfect, but it beats other compilers I have used (such as gcc).

If you are a tool consumer you had better be quite sure of your knowledge of the underlying system so you can say, with better than 90% probability, that a warning you receive is a false positive. Some readers may not know this, but we programmers have a bit of an issue with hubris. We think we are modeling in our heads what the code is doing, and sometimes what we have in our heads is, indeed, a valid model. That being said, be prepared to be humbled by the tools you are using. Good tools, written by good tool writers, embody the knowledge of people who have spent years, and in some cases decades, studying exactly what the meaning of a code construct is and ought to be. Think of the compiler as an automated guru who is pointing you to a higher quality of code. There are certainly false gurus in the world, so it pays to pick a good one, because the false ones will surely lead you into a world of programming pain.

KV

Dear KV,

I saw your response to *Acquisitive* in the June 2015 *Communications*.³ I liked your response, but would have liked to see you address the business side.

Once the acquisition is completed, then *Acquisitive*'s company owns the software and assumes all of the associated business risks. So my due diligence on the code would have included ensuring the code in question was actually written by the engineers at

Given that computer languages are complex beasts, being specific is not always easy.

the other company or that it was free and open source software where the engineers were in compliance with the associated open source license. There is a risk that one or more of the engineers brought the code from a previous employer or downloaded it from some online source where the ownership of the code was uncertain. In short, management's request of Acquisitive should be seen not only as checking the functionality and quality of the code, but also protecting the company against litigation over the associated IP.

Moving up in an organization comes with the need to understand the business and management issues of that organization. Management's request of Acquisitive might also be seen as a test of whether he has the right business instincts to move higher than the "architect" role to which he was promoted. Someone with a good tech background and strong business knowledge becomes a candidate for CTO or other senior roles.

Business and Management

Dear Business,

You are quite right to point out the issues related to the provenance of the software that Acquisitive has to review and that this ought to also be on the list when reviewing code that will be reused in a commercial or even an open-source context. The number of developers who do not understand source code licensing is, unfortunately, quite large, which I have discovered mostly by asking people why they chose a particular license for their projects. Often the answer is either "I did a search for open source" or "Oh, I thought license X was a good default." There are books on this topic, as I'm sure you know, such as Lindberg² but it is very difficult to get developers to read about, let alone understand, the issues addressed in those books. But for those who want to be, or find themselves thrust into the role of Acquisitive, this type of knowledge is as important as the ability to understand the quality of acquired code. Anyone who thinks working through a ton of bad code is problematic has not been deposed by a set of lawyers prior to a court case. I am told it is a bit like be-

A basic understanding of copyright and licensing can go a long way, at least in asking the correct questions.

ing a soccer goal tender, but instead of the players (lawyers) kicking a ball at you, they are kicking you instead.

From a practical standpoint, I would expect Acquisitive to ask for the complete commit log for all the code in question. Rational developers—and there are some—will actually put in a code comment when they import a foreign library. They may even notify their management and legal teams, if they have them, about the fact they are using code from some other place. Very few large systems are cut from whole cloth, so the likelihood a system being reviewed contains no outside code is relatively small. Asking the legal team for a list of systems that have been vetted and imported should also be on Acquisitive's checklist, although it does require talking to lawyers, which I am sure he is inclined to do.

Harking back to the theme of the original letter, even with these pieces of information in hand, Acquisitive should not trust what they were told by others. Spot-checking the code for connections to systems or libraries that are not called out is laborious and time consuming, but, at least in the case of open source code, not insurmountable. Some well-targeted searches of commonly used APIs in the code will often sniff out places where code might have been appropriated. Many universities now use systems to check their students' code for cheating, and the same types of systems can be used to check corporate code for similar types of cheats.

A basic understanding of copyright and licensing can go a long way, at least in asking the correct questions. In open source we have two major

types of licenses, those that control the sharing of code and those that do not. The GPL family of licenses is of the controlled type; depending on the version of the license (LGPL, GPLv2, and GPLv3) the programmer using the code may have certain responsibilities to share changes and fixes they make to the code they import. The BSD family of licenses does not require the programmer using the code to share anything with the originator of the code, and is used only to prevent the originator from being sued. It is also important to verify that the license you see in the file has not been changed. There have been cases of projects changing licenses in derived code, and this has caused a number of problems for various people. A reasonable description of common open source licenses is kept at opensource.org (<http://opensource.org/licenses>), and I would expect Acquisitive to have looked that over at least a few times during the review.

Lastly, I am not a lawyer, but when I deal with these topics I make sure I have one on my side I trust, because the last thing I want to do is bring a knife to a gun fight.

KV

Related articles on queue.acm.org

Commitment Issues

George Neville-Neil

<http://queue.acm.org/detail.cfm?id=1721964>

Making Sense of Revision-control Systems

Bryan O'Sullivan

<http://queue.acm.org/detail.cfm?id=1595636>

20 Obstacles to Scalability

Sean Hull

<http://queue.acm.org/detail.cfm?id=2512489>

References

1. Carlin, G. Seven words you can never say on television. *Class Clown*. 1972; <https://www.youtube.com/watch?v=lqVLTJfYnik>.
2. Lindberg, V. 2008. *Intellectual Property and Open Source: A Practical Guide to Protecting Code*. O'Reilly. <http://shop.oreilly.com/product/9780596517960.do>.
3. Neville-Neil, G.V. Lazarus code. *Commun. ACM* 58, 6 (June 2015), 32–33; <http://cacm.acm.org/magazines/2015/6/187314-lazarus-code/abstract>.

George V. Neville-Neil (kv@acm.org) is the proprietor of Neville-Neil Consulting and co-chair of the ACM Queue editorial board. He works on networking and operating systems code for fun and profit, teaches courses on various programming-related subjects, and encourages your comments, quips, and code snips pertaining to his *Communications* column.

Copyright held by author.



The Business of Software Thinking Thoughts

On brains and bytes.

OVER THE LAST 15 years, through this column, I have been thinking out loud about what it means to consider software as a knowledge storage medium. Rather than a product in the traditional sense, software is better viewed as a container for the real product. What the customer buys and the user employs is the executable knowledge contained in the software. When that knowledge is complete, internally consistent, and properly maps onto a problem space, the software is valuable. When the knowledge is incomplete or contradictory the software can be difficult or even dangerous to use. Discovering a software bug is simply when a lack of knowledge is made manifest, its appearance signals an *epiphany of ignorance*—an event in time where something that is not known becomes obvious.

While we can consider software as a knowledge medium, perhaps we should also think of software as a *thought* medium—an extension of our cognitive processes. In fact, since software often contains things that are manifest not correct knowledge, it is really a place where we store our *thinking*, even if that thinking happens to be wrong.

So, given our increasing reliance on software to run the world, perhaps we should give some thought to thinking.

Why We May Think

To take a simple evolutionary view, species usually develop capabilities



that have some survival advantage. While most animals think, humans have a much higher degree of this capability. But why? We should avoid a teleological argument of the form: we ended up thinking because that is how we ended up. Or its corollary: if we had not developed thinking no one would be around to wonder how and why we ended up thinking. Not that these recursive views are not correct; they are just not very helpful.

The most obvious evolutionary advantage of enhanced thinking would be to give a more efficient way to deal

with the world. Thinking has other functions: social cooperation, the ability to plan and forecast and the like. But if the foundational advantage is to better deal with the outside world then thinking should be closely aligned with the senses. It is through our senses that we experience the world, so it makes sense that thinking would build on this. We get hints of this when people say things like: "... that idea stinks, but this idea looks better and it somehow feels right ..." Lakoff and Nuñez have made a compelling argument

for this with respect to mathematics¹ but it could serve for other thought disciplines.

Near and Far

We cannot easily understand or deal with things unless they are “close together” either physically or conceptually. Our brains are adept at identifying or even imposing relationships that connote similarity; it is one of the fundamental functions of the brain. In fact this “like” construct is essential to our ability to reason and we have done a good job of extending this function by building whole systems, such as algebraic mathematics or the Linnaean classification of living organisms, by collecting different things together based on (our perception of) their likeness.

The complexities of the constructs we have built for thinking, such as our ability to abstract ideas, make it appear we have moved a long way from our sense-driven cognition processes. But we still clump them together according to their proximity to like things. And we often refer to them using verbs based on our senses.

But these refer to what thinking *does*, not what thinking *is*. So what *is* it?

I Am, Therefore I Think I Am

A traditional view of thinking views knowledge as being resident in some *place*: this person knows how to play chess and that one does not. This company knows how to build widgets and that one does not. The simplistic locational view of brain function recapitulates this and assumes that physical parts of our brain store knowledge in some static and persistent form. Thinking, particularly recovery from memory, would then be the retrieval of knowledge from those places. It is a simple model and is how we have constructed most digital computers. But it is probably wrong.

Purple People Eaters

When we think of purple people who eat or are eaten the “static knowledge” view of the brain would imply that neurons that store the concept of “purple” and those that store the knowledge of “people” would somehow send purple and people messages to each other, to some central processing function, or

Perhaps we should also think of software as a *thought medium*—an extension of our cognitive processes.

to our consciousness. While the brain does have physical locations that specialize in processing certain kinds of information, there is no “purple” neuron, no “color” clump of neurons, and no specific area of the brain that deals with the knowledge of people, purple or otherwise.

Our knowledge of purple and of people and of everything else is likely stored all over the brain and it is stored dynamically not statically. The brain is an enormous network of connections along which signals are continuously traveling. The function of neurons is to amplify and pass on these signals not to store them for later use. These messages start before we are born and they end when we die. They are active when we are reading articles in *Communications* and when we are asleep.

Thought—conscious or unconscious—can be viewed as a self-sustaining fractal pattern of signals. Embedded in these patterns are sub-patterns that carry the knowledge of all the things we know and all the things we have known. The patterns continuously morph and refresh. Should they ever completely stop they would not restart. The knowledge carried by these patterns is like a radio signal imposed on a carrier in which is embedded many other signals.

Patterns Within Flows

The “strongest” of these patterns are our most conscious and intentional thoughts—those that are strong enough to be accessible to and recognized by the “consciousness” pattern. Our habits might also be strong patterns, though we may be quite unaware of them. Some patterns resemble other patterns and these simi-

Calendar of Events

October 3–7

CHI PLAY '15: The Annual Symposium on Computer-Human Interaction in Play, London, UK,
Sponsored: ACM/SIG
Contact: Anna L. Cox,
Email: anna.cox@ucl.ac.uk

October 9–12

RACS '15: International Conference on Research in Adaptive and Convergent, Prague Czech Republic,
Contact: Esmaeil S. Nadimi,
Email: esi@mmmi.sdu.dk

October 12–16

CCS'15: The 22nd ACM Conference on Computer and Communications Security, Denver, CO,
Sponsored: ACM/SIG,
Contact: Indrajit Ray,
Email: indrajit@cs.colostate.edu

October 18–21

PACT '15: International Conference on Parallel Architectures and Compilation, San Francisco, CA,
Contact: Kathy Yelick,
Email: kayelick@lbl.gov

October 19–23

CIKM'15: 24th ACM International Conference on Information and Knowledge Management, Melbourne VIC Australia,
Sponsored: ACM/SIG,
Contact: James Bailey,
Email: baileyj@unimelb.edu.au

October 22–23

ESEM '15: 2015 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, Beijing, China,
Contact: Guenther Ruhe,
Email: ruhe@ucalgary.ca

October 25–30

SPLASH '15: Conference on Systems, Programming, Languages, and Applications: Software for Humanity, Pittsburgh, PA,
Sponsored: ACM/SIG,
Contact: Jonathan Aldrich,
Email: jonathan.aldrich@cs.cmu.edu

INTERACTIONS



ACM's *Interactions* magazine explores critical relationships between people and technology, showcasing emerging innovations and industry leaders from around the world across important applications of design thinking and the broadening field of interaction design.

Our readers represent a growing community of practice that is of increasing and vital global importance.



To learn more about us, visit our award-winning website
<http://interactions.acm.org>

Follow us on Facebook and Twitter



To subscribe:
<http://www.acm.org/subscribe>

Association for
Computing Machinery



We would think of better ways to build software if we better understand how we think.

larities are themselves signals. Some signals are so weak they are almost gone. When they weaken further or are completely buried in other patterns they will be gone and we will have “forgotten.” Patterns can be made stronger by continually revisiting them as happens when we practice playing a musical instrument. Patterns that are very similar to others may become conflated over time and memories merge.

Pulling Patterns

Thought, like the Von Neumann architecture, uses much the same mechanisms for “data” as for “process”—for knowledge and how to access that knowledge. It is likely that some of these patterns are functional rather than factual. That is, they enable actions rather than store data; they are verbs rather than nouns. Some patterns are “retrieval patterns” that search other signals to see how similar they are and perhaps perform some organization on them. This organization may consist of:

- ▶ combining patterns where one is subsumed into another or they are merged—this is the “like” construct;
- ▶ comparing patterns to identify differences and similarities—which might be compared to other differences and similarities;
- ▶ patterns that organize other patterns rather like indexes;
- ▶ meta-patterns that set out patterns based on similarities and differences;
- ▶ meta-meta patterns, rather like this list; and
- ▶ hybrid patterns that hook together other pattern types (including hybrid patterns).

Consciousness

Consciousness, as a pattern that is more aware of itself (read: able to process) than other patterns, seems to be the thing that separates humans from animals. Animals think, but they do not appear to think about thinking. This introspection pattern is likely a main element of consciousness and thinking-about-thinking is evident in the very name of the modern human, which is *homo sapiens sapiens*.

Ontology Recapitulates Psychology

Software languages and designs appear to recapitulate brain function—in fact, it is difficult to see how they could be much different. We use proximity constructs in “modularization.” We have search patterns and indexes and “like” constructs we call inheritance, we push and pop data into our memory as onto a stack. We refresh using constructors and destructors. We have process and data, operators, and operands. This seems quite obvious. But if software is thought—even “bad” or “incorrect” thought—then the building blocks of thought must be the building blocks of software.

Cognitive Machine

Our most entrenched software mechanisms and constructs come, not from the outside world, but from the inside world. We do not have object classes and inheritance because the world is structured this way, we have them because we are structured this way. We would think of better ways to build software if we better understand how we think.

The first sentence on the first page of the first book I ever read about software development reads: “This book has only one major purpose—to trigger the beginning of a new field of study: ... the psychology of computer programming.”² I read it in 1972.

It is time to read it again, I think. 

References

1. Lakoff, G. and Nunez, R. *Where Mathematics Comes From: How the Embodied Mind Brings Mathematics Into Being*. Basic Books, 2001.
2. Weinberg, G.M. *The Psychology of Computer Programming*. Van Nostrand Reinhold, 1971.

Phillip G. Armour (armour@corvusintl.com) is a vice president at Applied Pathways LLC, Schaumburg, IL, and a senior consultant at Corvus International Inc., Deer Park, IL.

Copyright held by author.

Historical Reflections Computing Is History

Reflections on the past to inform the future.

WITH CLOUD, BIG data, supercomputing, and social media, it's clear that computing has an eye on the future. But these days the computing profession also has an unusual engagement with history. Three recent books articulating the core principles or essential nature of computing place the field firmly in history. Purdue University has just published an account of its pioneering effort in computer science.⁴ Boole, Babbage, and Lovelace are in the news, with bicentennial celebrations in the works. *Communications* readers have been captivated by a specialist debate over the shape and emphasis of computing's proper history.³ And concerning the ACM's role in these vital discussions, our organization is well situated with an active History Committee and full visibility in the arenas that matter.

Perhaps computing's highly visible role in influencing the economy, reshaping national defense and security, and creating an all-embracing virtual reality has prompted some soul searching. Clearly, computing has changed the world—but where has it come from? And where might it be taking us? The tantalizing question whether computing is best considered a branch of the mathematical sciences, one of the engineering disciplines, or a science in its own right remains unsolved. History moves to center stage according to Subrata Dasgupta's *It Began with Babbage: The Genesis of Computer Science*.¹

^a Downloads exceed 114,000 for Thomas Haigh's Historical Reflections column "The Tears of Donald Knuth," *Commun. ACM* 58, 1 (Jan. 2015), 40–44, as of August 26, 2015.

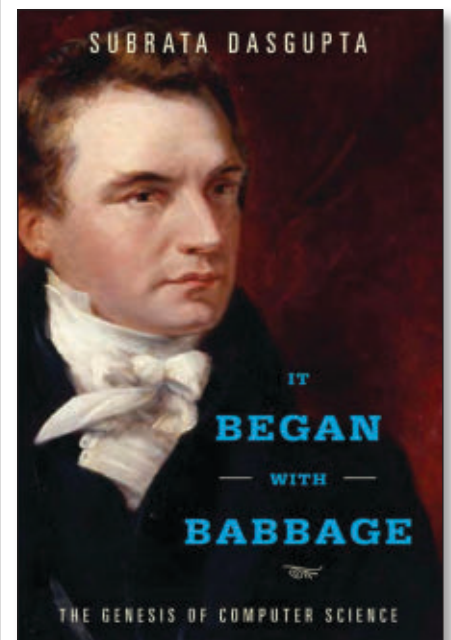
Turing's complex legacy is of enhanced importance today with the expansion of the A.M. Turing Award.

Dasgupta began his personal engagement with history in conversation with Maurice Wilkes and David Wheeler. Babbage, Lovelace, Hollerith, Zuse, Aiken, Turing, and von Neumann, among others, loom large in his pages.

Two recent books further suggest that computing is historically grounded. Peter Denning and Craig Martell's *Great Principles of Computing*² builds on Denning's 30-year quest to identify and codify "principles" as the essence of computing. The authors readily grant the origins of the Association for Computing Machinery, initially coupled to the study and analysis of computing machines. In their perspective on computing as science, they approvingly quote Edsger Dijkstra's quip "computer science is no more about computers than astronomy is about telescopes." Dijkstra and others in the founding generation closely connected to studies in logic, computability, and numerical analysis naturally saw computing as a mathematical or theoretical endeavor and resisted a focus on engineering questions and technological manifes-

tations. Similarly, Denning and Martell look beyond the 42 ACM-recognized computing domains, such as security, programming languages, graphics or artificial intelligence, to discern common principles that guide or constrain "how we manipulate matter and energy to perform computations," their apt description of the field. For each of their six principles—communication, computation, coordination, recollection, evaluation, and design—historical cases and historical figures shape their exposition. Communication is Claude Shannon, Harry Nyquist, Richard Hamming. These are historical principles.

In *Great Principles* the closer the authors get to cutting-edge science, the less their findings resemble the science-fair model of hypothesis, data collection, and analysis. They start from



Distinguished Speakers Program

<http://dsp.acm.org>

Students and faculty can take advantage of ACM's Distinguished Speakers Program to invite renowned thought leaders in academia, industry and government to deliver compelling and insightful talks on the most important topics in computing and IT today. ACM covers the cost of transportation for the speaker to travel to your event.

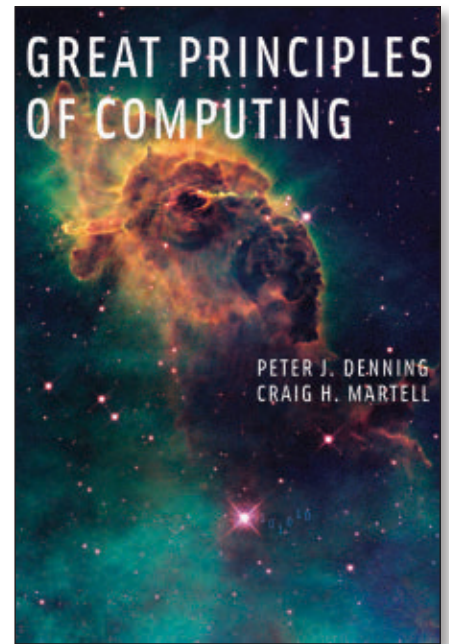


Association for
Computing Machinery

Dijkstra's view that "programming is one of the most difficult branches of applied mathematics." But programming is more than math. Programming languages from Fortran (1957) to Python (2000) are expressions of algorithms in an artificial language with its own syntax, often tailored for specific applications. Programmers with varied levels of skill work with compilers or interpreters, debugging tools, and version control as well as grapple with different means for avoiding errors. The practice of programming, however, is not cut-and-dried application of known laws. "Good programming is an artisan skill developed with good training and years of practice," they affirm.

Design as a core computing principle emerges from the authors' treatment of ENIAC and EDVAC in the 1940s through the information protection principles of Saltzer and Schroeder (1975) and forward to the design hints of Butler Lampson (1983). Judgment, intuition, and sense of history come to the fore. "Success of a design . . . depends on knowledge of history in the designer's field, which informs the designer on what works and what does not work." Design returns powerfully in their conclusion, which emphatically places "designers and their work at the center of the progress and innovation in computing." *Great Principles* does not stand apart from history; it embraces historical examples and historical thinking. And with design at its core, computing is history.

Matti Tedre's *The Science of Computing: Shaping a Discipline*⁵ examines three broad historical debates about the nature of computing: about computing as a distinctive theoretical field (starting in the 1930s), as an engineering field, and as a science in its own right. Tedre writes in the shadow of Denning's principles, with due tribute. His engagement with history is long and deep. Tedre sets up the pre-history in Leibniz, Boole, and Frege and closely examines the "decision problem" that animated Church and Turing, arriving at a surprising conclusion. He suggests, unmistakably, that "Turing's mathematical ideas had little if any influence on the invention of the modern computer." At Princeton in the mid-1930s the pieces were there—but they did not gel: Turing gives a seminar on his



just-published computable numbers paper, aided by Alonzo Church, but "there was rather bad attendance." With just two reprint requests, Turing despairs. And in a fellowship recommendation that von Neumann wrote for Turing in June 1937—just where you would expect a line about computability or decision problem—the great mathematician and soon-to-be namesake of von Neumann architecture praises instead Turing's "good work" in quasi-periodic functions! At this critical juncture Turing's influence on von Neumann is, at best, indirect and elusive.^b

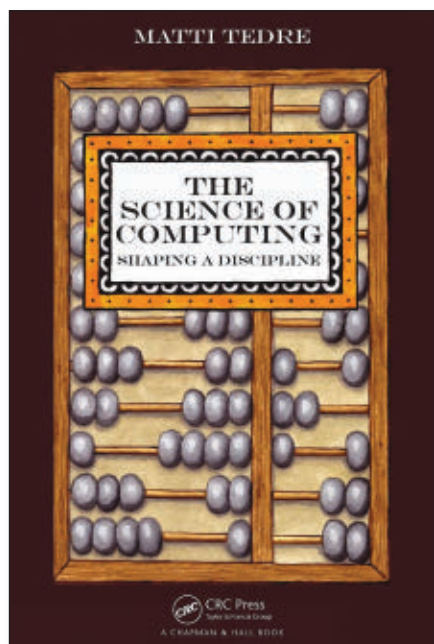
Tedre also closely examines the rival visions for "computer science" in the 1960s and the shifting emphases in ACM's model curricula. Three distinct debates engagingly frame the emerging scientific character of computing, including debates on formal verification, when advocates like C.A.R. Hoare (1985) sought to formally prove program correctness and create computing from axioms; on software engineering, which unsettled the theoretical and mathematical foundations of the pioneers; and on experimental computer science,

^b Andrew Hodges, *Alan Turing: The Enigma* (Simon & Schuster 1983), quotes "bad attendance," and "good work." Dasgupta¹ largely agrees (p. 58), then hedges (p. 113). By contrast, Martin Davis in *The Universal Computer* (2000) and George Dyson in *Turing's Cathedral* (2012) suggest a close connection between Turing and von Neumann.

which it seems everyone loved but no one quite practiced. Tedre gives a balanced treatment of each debate, attending to the intellectual and institutional dimensions, as people sought funding from the NSF, aimed at disciplinary identity, and struggled to create educational coherence. Computing emerges as a science, but there is no unfolding of a singular Newtonian paradigm.

Turing's complex legacy is of enhanced importance today with the expansion of the A.M. Turing Award, given for "major contributions of lasting importance to computing." The Turing Award recipients are dramatic personae for each of these books. Tedre, especially, heavily cites their contributions in *Communications*. The ACM History Committee, created in 2004, recently concluded a major re-vamping of the Turing Award website (<http://amturing.acm.org>). Michael R. Williams, professor emeritus at the University of Calgary, expanded the individual entries beginning with Alan Perlis in 1966, aiming at in-depth coverage for ACM members as well as accessible treatments that might spread the word. The History Committee has just launched a major oral-history initiative to ensure there are interviews with each of the 42 living Turing laureates, creating (where interviews are yet needed) a compelling video record.^c

c See ACM History Committee interviews at <http://history.acm.org/content.php?do=interviews>.



Clearly, computing has changed the world—but where has it come from? And where might it be taking us?

These oral histories, continued year by year, will complement the ongoing work on the Turing website, overseen now by Thomas Haigh.

The History Committee connects the ACM membership with professional historians of computing. Committee members represent research centers and museums, libraries and academic departments, industry and government laboratories, and varied ACM committees.³ Since 2009 the History Committee has supported 22 historical projects on ACM's storied history. So far the results include five completed Ph.D. dissertations, two published books, and a bevy of conference papers and other contributions. We responded to the ACM membership's curiosity about archival principles and methods with a workshop at the Charles Babbage Institute in May 2014.^d This month we will hold an ACM history workshop at the annual meetings of the Society for the History of Technology and the SIGCIS history of computing group.^e ACM members' interest in oral history methods and SIG-centered history are on the docket.

The computing-history gap that Donald Knuth was troubled by and that Thomas Haigh anatomized might be tractable.^f Despite the clear

d See "ACM History Committee Archiving Workshop" *ACM SIGSOFT Software Engineering Notes* <http://dl.acm.org/citation.cfm?doid=2693208.2693215> and http://history.acm.org/public/public_documents/ACM-archiving-workshop_2014-05.pdf.

e See http://www.historyoftechnology.org/features/annual_meeting/.

f See Thomas Haigh's column cited in footnote a and Martin Campbell-Kelly, "Knuth and the Spectrum of History," *IEEE Annals of the History of Computing* 36, 3 (July–Sept. 2014), 96.

challenges of doing professional history with rigorous computing content, we have evident successes. In her 2012 History Committee-supported Ph.D. dissertation ("Turing Award Scientists: Contribution and Recognition in Computer Science") Irina Nikiforova from Georgia Tech investigated intellectual and institutional patterns in which fields of computer science and which computer scientists were likely awardees. In another dissertation, completed in 2013 ("A House with the Window to the West: The Akademgorodok Computer Center (1958–1993)") Princeton's Ksenia Tatarchenko follows Andrei Ershov and his colleagues' efforts to build computer science in Soviet Russia and forge professional ties—across the "iron curtain"—to the ACM community. New York University's Jacob Gaboury's 2014 dissertation ("Image Objects: Computer Graphics at the University of Utah") investigates the prolific Evans and Sutherland network. Books done with ACM support are out from Cambridge University Press and forthcoming from ACM Books.^g In funding original research on ACM, as with enhanced publicity for the Turing awardees, we see many opportunities for constructive collaboration and professional dialogue in the years to come. ■

g With ACM funding Andrew Russell completed a set of interviews with European networking pioneers that led to his book *Open Standards and the Digital Age* (Cambridge University Press, 2014). ACM funding supported Bernadette Longo's biography of ACM founder: *Edmund Berkeley and the Social Responsibility of Computer Professionals* (ACM Books, forthcoming 2015).

References

1. Dasgupta, S. *It Began with Babbage: The Genesis of Computer Science*. Oxford University Press, 2014.
2. Denning, P. and Martell, C. *Great Principles of Computing*. MIT Press, 2015.
3. Hall, M. Understanding ACM's past. *Commun. ACM* 55, 12 (Dec. 2012), 5.
4. Pyle, R.L. *First in the Field: Breaking Ground in Computer Science at Purdue University*. Purdue University Press, 2015.
5. Tedre, M. *The Science of Computing: Shaping a Discipline*. CRC Press, 2015.

Thomas J. Misa (tmisa@umn.edu) is chair of the ACM History Committee.

Copyright held by author.

Viewpoint

Rise of Concerns about AI: Reflections and Directions

Research, leadership, and communication about AI futures.

DISCUSSIONS ABOUT ARTIFICIAL intelligence (AI) have jumped into the public eye over the past year, with several luminaries speaking about the threat of AI to the future of humanity. Over the last several decades, AI—automated perception, learning, reasoning, and decision making—has become commonplace in our lives. We plan trips using GPS systems that rely on the A* algorithm to optimize the route. Our smartphones understand our speech, and Siri, Cortana, and Google Now are getting better at understanding our intentions. Machine vision detects faces as we take pictures with our phones and recognizes the faces of individual people when we post those pictures to Facebook. Internet search engines rely on a fabric of AI subsystems. On any day, AI provides hundreds of millions of people with search results, traffic predictions, and recommendations about books and movies. AI translates among languages in real time and speeds up the operation of our laptops by guessing what we will do next. Several companies are working on cars that can drive themselves—either with partial human oversight or entirely autonomously. Beyond the influences in our daily lives, AI techniques are playing roles in science and medicine. AI is already at work in some hospitals helping physicians understand which patients are at



AI has been in the headlines with such notable advances as self-driving vehicles, now under development at several companies; Google's self-driving car is shown here.

highest risk for complications, and AI algorithms are finding important needles in massive data haystacks, such as identifying rare but devastating side effects of medications.

The AI in our lives today provides a small glimpse of more profound contributions to come. For example, the fielding of currently available technologies could save many thousands of

lives, including those lost to accidents on our roadways and to errors made in medicine. Over the longer-term, advances in machine intelligence will have deeply beneficial influences on healthcare, education, transportation, commerce, and the overall march of science. Beyond the creation of new applications and services, the pursuit of insights about the computational

foundations of intelligence promises to reveal new principles about cognition that can help provide answers to longstanding questions in neurobiology, psychology, and philosophy.

On the research front, we have been making slow, yet steady progress on “wedges” of intelligence, including work in machine learning, speech recognition, language understanding, computer vision, search, optimization, and planning. However, we have made surprisingly little progress to date on building the kinds of general intelligence that experts and the lay public envision when they think about “Artificial Intelligence.” Nonetheless, advances in AI—and the prospect of new AI-based autonomous systems—have stimulated thinking about the potential risks associated with AI.

A number of prominent people, mostly from outside of computer science, have shared their concerns that AI systems could threaten the survival of humanity.¹ Some have raised concerns that machines will become superintelligent and thus be difficult to control. Several of these speculations envision an “intelligence chain reaction,” in which an AI system is charged with the task of recursively designing progressively more intelligent versions of itself and this produces an “intelligence explosion.”⁴ While formal work has not been undertaken to deeply explore this possibility, such a process runs counter to our current understandings of the limitations that computational complexity places on algorithms for learning and reasoning. However, processes of self-design and optimization might still lead to significant jumps in competencies.

Other scenarios can be imagined in which an autonomous computer system is given access to potentially dangerous resources (for example, devices capable of synthesizing billions of biologically active molecules, major portions of world financial markets, large weapons systems, or generalized task markets⁹). The reliance on any computing systems for control in these areas is fraught with risk, but an autonomous system operating without careful human oversight and failsafe mechanisms could be especially dangerous. Such a system would not need to be particularly intelligent to pose risks.

The AI in our lives today provides a small glimpse of more profound contributions to come.

We believe computer scientists must continue to investigate and address concerns about the possibilities of the loss of control of machine intelligence via any pathway, even if we judge the risks to be very small and far in the future. More importantly, we urge the computer science research community to focus intensively on a second class of near-term challenges for AI. These risks are becoming salient as our society comes to rely on autonomous or semiautonomous computer systems to make high-stakes decisions. In particular, we call out five classes of risk: bugs, cybersecurity, the “Sorcerer’s Apprentice,” shared autonomy, and socioeconomic impacts.

The first set of risks stems from programming errors in AI software. We are all familiar with errors in ordinary software; bugs frequently arise in the development and fielding of software applications and services. Some software errors have been linked to extremely costly outcomes and deaths. The verification of software systems is challenging and critical, and much progress has been made—some relying on AI advances in theorem proving. Many non-AI software systems have been developed and validated to achieve high degrees of quality assurance. For example, the software in autopilot and spacecraft systems is carefully tested and validated. Similar practices must be applied to AI systems. One technical challenge is to guarantee that systems built via machine learning methods behave properly. Another challenge is to ensure good behavior when an AI system encounters unforeseen situations. Our automated vehicles, home robots, and intelligent cloud services must perform well even when they receive surprising or confusing inputs. Achieving such ro-

bustness may require self-monitoring architectures in which a meta-level process continually observes the actions of the system, checks that its behavior is consistent with the core intentions of the designer, and intervenes or alerts if problems are identified. Research on real-time verification and monitoring of systems is already exploring such layers of reflection, and these methods could be employed to ensure the safe operation of autonomous systems.^{3,6}

A second set of risks is cyberattacks: criminals and adversaries are continually attacking our computers with viruses and other forms of malware. AI algorithms are as vulnerable as any other software to cyberattack. As we roll out AI systems, we need to consider the new attack surfaces that these expose. For example, by manipulating training data or preferences and trade-offs encoded in utility models, adversaries could alter the behavior of these systems. We need to consider the implications of cyberattacks on AI systems, especially when AI methods are charged with making high-stakes decisions. U.S. funding agencies and corporations are supporting a wide range of cybersecurity research projects, and artificial intelligence techniques will themselves provide novel methods for detecting and defending against cyberattacks. For example, machine learning can be employed to learn the fingerprints of malware, and new layers of reflection can be employed to detect abnormal internal behaviors, which can reveal cyberattacks. Before we put AI algorithms in control of high-stakes decisions, we must be confident these systems can survive large-scale cyberattacks.

A third set of risks echo the tale of the Sorcerer’s Apprentice. Suppose we tell a self-driving car to “get us to the airport as quickly as possible!” Would the autonomous driving system put the pedal to the metal and drive at 125 mph, putting pedestrians and other drivers at risk? Troubling scenarios of this form have appeared recently in the press. Many of the dystopian scenarios of out-of-control superintelligences are variations on this theme. All of these examples refer to cases where humans have failed to correctly instruct the AI system on how it should behave. This is not a new problem. An important aspect of any AI system that interacts with people

is that it must reason about what people *intend* rather than carrying out commands literally. An AI system must analyze and understand whether the behavior that a human is requesting is likely to be judged as “normal” or “reasonable” by most people. In addition to relying on internal mechanisms to ensure proper behavior, AI systems need to have the capability—and responsibility—of working with people to obtain feedback and guidance. They must know when to stop and “ask for directions”—and always be open for feedback.

Some of the most exciting opportunities for deploying AI bring together the complementary talents of people and computers.⁵ AI-enabled devices are allowing the blind to see, the deaf to hear, and the disabled and elderly to walk, run, and even dance. AI methods are also being developed to augment human cognition. As an example, prototypes have been aimed at predicting what people will forget and helping them to remember and plan. Moving to the realm of scientific discovery, people working together with the Foldit online game⁸ were able to discover the structure of the virus that causes AIDS in only three weeks, a feat that neither people nor computers working alone could match. Other studies have shown how the massive space of galaxies can be explored hand-in-hand by people and machines, where the tireless AI astronomer understands when it needs to reach out and tap the expertise of human astronomers.⁷ There are many opportunities ahead for developing real-time systems that involve a rich interleaving of problem solving by people and machines.

However, building these collaborative systems raises a fourth set of risks stemming from challenges with fluidity of engagement and clarity about states and goals. Creating real-time systems where control needs to shift rapidly between people and AI systems is difficult. For example, airline accidents have been linked to misunderstandings arising when pilots took over from autopilots.^a The problem is that unless the human operator has been paying very close attention, he or she will lack a detailed understanding of the current situation and can make

poor decisions. Here again, AI methods can help solve these problems by anticipating when human control will be required and providing people with the critical information that they need.

A fifth set of risks concern the broad influences of increasingly competent automation on socioeconomics and the distribution of wealth.² Several lines of evidence suggest AI-based automation is at least partially responsible for the growing gap between per capita GDP and median wages. We need to understand the influences of AI on the distribution of jobs and on the economy more broadly. These questions move beyond computer science into the realm of economic policies and programs that might ensure that the benefits of AI-based productivity increases are broadly shared.

Achieving the potential tremendous benefits of AI for people and society will require ongoing and vigilant attention to the near- and longer-term challenges to fielding robust and safe computing systems. Each of the first four challenges listed in this Viewpoint (software quality, cyberattacks, “Sorcerer’s Apprentice,” and shared autonomy) is being addressed by current research, but even greater efforts are needed. We urge our research colleagues and industry and government funding agencies to devote even more attention to software quality, cybersecurity, and human-computer collaboration on tasks as we increasingly rely on AI in safety-critical functions.

At the same time, we believe scholarly work is needed on the longer-term concerns about AI. Working with colleagues in economics, political science, and other disciplines, we must address the potential of automation to disrupt the economic sphere. Deeper study is also needed to understand the potential of superintelligence or other pathways to result in even temporary losses of control of AI systems. If we find there is significant risk, then we must work to develop and adopt safety practices that neutralize or minimize that risk. We should study and address these concerns, and the broader constellation of risks that might come to the fore in the short- and long-term, via focused research, meetings, and special efforts such as the Presidential Panel on Long-Term AI Futures^b organized by the AAAI in 2008–2009 and the One Hundred

Year Study on Artificial Intelligence,^{10,c} which is planning centuries of ongoing studies about advances in AI and its influences on people and society.

The computer science community must take a leadership role in exploring and addressing concerns about machine intelligence. We must work to ensure that AI systems responsible for high-stakes decisions will behave safely and properly, and we must also examine and respond to concerns about potential transformational influences of AI. Beyond scholarly studies, computer scientists need to maintain an open, two-way channel for communicating with the public about opportunities, concerns, remedies, and realities of AI. **□**

^b See <http://www.aaai.org/Organization/presidential-panel.php>.

^c See <https://ai100.stanford.edu>.

References

1. Bostrum, N. *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press, 2014.
2. Brynjolfsson, E. and McAfee, A. *The Second Machine Age: Work Progress, and Prosperity in a Time of Brilliant Technologies*. W.W. Norton & Company, New York, 2014.
3. Chen, F. and Rosu, G. Toward monitoring-oriented programming: A paradigm combining specification and implementation. *Electr. Notes Theor. Comput. Sci.* 89, 2 (2003), 108–127.
4. Good, I.J. Speculations concerning the first ultraintelligent machine. In *Advances in Computers*, Vol. 6. F.L. Alt and M. Rubinfeld, Eds., Academic Press, 1965, 31–88.
5. Horvitz, E. Principles of mixed-initiative user interfaces. In *Proceedings of CHI '99, ACM SIGCHI Conference on Human Factors in Computing Systems* (Pittsburgh, PA, May 1999); <http://bit.ly/1OEyLFW>.
6. Huang, J. et al. ROSRV: Runtime verification for robots. *Runtime Verification*, (2014), 247–254.
7. Kamar, E., Hacker, S., and Horvitz, E. Combining human and machine intelligence in large-scale crowdsourcing. *AAMAS 2012* (Valencia, Spain, June 2012); <http://bit.ly/1h6gfbU>.
8. Khatib, F. et al. Crystal structure of a monomeric retroviral protease solved by protein folding game players. *Nature Structural and Molecular Biology* 18 (2011), 1175–1177.
9. Shahaf, D. and Horvitz, E. Generalized task markets for human and machine computation. *AAAI 2010*, (Atlanta, GA, July 2010), 986–993; <http://bit.ly/1gDIuho>.
10. You, J. A 100-year study of artificial intelligence? *Science* (Jan. 9, 2015); <http://bit.ly/1w664U5>.

Thomas G. Dietterich (tgdi@oregonstate.edu) is a Distinguished Professor in the School of Electrical Engineering and Computer at Oregon State University in Corvallis, OR, and president of the Association for the Advancement of Artificial Intelligence (AAAI).

Eric J. Horvitz (horvitz@microsoft.com) is Distinguished Scientist and Director of the Microsoft Research lab in Redmond, Washington. He is the former president of AAAI and continues to serve on AAAI’s Strategic Planning Board and Committee on Ethics in AI.

Copyright held by authors.



Watch the authors discuss their work in this exclusive *Communications* video. <http://cacm.acm.org/videos/rise-of-concerns-about-ai-reflections-and-directions>

^a See http://en.wikipedia.org/wiki/China_Airlines_Flight_006.

Viewpoint

Life After MOOCs

Online science education needs a new revolution.

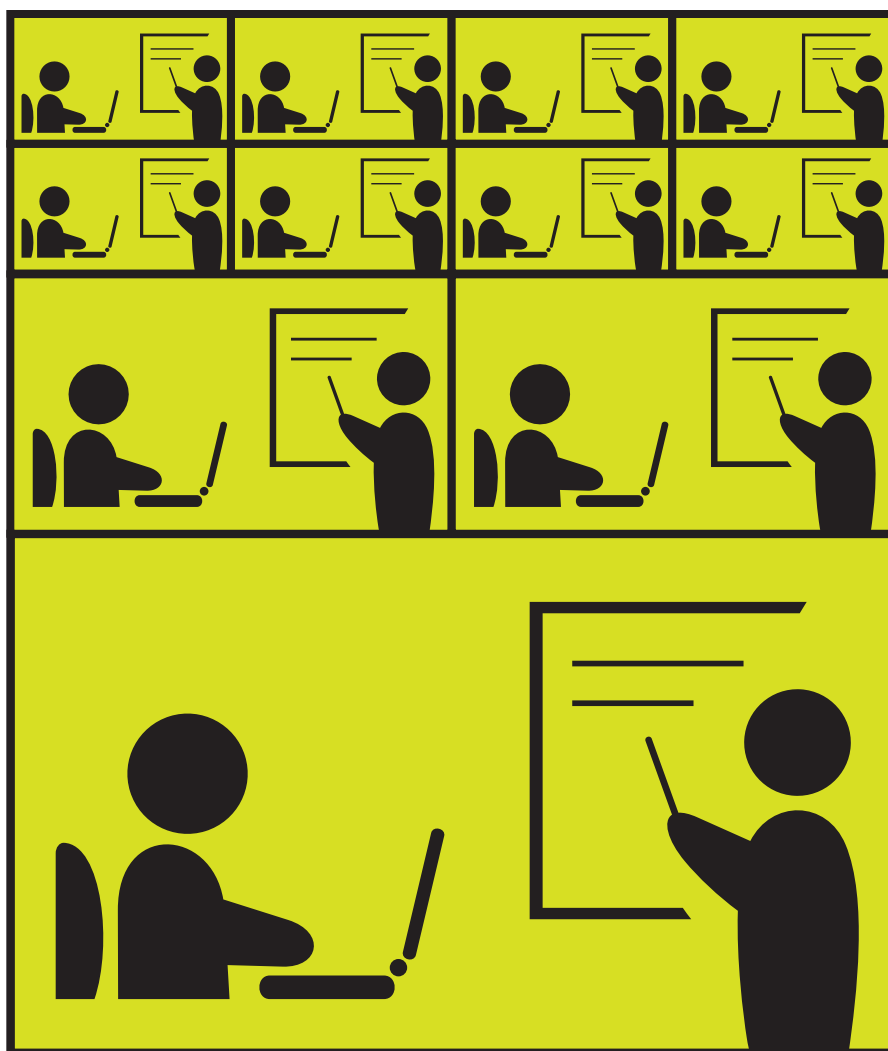
THREE YEARS AGO, Moshe Vardi published an editorial in *Communications* expressing concerns about the pedagogical quality of massive open online courses (MOOCs) and including the sentiment, “If I had my wish, I would wave a wand and make MOOCs disappear.”⁹ His editorial was followed by studies highlighting various limitations of MOOCs (see Karsenti⁵ for a review).

We share the concerns about the quality of early primitive MOOCs, which have been hyped by many as a cure-all for education. At the same time, we feel much of the criticism of MOOCs stems from the fact that truly disruptive scalable educational resources have not yet been developed. For this reason, if we had a wand, we would not wish away MOOCs but rather transform them into a more effective educational product called a massive adaptive interactive text (MAIT) that can compete with a professor in a classroom. We further argue that computer science is a discipline in which this transition is about to happen.

When Will Massive Open Online Courses Disappear?

Was the printing press a worthwhile invention? This may seem like a silly question, but some of the backlash against early MOOCs reminds us of a criticism of the printing press made by the prominent 15th-century polymath Johannes Trithemius. Believing printed books were inferior to hand-copied manuscripts, Trithemius wrote, “The printed book is made of paper and, like paper, will quickly disappear.”⁸

Anyone who has witnessed the



beauty of a Renaissance illuminated manuscript can sympathize with Trithemius. Likewise, anyone who has attended a lecture delivered by a brilliant teacher in a small classroom can sympathize with Vardi. Yet in reality, contemporary higher education often falls short of this ideal.

The Case for Radical Change in Science Education

Large universities continue to pack hundreds of students into a single classroom, despite the fact this “hoarding” approach has little pedagogical value.⁴ Hoarding is particularly objectionable in science, technol-

ogy, engineering, and mathematics (STEM) courses, where learning a complex idea is comparable to navigating a labyrinth. In the large classroom, once a student takes a wrong turn, the student has limited opportunities to ask a question in order to facilitate understanding, resulting in a learning breakdown, or the inability to progress further without individualized guidance.

A recent revolution in online education has largely focused on making low-cost equivalents of hoarding classes. These MOOCs, which are largely video-based, have translated all of the pedagogical problems with hoarding into an even less personal forum online. In other words, MOOCs have thus far focused on being *massive*, when they should strive to feel *individual*. Rather than reproducing the impersonal experience of listening to a professor's lecture in a large auditorium, online education should move toward replicating the experience of receiving one-on-one tutoring in the professor's office—the most productive (yet expensive) form of education.²

Furthermore, the majority of energy a student invests in a STEM course is spent outside of the classroom, reading a textbook and completing assignments. But the traditional textbook suffers from the same flaw as a large class in failing to address *individual* learning breakdowns. And although some publishers have recently founded projects aimed at developing truly interactive learning resources, results have been slow in coming.

Since universities and academic publishers have failed to address these shortcomings, we are calling for a second revolution in online education. This revolution will focus on the creation of MAITs, a new generation of interactive learning experiences for STEM fields that can adapt to learners' individual needs and simulate the experience of one-on-one education.

Our call for revolution may seem like a lofty proposal, but we believe the time is ripe for a number of reasons. First, the rise of MOOCs has already established a competitive online marketplace, in which only the most developed courses in a given STEM discipline will have a chance of long-term success. Second, large

Online education should move toward replicating the experience of receiving one-on-one tutoring.

investments are being made into sophisticated content platforms that can help improve upon the current video-based model. Third, a well-established research field is devoted to intelligent tutoring systems (ITSs), and next-generation electronic textbooks are already in development.^{1,7}

Efforts in ITS research have attempted to address certain inherent limitations of the traditional classroom, such as: most instructors teach to only a certain percentile of the class; most students do not receive the immediate feedback necessary to prevent learning breakdowns; and most instructors lack information about the many different learning breakdowns experienced by individual students. Yet despite the promise of ITSs, as Mazoue⁶ noticed, hardly any MOOCs have adopted ITSs. In light of the limited success of ITSs with the current generation of MOOCs, this Viewpoint defines a clear plan for how to make MOOCs truly disruptive by transforming them into MAITs.

What Is a MAIT?

A MAIT is defined by the following characteristics:

- ▶ Automated, individualized assessments;
- ▶ Interactivity;
- ▶ Adaptivity; and
- ▶ Modularity

Here, we illustrate these characteristics using our own experience in developing the *Bioinformatics* Specialization on Coursera, a series of six MOOCs followed by a Capstone Project^a accompanied by a textbook.³ In contrast to

initial ITS developments, which have largely aimed at entry-level courses, *Bioinformatics* is a series of complex interdisciplinary courses aimed at upper-level undergraduate and graduate students that covers algorithms, biology, and programming.^b

That we are MOOC developers may come as a surprise, since we have expressed doubts that MOOCs in their current form really represent a paradigm shift in STEM education. However, we see the creation of a MOOC as a natural first step toward producing a MAIT, and we are currently transitioning *Bioinformatics* toward a MAIT.

Automated, individualized assessments. When a student suffers a learning breakdown, that student needs immediate help in order to proceed. But traditional homework assignments are issued a week after the breakdown occurs. Teaching assistants (TAs) then must grade these assignments by hand, an undertaking that often proves repetitive. Furthermore, homework assignments are often unchanged year after year, and assignments at different universities have substantial overlap. Such a system makes no sense when grading in many STEM courses can be consolidated into a single automated system available at all universities.

In our call for automated assessments, we are not referring to primitive quizzes testing whether students are awake, but rather to robust assignments that require a sophisticated software system. Computer science is a unique discipline in that students' ability to program provides the opportunity to automatically check their knowledge through coding challenges. These coding challenges are far superior to traditional quizzes because, in order to implement a complex program, the student must possess a deep understanding of its underlying computational ideas.

Programming challenges already account for a significant fraction of assignments in many computer science courses such as introductory algorithms. However, thousands of computer science professors have implemented their own custom-made systems for grading student programs,

a See <http://coursera.org/specialization/bioinformatics/34>.

b https://www.youtube.com/playlist?list=PLQ-85lQlPqFM7jL47_tVFL61M4QM871Sv

an incredible illustration of academic inefficiency. A MAIT therefore promises to build a common repository of programming challenges and a user-friendly environment for learners, thus allowing professors and TAs to focus on teaching.

For example, in addition to our MOOC, we contributed to the development of Rosalind,^c a platform that automatically grades programming challenges in bioinformatics and allows a professor to form a customized Rosalind Classroom for managing assessments. In addition to Rosalind's 30,000 users, the Rosalind Classroom has been used over 100 times by professors wishing to incorporate its automated grading function into their offline courses. Grading half a million submissions to Rosalind has freed an army of TAs from the task of grading, thus saving time for interactions with students. Rosalind problems are individualized: the input parameters are randomly generated so no two students receive the same assignment.

Interactivity. A MAIT should incorporate elements of active learning. For example, *Bioinformatics* incorporates hundreds of “just in time” exercises and coding challenges that assess the student's progress at the exact moment this assessment is needed, facilitating the transition to the next topic. As such, *Bioinformatics* attempts to address learning breakdowns as soon as they occur.

A MAIT should also incorporate peer instruction, helping students interact with each other as well as with online TAs. If a learning breakdown persists after attempting an assessment, the student should be able to consult with peers who are having exactly the same breakdown. To achieve this goal, each paragraph of the interactive text powering *Bioinformatics* specialization is linked to a separate discussion forum.

Adaptivity. Most MOOCs incorporate elements of interactivity, but their educational materials are essentially static. In contrast, MAITs should be adaptive, an adjective that we apply in two distinct senses.

First, a MAIT should implement adaptive learning, meaning it can dif-

ferentiate students' responses and guide them through the material on individual learning paths according to these responses. Achieving true adaptive learning is the most challenging aspect of creating a MAIT, since it requires far more work than creating a textbook or MOOC.

Second, in order to achieve adaptive learning, the MAIT itself must be adaptive, meaning that its authors must be willing to change its content perpetually. This property is missing in most existing MOOCs because revising a video lecture (even to change a single sentence) is costly.

To make a MAIT adaptive, its authors should initially generate a compendium of learning breakdowns. We recently generated a compendium for *Bioinformatics* based on the analysis of 8,500 discussion forum posts. This compendium is a pedagogical gold mine that has helped us continually revise our course and eliminate many learning breakdowns.

Creating a compendium of learning breakdowns has also been an eye-opening experience. We never could have imagined our students' ability to catch every tiny logic error, every minor detail we had attempted to hide. At the same time, our students encountered many unpredictable, superficially implausible learning breakdowns. Most breakdowns only affected a small percentage of students but were made apparent by the scale of the MOOC.

After generating a compendium of learning breakdowns, a MAIT's authors should be willing to write many special adaptive modules, each one presented only to students with a specific breakdown. Unfortunately,

Adaptive learning is a particularly attractive feature of MAITs in interdisciplinary fields.

most current MOOCs are static, with limited changes introduced between consecutive offerings of the course. In our case, the creation of adaptive modules has nearly doubled the content needed for *Bioinformatics*. Assigning students to remedial modules should be done based on automated analysis of their responses, another important feature of a successful MAIT that will require future investment into data analysis.

Adaptive learning is a particularly attractive feature of MAITs in interdisciplinary fields. In these fields, students come from a variety of disciplines, and they often have gaps in their background and skills. In *Bioinformatics*, for example, biology, mathematics, and physics students typically lack knowledge of algorithms, whereas computer science students typically lack knowledge of statistics and biology. We have witnessed firsthand how automated assignments allow *Bioinformatics* students to succeed despite these gaps, but more work must be done to provide each student with an individual learning path through the course.

Modularity. Because the existence of a MAIT in a given field will likely flatten the textbook and MOOC markets in that field, some would rightly be concerned that a MAIT might lead to a rigid, standardized curriculum. To prevent this pitfall, MAITs should include an effort to modularize core content and provide resources for supplementing this content by additional crowdsourced learning modules.

An ancillary benefit of modularity is that a MAIT can serve as an educational hub for a community of educators. New professors teaching a subject for the first time can choose from an enormous menu of learning modules, while seasoned professors can contribute their own expertise to the growing project.

The Need for a High-Cost Development Team

Although professors creating new MOOCs often complain about the high cost of MOOC development, the cost of creating a MAIT will be much higher. We should cast aside the image of a professor on sabbatical writing a textbook or planning a new course from a

^c See <http://rosalind.info>.

café in some exotic locale. Instead, the production of a MAIT requires an entire development team with a budget of \$1 million or more.

Although this figure may seem preposterous, some educators, such as the developers of the Online Master of Science in Computer Science at Georgia Tech, have already invested comparable funds in developing their courses. MAITs should therefore be developed under the assumption that they have a sufficient budget in order to construct an educational product that can capture a large share of the MOOC market and truly disrupt both hoarding classes and traditional textbooks.

For example, *Bioinformatics* has already required over two years of development by a team consisting of professors, postdoctoral researchers, students, artists, and software engineers located in two countries and supported by three funding agencies and a private foundation. The total time investment made by this team was 50 times larger than the average of 100 hours required to develop a typical MOOC.⁵ The majority of development focused on creating an interactive text to power the course; lecture videos—which are often cited as a major investment in MOOC development—accounted for only a fraction of our budget. Yet *Bioinformatics* will require substantial additional investment in order to become a MAIT.

The high cost of MAIT development immediately raises the question of whether it makes sense to develop a million-dollar MAIT for small online courses, for example, attracting “just” 10,000 serious learners per year. We note that because of the rising costs of textbooks, a MAIT attracting just 10,000 learners per year indicates a potential educational market of over \$1 million per year. Furthermore, the high fixed cost of creating a MAIT is balanced by the negligible marginal cost of each additional learner. Finally, there are numerous opportunities to expand MAITs to developing countries, where the number of qualified professors is far smaller than the number of capable students.

The Future of MAITs

MAITs will eliminate the current model of hoarding classes practically

In looking for ways to improve our teaching, we found ourselves not looking forward, but backward, at the pedagogical style of Socrates.

overnight. Rather than attempting the futile task of creating a lecture that can be understood by hundreds of students from widely varying backgrounds, professors in hoarding classes will immediately see the inherent benefit in “flipping” these classes. In fact, some of our colleagues at leading universities have already used *Bioinformatics* to flip their classes. Rather than listening to lectures, students will complete assignments from the MAIT, which has already been fine-tuned to anticipate countless learning breakdowns. Energy the professor previously allocated to planning and delivering lectures can then be devoted to in-class discussions helping students understand complicated concepts, or even guided group projects that help them take the next steps.

Yet although we believe MAITs will first disrupt hoarding classes, we see MAITs as a disruptive technology to all STEM courses, both online and offline. Even the most talented teachers of small, offline courses may use MAITs to flip their courses when they realize that MAITs free them to imagine new ways to inspire their students.

Indeed, using the resources of a MAIT in an offline course does not just facilitate a professor’s transition toward a flipped classroom; it necessitates this transition. We observed this phenomenon in our own instruction of an offline course at the University of California, San Diego, which used the interactive text that powers *Bioinformatics*. Our flipped course blurred the boundary between instructor and

TA and forced us to completely rethink these roles. When students arrived in class, they already understood the majority of relevant course material. We would then help them answer each other’s questions about complicated concepts. We also divided students into small groups and guided them through additional challenge questions we had devised. As a result, class time was reinvested in direct interactions with students and group projects rather than preaching to them from a pulpit. It may sound like a strange way to run a course, but consider: Is this not the kind of educational experience students expect to receive when they enroll in a university?

We do not claim our flipped course has operated perfectly on its first attempts. However, its flaws have inspired us to become better educators in ways we never could have imagined. In looking for ways to improve our teaching, we found ourselves looking not forward, but backward, at the pedagogical style of Socrates. The irony has not been lost on us that our adoption of new technologies presented by online education forced our offline course to return to educational principles handed down from antiquity. **C**

References

1. Anderson, J.R. et al. R. Cognitive tutors: Lessons learned. *Journal of the Learning Sciences* 4 (1995), 167–207.
2. Bloom, B. The 2-Sigma problem: The Search for methods of group instruction as effective as one-on-one tutoring. *Educational Researcher* 13, 6 (1984), 4–16.
3. Compeau, P.E.C. and Pevzner, P.A. *Bioinformatics Algorithms: An Active Learning Approach*, Second ed. Active Learning Publishers, 2015.
4. Cuseo, J. The empirical case against large class size: adverse effects on the teaching, learning, and retention of first-year students. *The Journal of Faculty Development* 21, (2007), 5–21.
5. Karsenti, T. MOOCs: What the research says. *International Journal of Technologies in Higher Education* 10 (2013), 23–37; <http://bit.ly/1MPd8IH>.
6. Mazoue, J.G. Five myths about MOOCs. *Educause Reviews* (Sept.–Oct. 2013).
7. Miller, B.N. and Ranum, D.L. Beyond PDF and ePub: Toward an interactive textbook. In *Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education*, (2012), 150–155.
8. Trithemius, J. *De Laude Scriptorum (In Praise of Scribes)*. Klaus Arnold, Ed., Roland Behrendt. Tr. Colorado Press, 1974.
9. Vardi, M. Will MOOCs destroy academia? *Commun. ACM* 11, 5 (Nov. 2012), 5.

Phillip Compeau (pcompeau@cs.cmu.edu) is an assistant teaching professor in the Department of Computational Biology at Carnegie Mellon University, Pittsburgh, PA.

Pavel A. Pevzner (ppevzner@eng.ucsd.edu) is Ronald R. Taylor Chair Professor of Computer Science and Engineering in the Department of Computer Science and Engineering at the University of California at San Diego.

Copyright held by authors.

VRST 2015

The 21st ACM Symposium on Virtual Reality Software and Technology

<http://vrlab.buaa.edu.cn/vrst2015/>

The 21st ACM Symposium on Virtual Reality Software and Technology (VRST) is an international forum for the exchange of experience and knowledge among researchers and developers concerned with virtual reality software and technology. VRST will provide an opportunity for VR researchers to interact, share new results, show live demonstrations of their work, and discuss emerging directions for the field. The event is sponsored by ACM SIGCHI and SIGGRAPH.

VRST 2015 will be held in Beijing, the capital of China. From the magnificent Palace Museum, also known as the Forbidden City, to the beautiful Summer Palace and the Great Wall, Beijing is the political, economic and cultural center of China for over 800 years from the Yuan Dynasty. The numerous royal buildings with long history endow it with incomparable charm. On the other hand, as the host city of the 2008 Olympic Games, this oriental ancient city presented her best fashion fascination to the world. The conference will be hosted by China State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering in Beihang University (BUAA). VRST 2015 aims at bringing together VR researchers from around the world to present the state-of-the-art advances in this ever-growing dynamic area, and introducing VR research in China.

Important dates.

All deadlines are 15:59 UTC/GMT (Beijing time 23:59):

- **July 20th, 2015:** Abstract submission
- **July 27th, 2015:** Full/short papers submission
- **August 15th, 2015 :** Poster submission
- **September 8th, 2015:** Decisions announced
- **September 15th, 2015:** Camera-ready papers due
- **November 13th–November 15th, 2015:** Conference

Conference Chairs:

Qinping Zhao, Beihang University
Daniel Thalmann, Nanyang Technological University

Program Chairs:

Enhua Wu, University of Macau & Institute of Software,
Chinese Academy of Sciences
Ming C. Lin, University of North Carolina at Chapel Hill
Lili Wang, Beihang University

Local Chair:

Dangxiao Wang, Beihang University



Rethinking the fundamental abstractions of the file system.

BY T.S. PILLAI, V. CHIDAMBARAM, R. ALAGAPPAN,
S. AL-KISWANY, A.C. ARPACI-DUSSEAU, AND
R.H. ARPACI-DUSSEAU

Crash Consistency

THE READING AND writing of data, one of the most fundamental aspects of any von Neumann computer, is surprisingly subtle and full of nuance. For example, consider access to a shared memory in a system with multiple processors. While a simple and intuitive approach known as *strong consistency* is easiest for programmers to understand,¹⁴ many weaker models are in widespread use (for example, x86 total store ordering²²); such approaches improve system performance, but at the cost of making reasoning about system behavior more complex and error prone. Fortunately, a great deal of time and effort has gone into thinking about such memory models,²⁴ and, as a result, most multiprocessor applications are not caught unaware.

Similar subtleties exist in local file systems—those systems that manage data stored in your desktop computer, on your cellphone,¹³ or that serve as the underlying storage beneath large-scale distributed systems such as Hadoop Distributed File System (HDFS).²³

Specifically, a pressing challenge for developers trying to write portable applications on local file systems is *crash consistency* (that is, ensuring application data can be correctly recovered in the event of a sudden power loss or system crash).

Crash consistency is important. Consider a typical modern photo-management application such as iPhoto, which stores not only the photos a user takes, but also information relevant to a photo library, including labels, events, and other photo metadata. No user wants a system that loses photos or other relevant information simply because a crash occurs while the photo-management application is trying to update its internal database.

Much of the burden today in ensuring crash consistency is placed on the application developer, who must craft an *update protocol* that orchestrates modifications of the persistent state of the file system. Specifically, the developer creates a carefully constructed sequence of *system calls* (such as file writes, renames, and other file-system calls) that updates underlying files and directories in a recoverable way. The correctness of the application, therefore, inherently depends on the semantics of these system calls with respect to a system crash (that is, the *crash behavior* of the file system).

Unfortunately, while the standardized file-system interface has been in widespread use for many years, application-level crash consistency is currently dependent on intricate and subtle details of file-system behavior. Either by design or by accident, many modern applications depend on particular file-system implementation details and thus are vulnerable to unexpected behaviors in response to system crashes or power losses when run on different file systems or with different configurations.

Recent research, including work performed by our group at the University of Wisconsin–Madison,²¹ as well as elsewhere,²⁹ has confirmed that crashes are problematic: many applications



(including some widely used and developed by experienced programmers) can lose or corrupt data on a crash or power loss. The impact of this reality is widespread and painful: users must be prepared to handle data loss or corruption,¹⁵ perhaps via time-consuming and error-prone backup and restore; applications might tailor their code to match subtle file-system internals, a blatant violation of layering and modularization; and adoption of new file systems is slowed because their implementations do not match the crash behavior expected by applications.⁶ In essence, the file-system abstraction, one of the basic and oldest components of modern operating systems, is broken.

This article presents a summary of recent research in the systems community that both identifies these crash consistency issues and points the way toward a better future. First a detailed example illustrates the subtleties of the problem. We summarize the state of the art, illustrating the problems we (and others) have found are surprisingly widespread. Some of the promising research in the community aims to remedy these issues, bringing new thinking and new techniques to transform the state of the art.

An Example

Let's look at an example demonstrating the complexity of crash consistency: a simple database management system (DBMS) that stores its data in a single file. To maintain transactional atomicity across a system crash, the DBMS can use an update protocol called *undo logging*: before updating the file, the DBMS simply records those portions of the file that are about to be updated in a separate log file.¹¹ The pseudocode is shown in Figure 1; *offset* and *size* correspond to the portion of the *dbfile* that should be modified, and whenever the DBMS is started, the DBMS rolls back the transaction if the log file exists and is fully written (determined using the *size* field). The pseudocode in Figure 1 uses POSIX system calls (POSIX is the standard file-system interface used in Unix-like operating systems). In an ideal world, one would expect the pseudocode to work on all file systems implementing the POSIX interface. Unfortunately, the pseudocode does not work on *any* widely used file-system configuration; in fact, it requires a different set of measures to make it work on each configuration.

Because file systems buffer writes in memory and send them to disk later, from the perspective of an application most file systems can *reorder* the effects of system calls before persisting them on disk. For example, with some file systems (ext2, ext4, xfs, and btrfs in their default configurations, but not ext3), the deletion of the log file can be reordered before the write to the database file. On a system crash in these file systems, the log file might be found already deleted from the disk, while the database has been updated partially. Other file systems can persist a system call partially in seemingly nonsensical ways: in ext2 and nondefault configurations of ext3 and ext4, while writing (appending) to the log file, a crash might leave garbage data in the newly appended portions of the file; in such file systems, during recovery, one cannot differentiate whether the log file contains garbage or undo information.

Figure 2 shows the measures needed for undo logging to work on Linux file-system configurations (“.” refers to the current directory); the red parts are the additional measures needed. Comments in the figure explain which measures are required by different file systems: we considered the default configurations of ext2, ext3, ext4, xfs, and btrfs, and the *data=writeback* configuration of ext3/4 (denoted as ext3-wb and ext4-wb). Almost all measures simply resort to using the *fsync()* system call, which flushes a given file (or directory) from the buffer cache to the disk and is used to prevent the file system from reordering updates. The *fsync()* calls can be arbitrarily costly, depending on how the file system implements them; an efficient application will thus try to avoid *fsync()* calls when possible. With only a subset of the *fsync()* calls, however, an implementation will be consistent only on some file-system configurations.

Note that it is not practical to use a verified implementation of a single update protocol across all applications; the update protocols found in real applications vary widely and can be more complex than in Figure 2. The choice can depend on performance characteristics; some applications might aim for sequential disk I/O and

Figure 1. Incorrect undo-logging pseudocode.

```
creat(log);
# Making a backup in the log file
write(log, "<offset>,<size>,<data>");
write(dbfile, offset, data); # Actual Update
unlink(log); # Deleting the log file
```

Figure 2. Undo-logging pseudocode that works correctly in Linux file systems.

```
creat(log);
write(log, "<offset>,<chksum>,<size>,<data>");
fsync(log);
fsync(.);
write(dbfile, offset, data);
fsync(dbfile);
unlink(log);
fsync(.);
```

Log file can end up with garbage, in ext2, ext3-wb, ext4-wb

write(log) and write(dbfile) can re-order in all considered configurations

creat(log) can be re-ordered after write (dbfile), according to warnings in Linux manpage. Occurs on ext2.

write(dbfile) can re-order after unlink(log) in all considered configurations except ext3's default mode

If durability is desired, in all considered configurations

prefer an update protocol that does not involve seeking to different portions of a file. The choice can also depend on usability characteristics. For example, the presence of a separate log file unduly complicates common workflows, shifting the burden of recovery to include user involvement. The choice of update protocol is also inherently tied to the application's concurrency mechanism and the format used for its data structures.

Current State of Affairs

Given the sheer complexity of achieving crash consistency among different file systems, most developers write incorrect code. Some applications (for example, Mercurial) do not even try to handle crashes, instead assuming that users will manually recover any data lost or corrupted as a result of a crash. While application correctness depends on the intricate crash behavior of file systems, there has been little formal discussion on this topic.

Two recent studies investigate the correctness of application-level crash consistency: one at the University of Wisconsin–Madison²¹ and the other at Ohio State University and HP Labs.²⁹ The applications analyzed include distributed systems, version-control systems, databases, and virtualization software; many are widely used applications written by experienced developers, such as Google's LevelDB and Linus Torvalds's Git. Our study at the University of Wisconsin–Madison found more than 30 vulnerabilities exposed under widely used file-system configurations; among the 11 applications studied, seven were affected by data loss, while two were affected by silent errors. The study from Ohio State University and HP Labs had similar results: they studied eight widely used databases and found erroneous behavior in all eight.

For example, we found that if a file system decides to reorder two `rename()` system calls in HDFS, the HDFS namenode does not boot² and results in unavailability. Therefore, for portable crash consistency, `fsync()` calls are required on the directory where the `rename()` calls occur. Presumably, however, because widely used file-system configurations rarely reorder the `rename()` calls, and

Try It Yourself!

Many application-level crash-consistency problems are exposed only under uncommon timing conditions or specific file-system configurations, but some are easily reproduced. As an example, on a default installation of Fedora or Ubuntu with a Git repository, execute a git-commit, wait for five seconds, and then pull the power plug; after rebooting the machine, you will likely find the repository corrupted. Fortunately, this particular vulnerability is not devastating: if you have a clone of the repository, you likely can recover from it with a little bit of work. (Note: do not do this unless you are truly curious and will be able to recover from any problems you cause.)

The Unspoken Agreement

What *can* applications rely on? File-system developers seem to agree on two rules that govern what information is preserved across system crashes. The first is subtle: information already on disk (file data, directory entries, file attributes, among others) is preserved across a system crash, unless one explicitly issues an operation affecting it.

The second rule deals with `fsync()` and similar constructs (`msync()`, `O_SYNC`, and so on) in Unix-like operating systems. An `fsync()` on a file guarantees the file's data and attributes are on the storage device when the call returns, but with some subtleties. A major subtlety with `fsync()` is the definition of *storage device*: after information is sent to the disk by `fsync()`, it can reside in an on-disk cache and hence can be lost during a system crash (except in some special disks). Operating systems provide ad hoc solutions to flush the disk cache *to the best of their ability*; since you might be running atop a fake hard drive,⁸ nothing is promised. Another subtlety relates broadly to directories: directory entries of a file and the file itself are separate entities and can each be sent separately to the disk; an `fsync()` on one does not imply the persistence of others.

Best Practices for Application Developers

Developers can alleviate the problem of crash consistency within their applications by following these recommended practices:

Use a library. Implementing consistency directly atop the file-system interface is like pleading insanity in court: you do it only if you have no other choice. A wiser strategy is to use a library, such as SQLite, that implements crash consistency below your application whenever possible.

Document guarantees and requirements. Consistency guarantees provided by applications can be confusing; some developers can be unclear about the guarantees provided by their own applications. Documenting file-system behaviors that the application requires to maintain consistency is more complicated, since both application developers and users are often unclear about file-system behavior. The best documentation is a list of supported file-system configurations.

Test your applications. Because of the confusing crash behavior exhibited by file systems, it is important to test applications. Among the tools publicly available for finding application crash vulnerabilities, ALICE²¹ has been used successfully for testing eleven applications; ALICE also clearly shows which program lines lead to a vulnerability. The public version of ALICE, however, does not work with `mmap()` memory and some rare system calls. There is another tool designed for testing file systems⁹ that works with any application that runs on Linux, but it is less effective.

Java (in which HDFS is written) does not directly allow calling `fsync()` on a directory, the issue is currently ignored by HDFS developers.

As another example, consider LevelDB, a key-value store that adds any inserted key-value pairs to the end of a log file. Periodically, LevelDB

switches to a new log file and compacts the previous log file for faster record retrieval. We found that, during this switching, an `fsync()` is required on the old log file that is about to be compacted;¹⁹ otherwise, a crash might result in some inserted key-value pairs disappearing.

Many vulnerabilities arise because application developers rely on a set of popular beliefs to implement crash consistency. Unfortunately, much of what seems to be believed about file-system crash behavior is not true. Consider the following two myths:

► **Myth 1: POSIX defines crash behavior.** POSIX¹⁷ defines the standard file-system interface (`open`, `close`, `read`, and `write`) exported by Unix-like operating systems and has been essential for building portable applications. Given this, one might believe that POSIX requires file systems to have a reasonable and clearly defined response to crashes, such as requiring that directory operations be sent to the disk in order.¹⁸ Unfortunately, there is little clarity as to what exactly POSIX defines with regard to crashes,^{3,4} leading to much debate and little consensus.

► **Myth 2: Modern file systems require and implement in-order metadata updates.** Journaling, a common technique for maintaining file-system metadata consistency, commits different sets of file-system metadata updates (such as directory operations) as atomic transactions. Journaling is popular among modern file systems and has traditionally committed metadata updates in order;¹² hence, it is tempting to assume modern file systems guarantee in-order metadata updates. Application developers should not assume such guarantees, however. Journaling is an internal file-system technique; some modern file systems, such as `btrfs`, employ techniques other than journaling and commonly reorder directory operations. Furthermore, even file systems that actually use journaling have progressively reordered more operations while maintaining internal consistency. Consider `ext3/4`: `ext3` reorders only overwrites of file data, while `ext4` also reorders file appends; according to Theodore Ts'o, a maintainer of `ext4`, future journaling file systems might reorder more (though unlikely with `ext4`).

Should file-system developers be blamed for designing complicated file systems that are unfavorable for implementing crash consistency? Some complex file-system behaviors can (and should) be fixed. Most behaviors that make application consistency dif-



Recent research has confirmed that crashes are problematic: many applications (including some widely used and developed by experienced programmers) can lose or corrupt data on a crash or power loss.



ficult, however, are essential for general-purpose file systems.

To illustrate, consider reordering, the behavior that is arguably the least intuitive and causes the most crash-consistency vulnerabilities. In our study, a file system that provided in-order operations (and some minimal atomicity) exposed only 10 vulnerabilities, all of minor consequences; in comparison, 31 were exposed in `btrfs` and 17 in `ext4`. In current environments with multiple applications running simultaneously, however, a file system requires reordering for good performance. If there is no reordering, `fsync()` calls from important applications will be made to wait for writes from nonessential tasks to complete. Indeed, `ext3` in its default configuration provides an (almost) in-order behavior, but has been criticized for unpredictably slow `fsync()` calls.⁷

Moving Forward

Fortunately, not all is bleak in the world of crash consistency, and recent research points toward a number of interesting and plausible solutions to the problems outlined in this article. One approach is to help developers build correct update protocols. At least two new open source tools are available publicly for consistency testing (though neither is mature yet): `ALICE`,²⁰ the tool created for our research study at the University of Wisconsin–Madison, and a tool designed by Linux kernel developers⁹ for testing file-system implementations. `ALICE` is more effective for testing applications since it verifies correctness on a variety of simulated system crashes for a given application test case. In contrast, the kernel tool verifies correctness only on system crashes that occur with the particular execution path traversed by the file system during a run of the given test case.

Two other testing tools are part of recent research but are not yet publicly available: `BOB`²¹ from our study, and the framework used by researchers from Ohio State University and HP Labs.²⁹ Both of these are similar to the kernel tool.

A second approach for better application crash consistency is for file systems themselves to provide better, more easily understood abstractions

that enable both correctness and high performance for applications. One solution would be to extend and improve the current file-system interface (in the Unix world or in Windows); however, the interface has been built upon many years of experience and standardization, and is hence resistant to change.¹⁶ The best solution would provide better crash behavior with the current file-system interface. As previously explained, however, in-order updates (that is, better crash behavior) are not practical in multitasking environments with multiple applications. Without reordering in these environments, the performance of an application depends significantly on the data written by other applications in the background and will thus be unpredictable.

There is a solution. Our research group is working on a file system that maintains order only within an application. Constructing such a file system is not straightforward; traditional file systems enforce some order between metadata updates¹⁰ and therefore might enforce order also between different applications (if they update related metadata). Another possible approach, from HP Labs,²⁶ does change the file-system interface but keeps the new interface simple, while being supported on a production-ready file system.

A third avenue for improving the crash consistency of applications goes beyond testing and seeks a way of formally modeling file systems. Our study introduces a method of modeling file systems that completely expresses their crash behavior via abstract persistence models. We modeled five file-system configurations and used the models to discover application vulnerabilities exposed in each of the modeled file systems. Researchers from MIT⁵ have more broadly considered different formal approaches for modeling a file system and found Hoare logic to be the best.

Beyond local file systems, application crash consistency is an interesting problem in proposed storage stacks that will be constructed on the fly, mixing and matching different layers such as block remappers, logical volume managers, and file systems.^{27,28} An expressive language is required for specifying the complex storage guarantees and requirements of the different lay-

ers in such storage stacks. Our group is also working on such a language, along with methods to prove the overall correctness of the entire storage stack.¹

Conclusion

This article aims to convince readers that application-level crash consistency is a real and important problem. Similar problems have been faced before in other areas of computer systems, in the domains of multiprocessor shared memory and distributed systems. Those problems have been overcome by creating new abstractions, understanding various trade-offs, and even thinking about the problem with analogies to baseball.²⁵ Similar solutions are possible for application crash consistency, too, but only with the involvement of the wider systems community. **C**

Related articles on queue.acm.org

Abstraction in Hardware System Design

Rishiyur S. Nikhil

<http://queue.acm.org/detail.cfm?id=2020861>

Storage Systems: Not Just a Bunch of Disks Anymore

Erik Riedel

<http://queue.acm.org/detail.cfm?id=864059>

Keeping Bits Safe: How Hard Can It Be?

David S. H. Rosenthal

<http://queue.acm.org/detail.cfm?id=1866298>

References

1. Alagappan, R., Chidambaram, V., Sankaranarayanan Pillai, T., Arpaci-Dusseau, A.C., Arpaci-Dusseau, R.H. Beyond storage APIs: Provable semantics for storage stacks. In *Proceedings of the 15th Workshop on Hot Topics in Operating Systems* (Kartaouse Ittingen, Switzerland, May 2015).
2. Al-Kiswani, S. Namenode fails to boot if the file system reorders rename operations, 2014; <http://issues.apache.org/jira/browse/HDFS-6820>.
3. Aurora, V. POSIX v. reality: A position on O PONIERS, 2009; <http://lwn.net/Articles/351422/>.
4. Austin Group Defect Tracker. 0000672: Necessary step(s) to synchronize filename operations on disk, 2013; <http://austingroupbugs.net/view.php?id=672>.
5. Chen, H., Ziegler, D., Chlipala, A., Kaashoek, M. F., Kohler, E., Zeldovich, N. Specifying crash safety for storage systems. In *Proceedings of the 15th Workshop on Hot Topics in Operating Systems* (Kartaouse Ittingen, Switzerland, May 2015).
6. Corbet, J. Ext4 and data loss, 2009; <https://lwn.net/Articles/322823/>.
7. Corbet, J. That massive filesystem thread, 2009; <http://lwn.net/Articles/326471/>.
8. Davies, C. Fake hard drive has short-term memory not 500GB. SlashGear, 2011; <http://www.slashgear.com/fake-hard-drive-has-short-term-memory-not-500gb-08145144/>.
9. Edge, J. Testing power failures, 2015; <https://lwn.net/Articles/637079/>.
10. Ganger, G.R., Patt, Y.N. 1994. Metadata update performance in file systems. In *Proceedings of the 1st Symposium on Operating Systems Design and Implementation*. (Monterey, CA, Nov. 1994), 49–60.
11. Garcia-Molina, H., Ullman, J.D., Widom, J. *Database Systems: The Complete Book*. Prentice Hall Press, 2008.

12. Hagmann, R. Reimplementing the Cedar file system using logging and group commit. In *Proceedings of the 11th ACM Symposium on Operating Systems Principles*, (Austin, TX, Nov. 1987).
13. Kim, H., Agrawal, N., Ungureanu, C. Revisiting storage for smartphones. In *Proceedings of the 10th Usenix Symposium on File and Storage Technologies* (San Jose, CA, Feb. 2012).
14. Lamport, L. How to make a multiprocessor computer that correctly executes multiprocess programs. *IEEE Trans. Computers* 28, 9 (1979), 690–691.
15. Mercurial. Dealing with repository and diststate corruption, 2014; <http://mercurial.selenic.com/wiki/RepositoryCorruption>.
16. Microsoft. Alternatives to using transactional NTFS; [https://msdn.microsoft.com/en-us/library/windows/desktop/hh802690\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/hh802690(v=vs.85).aspx).
17. Open Group Base Specifications. POSIX.1-2008 IEEE Std 1003.1, 2013; <http://pubs.opengroup.org/onlinepubs/9699919799/>.
18. Sankaranarayanan Pillai, T. Possible bug: `fsync()` required after calling `xrename()`, 2013; <https://code.google.com/p/leveldb/issues/detail?id=189>.
19. Sankaranarayanan Pillai, T. Possible bug: Missing a `fsync()` on the log file before compaction, 2013; <https://code.google.com/p/leveldb/issues/detail?id=187>.
20. Sankaranarayanan Pillai, T., Chidambaram, V., Alagappan, R., Al-Kiswani, S., Arpaci-Dusseau, A.C. and Arpaci-Dusseau, R.H. ALICE: Application-Level Intelligent Crash Explorer; <http://research.cs.wisc.edu/adsl/Software/alice/>.
21. Sankaranarayanan Pillai, T., Chidambaram, V., Alagappan, R., Al-Kiswani, S., Arpaci-Dusseau, A.C. and Arpaci-Dusseau, R.H. 2014. All file systems are not created equal: on the complexity of crafting crash-consistent applications. In *Proceedings of the 11th Symposium on Operating Systems Design and Implementation* (Broomfield, CO, Oct. 2014).
22. Sewell, P., Sarkar, S., Owens, S., Nardelli, F.Z. and Myreen, M.O. x86-TSO: A rigorous and usable programmer's model for x86 multiprocessors. *Commun. ACM* 53, 7 (July 2010): 89–97.
23. Shvachko, K., Kuang, H., Radia, S. and Chansler, R. The Hadoop Distributed File System. In *Proceedings of the 26th IEEE Symposium on Mass Storage Systems and Technologies* (Incline Village, NV, May 2010).
24. Sorin, D.J., Hill, M.D., Wood, D.A. A Primer on Memory Consistency and Cache Coherence. Morgan & Claypool Publishers, 2011.
25. Terry, D. Replicated data consistency explained through baseball. *MSR Technical Report* (Oct. 2011).
26. Verma, R., Mendez, A.A., Park, S., Mannarswamy, S.S., Kelly, T.P., and Morrey III, C.B. Failure-atomic updates of application data in a Linux file system. In *Proceedings of the 13th Usenix Symposium on File and Storage Technologies* (Santa Clara, CA, Feb. 2015).
27. VMware. Software-defined storage (SDS) and storage virtualization; <http://www.vmware.com/software-defined-datacenter/storage>.
28. VMware. The VMware perspective on software-defined storage; <http://www.vmware.com/files/pdf/solutions/VMware-Perspective-on-software-defined-storage-white-paper.pdf>.
29. Zheng, M., Tucek, J., Huang, D., Qin, F., Lillibridge, M., Yang, E. S., Zhao, B. W., Singh, S. Torturing databases for fun and profit. In *Proceedings of the 11th Symposium on Operating Systems Design and Implementation* (Broomfield, CA, Oct. 2014).

T. Sankaranarayanan Pillai, Vijay Chidambaram, and Ramnathan Alagappan (madthanu, vijayc, ra@cs.wisc.edu) are Ph.D. candidates in the Department of Computer Science at the University of Wisconsin–Madison. Chidambaram is joining the faculty at the University of Texas at Austin.

Samer Al-Kiswani (samera@cs.wisc.edu) is a postdoctoral fellow in the Department of Computer Science at the University of Wisconsin–Madison.

Andrea Arpaci-Dusseau and Remzi Arpaci-Dusseau (dusseau, remzi@cs.wisc.edu) are professors of computer science at the University of Wisconsin–Madison.

Copyright held by authors.
Publication rights licensed to ACM. \$15.00

We have to choose to build a Web that is accessible to everyone.

BY RICH HARRIS

Dismantling the Barriers to Entry

A WAR IS being waged in the world of Web development. On one side is a vanguard of toolmakers and tool users, who thrive on the destruction of bad old ideas (“old,” in this milieu, meaning anything that debuted on Hacker News more than a month ago) and raucous debates about transpilers and suchlike.

On the other side is an increasingly vocal contingent of developers who claim—not entirely without justification—the head-spinning rate of innovation makes it impossible to stay up to date, and the Web is disintegrating into a jumble of hacks upon opinions, most of which are wrong, and all of which will have changed by the time hot-new-thing.js reaches version 1.0.0.

This second group advocates a return to the basics, eschewing modern JavaScript libraries and frameworks in favor of untamed DOM APIs (the DOM being the closest we unwashed Web developers ever get to “bare metal”). Let’s call it the back-to-the-land movement. The back-to-the-landers argue tools slow

the Web down, harm accessibility, and increase fragility. You can often find them linking to vanilla-js.com in the comments of programming blogs.

Here is Peter-Paul Koch, the creator of quirksmode.org, in a recent article⁶ (emphasis original):

“The movement toward toolchains and ever more libraries to do ever less useful things has become hysterical, and with every day that passes I’m more happy with my 2006 decision to ignore tools and just carry on. *Tools don’t solve problems anymore, they have become the problem.*”

Setting aside the “get off my lawn” tone of much of this commentary, the movement does have valid concerns. But we expect more of the Web than we used to—real-time collaboration, personalized apps, rich interactivity.





We cannot expect software engineers to build those experiences without tools any more than we expect civil engineers to build suspension bridges by hand. As Facebook's Sebastian Markbåge says in a direct response to Koch,⁷ "the only time you can say that the Web is "good enough" is when you are building for yesterday's Web."

As in any war, there are false dichotomies (simplicity versus power), hypocrisies (abandoning libraries then writing acres of app code that do the same thing, albeit without documentation or tests), and casualties. It is the casualties I want to talk about.

Front-Enders: An Endangered Species?

Until relatively recently, "front end developer" was a slightly derisive term

for someone who could cobble together some HTML and CSS and sprinkle some JavaScript on top of it, perhaps after searching Stack Overflow for "how to hide element with jQuery." The front-ender was responsible for adding the Google Analytics script snippet to the CMS article template, and perhaps adding a carousel of sliding images (the traditional cure for the marketing department's indecision about what to put on the homepage), but was never trusted with anything particularly important.

Then along came Backbone,¹ which was the starting pistol in the race towards ever more elaborate JavaScript application frameworks. Many modern Web apps push almost all the logic out to the client, the result being that as applications become more sophisti-

cated, so must the tools—and the people using them.

As a consequence, many commentators have placed the traditional front-ender on extinction watch. Trek Glowacki, a core member of the Ember.js team (Ember is one of the aforementioned client-side application frameworks), wrote in response to a lament about build tools:

"I know everyone on Ember core sympathizes with Web developers whose careers started during the 'download a zip, add some script tags, FTP into production' era for the 'front end' and now feel a bit startled that all their favorite tools are becoming increasingly complex. But, the fact remains, that era is ending."⁵

In other words, "get with the program." Glowacki is not wrong, just like

Koch isn't wrong, but there is a problem with modern tools—newcomers to the field, after they have been greeted with an overwhelming number of choices, are expected to learn a dizzying array of new concepts (insert joke about “transclusion” here) before they can actually build anything. The incredible power of those tools is only really available to a select few—those with the determination to ascend a steep learning curve, and the time and inclination to keep pace with our community's frantic innovation.

“Learn to Code” Is Not the Answer

Back when the Web was a simpler place, it was a welcoming environment for newbie programmers. There were fewer tools, and the ones we had were a good deal less sophisticated, but we made up for it with the power of “view source.” In those Wild West days, before we cared about *best practices*, it was surprisingly easy to reverse engineer a lot of Web software.

Web development has matured spectacularly in a few short years. But the tools that have supplanted “view source” (which is useless in an age of transpiled, minified code) are not accessible to the vast majority.

It is not simply a question of better training for those who would be professional software engineers. The power and beauty of the Web was always that anyone could participate as a creator as well as a consumer—scientists, academics, artists, journalists, activists, entertainers, educators—most of whom have yet to unlock the thrilling possibilities of modern Web technologies.

One way we have tried to address this problem is with the “learn to code” movement, which has spawned an entire industry of startups (startup culture itself being one of the prime drivers of learn to code). Politicians love it because it makes them look forward-thinking, though no one is quite sure if Michael Bloomberg ever did finish his Codecademy course.²

There is plenty to admire about learn to code, of course. Many people have developed skills that would otherwise have been out of reach. But the movement rests on two odd assumptions—firstly our priority should be to make more programmer talent

rather than making programming more accessible, and secondly that “learning to code” consists of absorbing facts about programming languages and practicing the formation of correct syntax.

In reality, learning how to program is a process of developing the ability to model problems in such a way that a computer can solve them—something that only happens through experience. You do not learn a foreign language by learning how to conjugate verbs and pluralize nouns; you learn by picking up phrases and practicing them, and reading and listening to native speakers until it becomes natural. Every language teacher knows this, yet to a large extent it is not how we teach programming languages.

We do not need the 1,437th explanation of prototypal inheritance or JavaScript's ‘this’ keyword. What we need are tools that allow novices to express their ideas without a complete knowledge of the process by which it happens.

Enter Ractive.js

A few years ago I was in need of such a tool, having recently joined the interactive news team at theguardian.com. News interactives typically contain a lot of *state*, represented in several different visually rich forms, and have to handle many different modes of user interaction—a recipe for buggy code, especially when written against news industry deadlines (we laugh at the term “agile”). I was well aware my jQuery spaghetti was always a few keystrokes away from implosion, but more advanced tools such as Angular were both too intimidating and yet somehow inadequate for the task at hand.

I had been looking forward to the day when someone would let me in on the secret to doing it properly, but that day never came. There simply were not any tools designed to make my job easier, so I resolved to create one myself.

Laid bare, the problem is relatively simple to articulate. The state of a Web app UI at any given moment can be described as a function of application state, and our task is to manipulate the DOM until the reality matches the intention.

On the server, it is easy: write a template, compile it to a function with a templating engine, call it with some

data, and serve the resulting HTML to the client. But string templating is a bad technique once you are in the browser. Repeatedly generating HTML and inserting it into the document means trashing the existing DOM, which taxes the garbage collector and destroys state (such as which element is focused, and where the cursor is). Because of that, developers typically break their applications apart into microscopic chunks, with dedicated custom Model and View classes tied together with an events system. MVC duct tape is the new jQuery spaghetti.

Ractive.js¹⁰ was designed to allow developers to use the declarative power of templates to their fullest extent without the sacrifices that come from string-based templating systems. The idea, novel at the time (though less so now, as other tools have adopted a similar approach), was that a template parser that understood both HTML and template tags could generate a tree structure that a data-binding engine could later use to manipulate the DOM with surgical precision. The developer need do nothing more than occasionally provide new data.

This is not the virtual DOM diffing technique used by React.js and other similar libraries. That approach has some deeply interesting properties, but data-binding—that is, updating the parts of the DOM that are known to correspond to particular values that have changed, rather than re-rendering everything and not updating the bits that have not changed—is typically a great deal more performant.

Since then, Ractive has added (and in some cases pioneered) many new features: a component system, declarative animations and transitions, full SVG support, encapsulated CSS, server-side rendering, and more. In terms of mindshare, we are a minnow next to the likes of Angular, Ember, Meteor and React, even though we have contributors from all around the world and Ractive is used for all kinds of websites, from e-commerce to enterprise monitoring software.

But the thing the team and I are most proud of is the way it has allowed less experienced developers to bring their ideas to life on the Web.

A magazine article is a suboptimal place for code samples demonstrating

an interactive UI library, but if you are curious you should visit <http://learn.ractivejs.org> for an interactive tutorial.

Lessons Learned

The question: “Will this make it easier or more difficult for novice developers to get started?” is always on our minds when we are building Ractive. Interestingly, we have never found this has required us to sacrifice power for more experienced developers—there is no “dumbing down” in software development, only clear APIs versus convoluted APIs. By focusing on the beginner experience, we make life better for all of our users.

Over the years, we have distilled this mind-set into a toolmaker’s checklist. Some of these points are, frankly, aspirational. But we have found them to be useful guidelines even when we fall short, and they apply to tools of all kinds.

Readme-driven development. Often, when we write code designed to be used by other people, we focus on the implementation first, then slap an interface on it as a final step. That is natural—figuring out the right algorithms and data structures is the interesting part, after all—but completely backward.

When the API is an afterthought, you are going to get it wrong nine times out of ten. The same is true of the implementation, but there is a crucial difference—you can fix a lousy implementation in a subsequent release, but changing an API means breaking everyone else’s code and thereby discouraging them from upgrading. (Worse, you could try to accommodate both the old and the new API, printing deprecation warnings where necessary, and causing Zalgo to appear in your codebase as a result. I speak from experience.)

Instead, try to write the first draft of your README, code samples and all, before writing any code. You will often find that doing so forces you to articulate the problem you are trying to solve with a great deal more clarity. Your starting vocabulary will be richer, your thoughts will be better arranged, and you will end up with a more elegant API.

The Ractive API for getting and setting data is a case in point. We were very clear that we wanted to allow users to use plain old JavaScript objects (POJOs), rather than insisting they wrap values



The question: “Will this make it easier or more difficult for novice developers to get started?” is always on our minds when we are building Ractive.



in a Ractive-specific observable class (think ‘Backbone.Model’ or ‘ko.observable’). That posed some implementation challenges, but it was unquestionably the right move. We are currently in the process of overhauling the internal architecture, which will deliver significant performance boosts to many users without breaking their apps.

The phrase “Readme-driven development” was coined, or at least popularized, by Tom Preston-Werner.⁹

Eliminate dependencies. Dependency management in JavaScript is a pain, even for experts—especially in the browser. There are tools designed to make the situation easier, such as Browserify and RequireJS (or Webpack, Esperanto, and JSPM, if you are part of the revolutionary vanguard), but they all have steep learning curves and sometimes go wrong in ways that are spectacularly difficult to debug.

So the silent majority of developers use the tried-and-tested solution of manually adding `<script>` tags. This means that libraries must be included on the page *after* their dependencies (and *their* dependencies, and so on). Forgot to include `underscore.js` before `backbone.js`? Here you go n00b, have a cryptic “Cannot read property ‘extend’ of undefined” error.

Often, the dependencies are not actually necessary—it is incredibly common to see libraries depend on jQuery for the sake of one or two easy-to-implement methods, for example. (Yes, it is probably already on the page. But which version?) When they are necessary, library authors should provide a version of the library with dependencies bundled alongside the version without. Do not worry about potential duplication; that is the least of our worries at this stage.

Do not over-modularize. Since the advent of `node.js` and `npm`, a vocal group of developers has evangelized the idea that code should only be released in the form of tiny modules that do very specific jobs. This is at least part of the reason `npm` has more packages than any other package manager.

On the face of it, this seems like an excellent idea, and a good way to cut down on the amount of imported-but-unused code in an app or library. But the end result is the burden of thinking rigorously about architectural

questions is pushed from toolmakers to app authors, who must typically write large amounts of glue code to get the various tiny modules to talk to each other.

No one is going to build the next jQuery, because they would instantly be subjected to modularity shaming (an excellent phrase coined by Pete Hunt, formerly of the React.js team). And that is a crushing shame, because it means we will not have any more libraries with the same level of learnability and philosophical coherence.

In case you think I am overstating things, there is literally a package on npm called “no-op.” Its source code is as follows:

```
module.exports = function noop() {}
```

It has had three releases. It has a test suite! At least it does not use Travis-CI for continuous integration, unlike the “max-safe-integer” package, which exports the number 9007199254740991. These packages are not jokes. They were created unironically by leading members of the JavaScript community.

Tiny modules can be just as bad as monolithic frameworks. As usual, there is a happy medium we should aim for.

Universal module definition (UMD). Speaking of modules, you should ideally make your code consumable in as many different ways as possible. The three most common formats are AMD (used via RequireJS and its various clones), CommonJS (used in node.js, or via Browserify), and browser globals.

The Universal Module Definition lets you target all three of these environments. There are a few different versions, but the basic pattern is illustrated in Figure 1.

The first part detects a CommonJS environment, the second detects AMD, and if neither of those is found it falls back to creating a browser global.

Prominent download links. It goes without saying these days that if you want to release an open source library, it should exist in a public VCS repository (GitHub being the de facto standard) and be published to npm. Both of those are true, but it is important to have a download link available for users who are not comfortable using git or npm, or who want to quickly try out a library



The thing the team and I are most proud of is the way [Ractive] has allowed less experienced developers to bring their ideas to life on the Web.



without rigging up a new project with a package.json and a build step.

This need not involve lots of manual labor or complex automation (though it is straightforward to set up with services like cdnjs.com). One easy way to provide a download link is to include the built library in the GitHub repo (for example, dist/my-library.min.js) and tag specific commits so it is easy to link to specific versions shown in Figure 2.

Good error messages. Error and warning messages will never be a source of joy, but they can at least be a source of enlightenment. A well-crafted error message is worth pages of documentation, because it appears exactly when the developer needs it.

On the Ractive team, we decided a few months ago that we were doing more harm than good by trying to shield developers from their mistakes. Now, we print verbose warnings to the console explaining how they can guard against common bugs and make their applications more performant. (This can be disabled if the developer so wishes.) Where it makes sense, we include links to relevant documentation inside error messages. In most browsers, these turn into clickable hyperlinks.

At one stage, we had a class of bugs that were very difficult to unravel. We did not know quite what was causing the problem, but we were able to detect the state that gave rise to it, so we started throwing errors when that state was reached that included a friendly “please raise an issue with a reproduction!” message, linking to our issues page. Users felt empowered to do something about what would otherwise have been a highly frustrating experience (in some cases becoming first-time GitHub contributors), and we gathered the test cases we needed to solve the bug.

Avoid this command line. This guideline only really applies to browser-based tools, but it is an important one: if your introductory instructions involve using the command line, you have already lost half your audience.

That might sound hyperbolic unless you have spent a lot of time with novice developers. But try to remember how *lost* you felt the first time you opened the terminal. GUIs make the things we are working with—folders

Figure 1. The Universal Module Definition ensures your library can be used anywhere.

```
(function (global, factory) {
  typeof exports === 'object' && typeof module !== 'undefined' ? module.exports =
  factory() :
  typeof define === 'function' && define.amd ? define(factory) :
  global.MyLibrary = factory()
})(this, function () {
  var MyLibrary = {};
  /* some code happens... */
  return MyLibrary;
}));
```

Figure 2. npm and git are all you need to manage releases.

```
# create the dist files (npm run is a great task runner!)
npm run build

# create a version 0.2.0 tag and add it
# to the 'releases' tab on the repo
git tag -a v0.2.0 -m 'version 0.2.0'
git push origin v0.2.0
```

and files and drives and servers—into almost physical, tangible things our brains are well evolved to understand, whereas the command line forces you to build a complex mental model.

Have you ever taken a wrong turn on the way to the restroom and ended up backstage? That is how most people feel when they open the terminal—like they are behind the curtain, and not in a good way.

Examples, examples, examples. Inviting people to consult the API documentation is polite developer-speak for “RTFM,” but no one wants to read the “fine” manual. What people really want—especially people who are not yet experts in your domain, and have not developed the right mental vocabulary—are examples.

I cannot articulate it any better than Mike Bostock, the creator of d3⁴, so I will not try. Instead I will just recommend his article “For Example.”³ The proliferation of copy-and-paste-able examples is one of the main reasons for d3’s massive success.

Eliminate jargon. Naming things is difficult, so do not bother. As far as possible, stick to vocabulary people are already familiar with (but do not make any assumptions about prior knowledge). Favor the slightly wordy but universally comprehensible over terse jargon.

You might need a more complex

vocabulary to describe the primitives *inside* your tool, but the less you force your users to become familiar with it, the better.

Empathize. While this is most nebulous item on the checklist, it is also the most important. The motivation to go the extra mile, and try to help people you do not know get the most out of your open source software, springs from empathy.

If your empathy reserves need a top-up, try reading a paper in a field with which you are unfamiliar. For most mortals, reading *Communications* front to back should suffice; you, dear reader, may need something stronger. Try *Papers We Love*.⁸ The bewilderment you feel closely matches that of the average human trying to learn Web development—or, for that matter, a highly experienced developer coming to your domain of expertise for the first time.

We Have to Build the Future We Want

It is depressingly common to hear people suggest the increasing complexity of the Web platform is inevitable, the price we pay for progress. This is a classic self-fulfilling prophecy—once we decide it is true (or worse, *right*) that Web development is best left to the professionals, we will stop striving to make it more accessible for everyone else.

This would be a tragedy of the highest order were it to come to pass. The Web has been a gateway drug for an entire generation of programmers (your present correspondent included), many of whom would never have otherwise experienced the sheer joy of computer science. There is no intrinsic reason it cannot continue to be. But it is up to us: we have to choose to build a Web that is accessible to everyone. ■

Related articles on queue.acm.org

Debugging AJAX in Production

Eric Schrock

<http://queue.acm.org/detail.cfm?id=1515745>

The Story of the Teapot in DHTML

Brian Beckman and Erik Meijer

<http://queue.acm.org/detail.cfm?id=2436698>

Best Practices on the Move: Building Web Apps for Mobile Devices

Alex Nicolaou

<http://queue.acm.org/detail.cfm?id=2507894>

References

1. <http://backbonejs.org>
2. Bloomberg, M. 2012; <https://twitter.com/mikebloomberg/status/154999795159805952>
3. Bostock, M. 2013; <http://bost.ocks.org/mike/example/>
4. <http://d3js.org/>
5. Glowacki, T. Comment on “Will there be continued support for people that do not want to use Ember-CLI?” (2015); <http://discuss.emberjs.com/t/will-there-be-continued-support-for-people-that-do-not-want-to-use-ember-cli/7672/3>
6. Koch, P.-P. Tools don’t solve the Web’s problems, they are the problem. http://www.quirksmode.org/blog/archives/2015/05/tools_dont_solve.html
7. Markbåge, S. Tooling is not the problem of the Web (2015); <https://medium.com/@sebmabage/tooling-is-not-the-problem-of-the-Web-cb0ae1fdbbc6>
8. <http://paperswelope.org/>
9. Preston-Werner, T. Readme driven development. <http://tom.preston-werner.com/2010/08/23/readme-driven-development.html>
10. <http://ractivejs.org>

Rich Harris is an interactive journalist at theguardian.com, where he uses Web technologies to tell stories in new ways through interactivity and data visualization. He is the creator and lead author of a number of open source projects.

Copyright held by author.
Publication rights licensed to ACM. \$15.00

DOI:10.1145/2714561

The Dissent system aims for a quantifiably secure, collective approach to anonymous communication online.

BY JOAN FEIGENBAUM AND BRYAN FORD

Seeking Anonymity in an Internet Panopticon

IN TODAY'S "BIG DATA" Internet, users often need to assume, by default, that their every statement or action online is monitored and tracked. Users' statements and actions are routinely linked with detailed profiles built by entities ranging from commercial vendors and advertisers to state surveillance agencies to online stalkers and criminal organizations. Indeed, recent revelations have raised the stakes enormously in Internet monitoring. Documents leaked by former National Security Agency contractor Edward Snowden revealed the U.S. government is conducting warrantless surveillance on a massive scale, and the long-term goal of the National Security Agency is to be "able to collect virtually everything available in the digital world."¹⁶

Internet users often have a legitimate need to be anonymous, or "not named or identified" by *Webster's*

definition of the term, to protect their online speech and activities from being linked to their real-world identities. Although the study of anonymous-communication technology is often motivated by high-stakes use cases (such as battlefield communication, espionage, or political protest against authoritarian regimes), anonymity actually plays many well-accepted roles in established democratic societies. For example, paying cash, voting, opinion polling, browsing printed material in a book store or library, and displaying creativity and low-risk experimentalism in forums (such as Slashdot and 4chan) are everyday examples of anonymous activity. Author J.K. Rowling used a pen name on a 2013 post-Harry Potter novel, presumably not out of fear of censorship or reprisal but merely "to publish without hype or expectation and . . . to get feedback under a different name."²²

Obtaining and maintaining anonymity on the Internet is a challenge. The state of the art in deployed tools (such as Tor²⁰) uses "onion routing" to relay encrypted connections on a detour passing through randomly chosen relays scattered around the Internet. Onion routing is scalable, supports general-purpose point-to-point communication, and appears to be effective against many of the attacks

» key insights

- With retailers, email service providers, advertisers, surveillance agencies, and stalkers all potentially monitoring, tracking, and profiling ordinary Internet users, those users can turn to anonymous communication to prevent the linking of their online activity to their real-world identities.
- Currently deployed anonymity tools, with Tor the best known, are based on "onion routing," a scalable general technique that is effective in many scenarios but inherently vulnerable to several attacks that are increasingly feasible.
- The Dissent project takes a collective approach to online anonymity, based on different algorithmic foundations from onion routing, offering concrete advantages, as well as some disadvantages, versus Tor.



currently known to be in use.¹⁰ Unfortunately, onion routing is also known to be vulnerable to several classes of attacks for which no solution is known or believed to be forthcoming soon; for example, using traffic confirmation, an attacker who compromises a major ISP or Internet exchange might in principle be able to de-anonymize many Tor users in a matter of days.¹² With intersection attacks, an adversary can rapidly narrow the anonymity of a target via actions linkable across time, much like Paula Broadwell and the “High Country Bandits” were de-anonymized.¹⁷ Finally, through software exploits or user error, an attacker can often circumvent anonymity tools entirely.²⁴

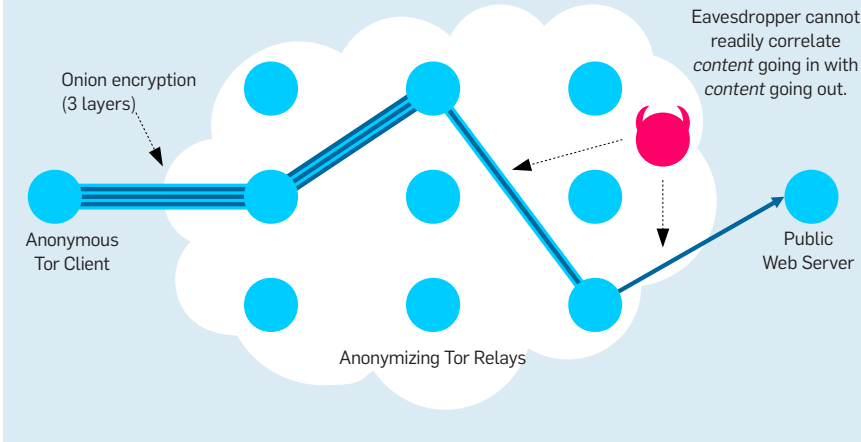
Currently deployed approaches to

anonymity also appear unable to offer accurate, principled measurement of the level or quality of anonymity a user might obtain. Considerable theoretical work has analyzed onion routing⁸ but relies on idealized formal models making assumptions that are unenforceable and may be untrue in real systems (such as users choose relays and communication partners at random) or depending on parameters unknown in practice (such as probability distributions representing user behavior).

Onion routing vulnerabilities and measurability limitations may stem from an attempt by developers of anonymity to achieve an impossible set of goals and defend an ultimately indefensible position. Currently deployed

tools offer a general-purpose, unconstrained, individualistic form of anonymous Internet access. However, many methods are available for “fingerprinting,” or tying unconstrained, individualistic network communication patterns to individual users. We suspect the only way to achieve measurable, provable levels of anonymity, and stake out a position defensible in the long term, is to develop more collective anonymity protocols and tools. It may be necessary for anonymity tools to constrain the normally individualistic behaviors of participating nodes, along with the expectations of users and possibly the set of applications and usage models to which these protocols and tools apply.

Figure 1. Onion routing.



Toward this end, we offer a high-level view of the Dissent project, a “clean-slate” effort at Yale University that began in the fall of 2009 to build practical anonymity systems embodying a collective model for anonymous communication (<http://dedis.cs.yale.edu/dissent/>). Dissent’s collective approach to anonymity is not and may never be a “drop-in” functional replacement for Tor or the individualistic, point-to-point onion routing model it implements. Rather, Dissent sets out to explore radically different territory in the anonymous-communication design domain, an approach that presents advantages, disadvantages, and many as-yet-unanswered questions. An advantage is the collective approach, making it easier to design protocols that provably guarantee certain well-defined anonymity metrics under arguably realistic environmental assumptions. A disadvantage is the collective approach is most readily applicable to multicast-oriented communication and is much less efficient or scalable than onion routing for point-to-point communication.

Dissent follows in the tradition of Herbivore,¹⁸ the first attempt (2003–2004) to build provable anonymity guarantees into a practical system and employ “dining cryptographers,” or DC-nets.³ Dissent utilizes both DC-nets and “verifiable shuffles,”¹⁵ showing for the first time how to scale the formal guarantees embodied in these techniques to offer measurable anonymity sets on the order of thousands of participants.²³ Dissent’s methods

of scaling individual anonymity sets are complementary and synergistic with techniques Herbivore pioneered for managing and subdividing large peer-to-peer anonymity networks; combining these approaches could enable further scalability improvements in the future.

Dissent incorporates the first systematic countermeasures to major classes of known attacks (such as global traffic analysis and intersection attacks).^{14,25} Because anonymity protocols alone cannot address risks (such as software exploits or accidental self-identification), the Dissent project also includes Nymix, a prototype operating system that hardens the user’s computing platform against such attacks.²⁴ Even with Nymix, however, Dissent can offer only network-level anonymity, in which the act of communicating does not reveal which user sent which message. No anonymity system can offer users personal anonymity if they disclose, say, their real-world identities in their message content.

While Dissent is still a research prototype, not yet ready for widespread deployment and may never be a direct replacement for onion routing tools like Tor due to possibly fundamental trade-offs, we hope it will increase the diversity of practical approaches and tools available for obtaining anonymity online.

Next, we present onion routing and Tor basics. We then describe four problems with onion routing that have remained unsolved for many years and may, unfortunately, be unsolvable. We then provide an overview of the Dissent

approach to anonymous communication and, finally, discuss open problems and future directions.

Onion Routing and Tor

Tor is the most widely deployed, general-purpose system for anonymous Internet communication.²⁰ Tor’s technical foundation is onion routing¹¹ derived in turn from mixnets.⁵

Onion routing uses successive layers of encryption to route messages through an overlay network, such that each node knows the previous and the next node in the route but nothing else. More precisely, let (V, E) be a connected, undirected network and $R \subseteq V$ be a set of nodes serving as relays. The set R is known to all nodes in V , as is the public key K_r , usable in some globally agreed-upon public-key cryptosystem, for each node $r \in R$. There is a routing protocol any node in V can use to send a message to any other node, but the nodes do not need to know the topology (V, E) .

If node s wishes to send message M to node d anonymously, s first chooses a sequence $(r_1, r_2, \dots, r_n)$ of relays. It then constructs an “onion” with n layers containing both the message and the routing information needed to deliver it without revealing node s ’s identity to any node except the first relay r_1 . The core of the onion is (d, M) , or the destination node and the message itself. The n th, or innermost, layer of the onion is

$$O_n = (r_n, \text{ENC}_{K_{r_n}}(d, M))$$

or the n th relay node and the encryption of the core under the n th relay’s public key. More generally, the i th layer O_i , $1 \leq i \leq n-1$, is formed by encrypting the $(i+1)$ st layer under the public key of the i th relay and then prepending the i th relay’s identity r_i :

$$O_i = (r_i, \text{ENC}_{K_{r_i}}(O_{i+1}))$$

When it has finished constructing the outermost layer

$$O_1 = (r_1, \text{ENC}_{K_{r_1}}(O_2))$$

node s sends $\text{ENC}_{K_{r_1}}(O_2)$ to r_1 , using the routing protocol of the underlay network (V, E) . When relay r_i , $1 \leq i \leq n$, receives the encryption of O_i with public

key K_{r_i} , it decrypts it using the private key k_{r_i} corresponding to K_{r_i} , thus obtaining both the identity of the next node in the route and the message it needs to send to this next node it sends using the underlying routing protocol. When $i = n$, the message is just the core (d, M) , because, strictly speaking, there is no O_{n+1} . We assume d can infer from routing protocol “header fields” of M that it is the intended recipient and need not decrypt and forward (see Figure 1).

Tor is a popular free-software suite based on onion routing. As explained on the Tor project website, <https://www.torproject.org>,²⁰ “Tor protects you by bouncing your communications around a distributed network of relays run by volunteers all around the world; it prevents somebody watching your Internet connection from learning what sites you visit, and it prevents the sites you visit from learning your [network] location.” The project provides free application software that can be used for Web browsing, email, instant messaging, Internet relay chat, file transfer, and other common Internet activities. Users can also obtain free downloads that integrate the underlying Tor protocol with established browsers and email clients. Moreover, Tor users can easily (but are not required to) transform their Tor installations into Tor relays, thus contributing to the overall capacity of the Tor network. Tor has more than two million daily users worldwide, with slightly over 15% of them in the U.S., and approximately 6,000 relays. These and other statistics are regularly updated on the Tor Metrics Portal.²¹

The IP addresses of Tor relays are listed in a public directory so Tor clients can find them when building circuits. (Tor refers to routes as “circuits,” presumably because Tor is typically used for Web browsing and other TCP-based applications in which traffic flows in both directions between the endpoints.) This makes it possible for a network operator to prevent its users from accessing Tor. The operator can simply disconnect the first hop in a circuit, or the connection between the client and the first Tor relay, because the former is inside the network and the latter is outside; this forces the Tor traffic to flow through a network gateway

where the operator can block it. Several countries that operate national networks, including China and Iran, have blocked Tor in precisely this way. Website operators can also block Tor users simply by refusing connections from the last relay in a Tor circuit; Craigslist is an example of a U.S.-based website that does so. As a partial solution, the Tor project supports “bridges,” or relays whose IP addresses are not listed in the public directory, of which there are approximately 3,000 today. Tor bridges are just one of several anti-blocking, or “censorship-circumvention,” technologies.

There is inherent tension in onion routing between low latency, one aspect of which is short routes (or, equivalently, low values of k), and strong anonymity. Because its goal is to be a low-latency anonymous-communication mechanism, usable in interactive, real-time applications, Tor uses three-layer onions, or sets $k = 3$, as in Figure 1. Despite this choice of small k , many potential users reject Tor due to its performance impact.⁶

Attacks on Onion Routing

Four categories of known attacks to which onion routing is vulnerable and for which no general defenses are known are outlined in the following sections.

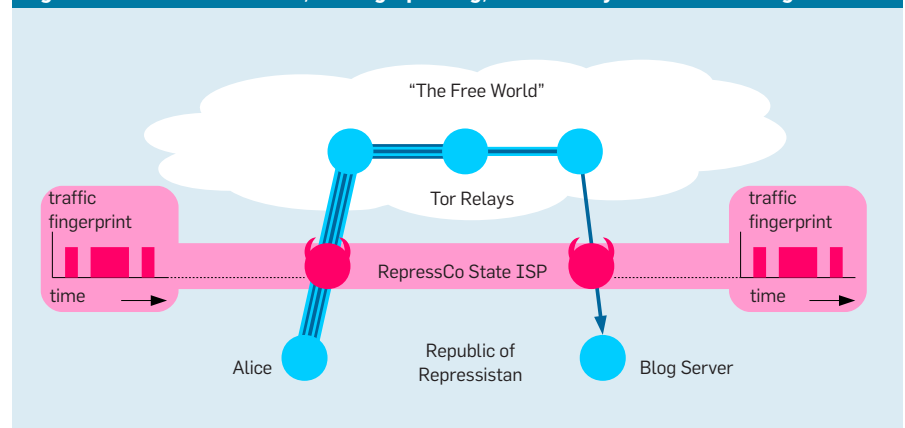
Global traffic analysis. Onion routing was designed to be secure against a local adversary, or one that might eavesdrop on some network links and/or compromise some relay nodes but only a small percentage of each. It was not designed for security against traffic analysis by a global adversary able to monitor large portions of the network constantly.

The most well known global-traffic-analysis attack—“traffic confirmation”—was understood by Tor’s designers but considered an unrealistically strong attack model and too costly to defend against.²⁰ In the standard scenario (see Figure 2), we assume the attacker cannot break Tor’s encryption but can monitor both the encrypted traffic flowing from the user to the first, or “entry” relay, and the traffic flowing from the final, or “exit” relay, to the user’s communication partner. This situation, while unlikely a decade ago, might be realistic today if both the user and the communication target are located in a single country, and the attacker is an ISP controlled or compromised by a state-level surveillance agency. In this case, the attacker needs to monitor, in principle, only the entry and exit traffic streams and correlate them through known fingerprinting methods.

For decades, this “global-passive-adversary” attack model was regarded as unrealistically strong and used to justify “conservative” assumptions in formal models.⁸ Unfortunately, this adversarial model is now not only realistic but in fact too weak. With the commercialization and widespread deployment of routers able to perform deep packet inspection and modification, including “man-in-the-middle” attacks against encrypted SSL streams at line rate,⁹ it has become clear to security and privacy professionals that any realistic adversary must be assumed to be active, or able to modify traffic streams at will.

Active attacks. An attacker’s ability to interfere actively in an anonymity network creates an array of new attacks, as

Figure 2. Traffic confirmation, or “fingerprinting,” to de-anonymize onion-routing circuits.



well as ways to strengthen existing traffic-analysis attacks. Figure 3 outlines one type of congestion attack⁷ in which we assume the attacker can directly monitor only one hop of a Tor circuit (such as the traffic from the exit relay to the target Web server). The attacker in this case might be “in the network” or simply own or have compromised the Web server. The attacker wishes to determine the set of relays through which a long-lived circuit owned by a particular user has passed.

The attacker chooses one relay at a time from Tor’s public database and remotely attempts to increase its load by congesting it; for example, the attacker might simulate many ordinary Tor users to launch a denial-of-service attack on the relay. The attacker’s power can be amplified by creating artificially long “flowerpetal” circuits that visit the target relay multiple times, each visit interspersed with a visit to another relay, as in Figure 3. Regardless of how congestion is incurred, it slows all circuits passing through the relay, including the victim circuit, if and only if the circuit passes through the targeted relay. The attacker can thus test whether a particular victim circuit flows through a particular router simply by checking whether the victim circuit’s average throughput (which can be measured at any point along the circuit) slows down during the period of attacker-generated congestion. The attacker repeatedly probes different relays this way until the victim’s entry and middle relays are identified. Finally, the attacker might fully de-anonymize the user by focusing traffic analysis on, or hacking, the user’s entry relay.

Intersection attacks. In most practical uses of anonymous communication, a user typically needs to send not just a single “one-off” message anonymously but a sequence of messages explicitly related and hence inherently linkable to each other; for example, Tor clients must maintain persistent TCP connections and engage in back-and-forth “conversations” with websites in order to support interactive communication, sending new HTTP requests that depend on the Web server’s responses to the client’s previous HTTP requests. It is manifestly obvious, at least to the Web server (and probably



Dissent preserves maximum security provided only that not all of a group’s servers maliciously collude against their clients.



to any eavesdropper who can monitor the connection between the Tor exit relay and the website), which packets comprise the same Web communication session, even if it is not (yet) clear who initiated the session. Further, if the user leaves an anonymous browser window open for an extended period or regularly logs into the same Web-based online email account, an eavesdropper might be able to link many of the user’s browsing sessions together over a long period of time. Even if each message gives the attacker only a small and statistically uncertain amount of information, just slightly narrowing the identity of the anonymous user, combining this information across many observation points at different times rapidly strengthens the attacker’s knowledge and can eventually identify and de-anonymize the target.

In one example of this attack (see Figure 4), an authoritarian government compels its ISPs or cellular carriers to turn over logs of which customers were online and actively using the network during which periods of time. An anonymous dissident posts blog entries to a pseudonymous blog at different points in time. Assume the attacker controls none of the user’s onion relays. Neither does the attacker control the blog server but merely observes the times at which the blog entries appeared and the fact the posts are manifestly linkable to each other, and so can correlate this information with the ISP logs. Perhaps the subject of the blog is official corruption in a particular city, enabling the authoritarian state to guess the dissident lives in that city and narrow attention to a small set of local ISPs. The attacker merely retrieves the sets of users who were online at each time a blog post appeared and intersects those sets. Although many thousands of users may be online at each of these posting times individually, all users other than the dissident in question are likely to have gone offline during at least one of these times (due to normal churn, the partly random comings and goings of most users), allowing the attacker to eliminate them from the victim’s anonymity set. The attacker needs only to “wait and watch” until the dissident has posted enough blog entries, and

the intersection of the online-user sets will shrink to a singleton.

The strength of this attack in practice is amply demonstrated by the fact that similar reasoning is used regularly in law enforcement.¹⁷ When an anonymous bomb threat was posted at Harvard via Tor in December 2013, the FBI caught the student responsible by effectively intersecting the sets of Tor users and Harvard network users at the relevant time. Paula Broadwell, whose extramarital affair with General David Petraeus led to the end of his career as director of the CIA in 2012, was de-anonymized through the equivalent of an intersection attack. De-anonymized in similar fashion were the “High Country Bandits” in 2010, as, per Ars Technica, “... a rather grandiose name for a pair of middle-aged white men who had been knocking down rural banks in northern Arizona and Colorado, grabbing a few thousand dollars from a teller’s cash drawer and sometimes escaping on a stolen all-terrain vehicle.” Intersection attacks also are the foundation of the National Security Agency’s CO-TRAVELER cellphone-location program linking known surveillance targets as their respective cellphones move together from one cell tower to another.

Software exploits and self-identification. No anonymous communication system can succeed if other software the user is running gives away the user’s network location. In an attack against the Tor network detected in August 2013, a number of “hidden services,” or websites with locations protected by Tor and accessible only through Tor, were compromised so as to send malicious JavaScript code to all Tor clients that connected to them (see Figure 5). This JavaScript code exploited a vulnerability in a particular version of Firefox distributed as part of the Tor Browser Bundle. This code effectively “broke out” of the usual JavaScript sandbox and ran native code as part of the browser’s process. This native code then invoked the host operating system to learn the client’s true (de-anonymized) IP address, MAC address, and more, sending them to an attacker-controlled server. The attacker in this case was initially suspected and later confirmed to be the FBI, employing “black hat” hacking techniques to take

down hidden services carrying child pornography and trace their users.

Collective Anonymity in Dissent

As a step toward addressing these challenges, we introduce Dissent, a project

at Yale University that expands the design space and explores starkly contrasting foundations for anonymous communication.

Alternative foundations for anonymity. Quantification and formal analysis

Figure 3. Example congestion-based active attack.

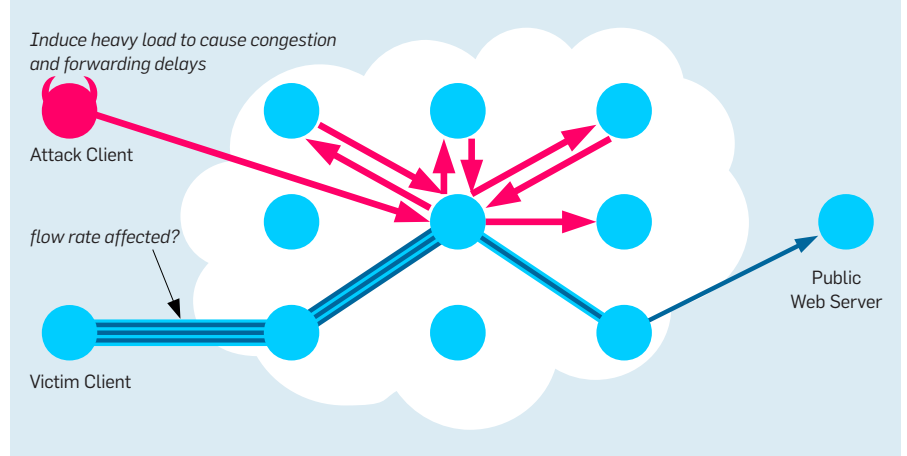


Figure 4. Example intersection attack.

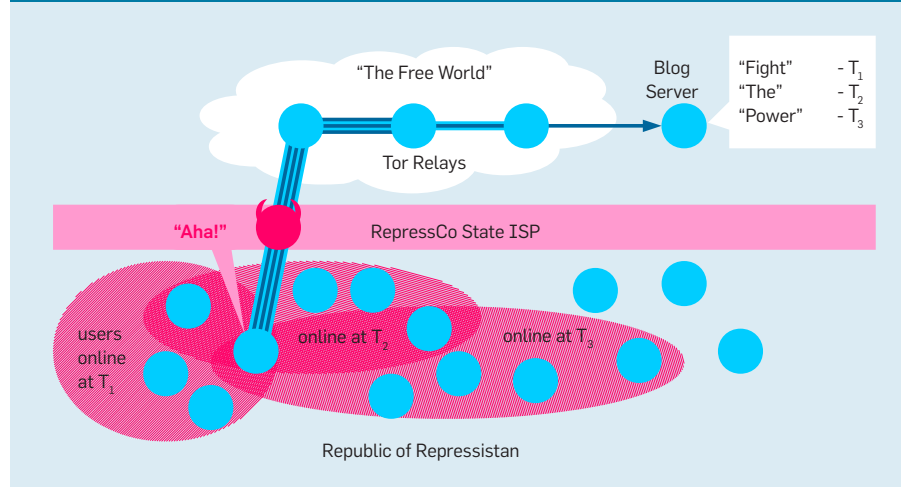
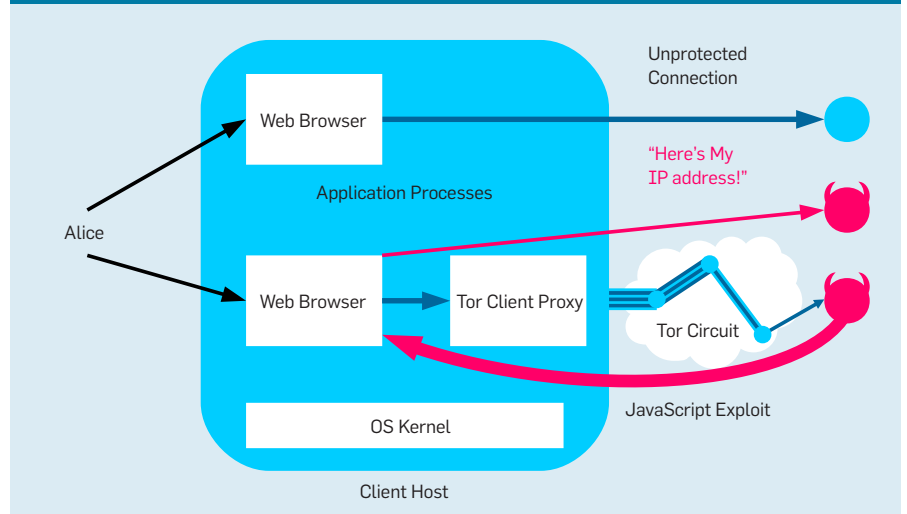


Figure 5. Example software-exploit attack.



of onion routing security under realistic conditions has proved an elusive goal.⁸ Dissent thus builds on alternative anonymity primitives (such as verifiable shuffles and dining cryptographers) with more readily provable properties.

Verifiable shuffles. In a typical cryptographic shuffle, participating nodes play two disjoint roles: a set of n clients with messages to send and a set of m shufflers that randomly permute these messages. Communication proceeds in synchronous rounds. In each, each of the n clients encrypts a single message under m concentric layers of public-key encryption, using each of the m shufflers' public keys, in a standardized order. All n clients send their ciphertexts to the first shuffler, which holds the private key to the outermost layer of encryption in all the clients' ciphertexts. The first shuffler waits until it receives all n clients' ciphertexts, then unwraps this outermost encryption layer, randomly permutes the entire set of ciphertexts, and forwards the permuted batch of n ciphertexts to the next shuffler. Each shuffler in turn unwraps another layer of encryption, permutes the batch of ciphertexts, and then forwards them to the next shuffler. The final shuffler then broadcasts all the fully decrypted cleartexts to all potentially interested recipients.

In an "honest-but-curious" security model in which we assume each shuffler correctly follows the protocol

(without, say, inserting, removing, or modifying any ciphertexts), the output from the last shuffler offers provable anonymity among all non-colluding clients, provided at least one of the shufflers keeps its random permutation secret. Unfortunately, if any of the shufflers is actively dishonest, this anonymity is easily broken. For example, if the first shuffler duplicates the ciphertext of some attacker-chosen client, the attacker may be able to distinguish the victim's cleartext in the shuffle's final output simply by looking for the cleartext that appears twice in the otherwise-anonymized output batch.

A substantial body of work addresses these vulnerabilities to such active attacks. In a "sender-verifiable" shuffle,^{2,4} each client inspects the shuffle's output to ensure its own message was not dropped, modified, or duplicated before allowing the shuffled messages to be fully decrypted and used. More sophisticated and complex provable shuffles (such as one by Neff¹⁵) enable each shuffler to prove to all observers the correctness of its entire shuffle, or that the shuffler's output is a correct permutation of its input, without revealing any information about which permutation it chose.

Both types of verifiable shuffles offer cryptographic guarantees that the process of shuffling reveals no information about which of the n clients submitted a given message appearing

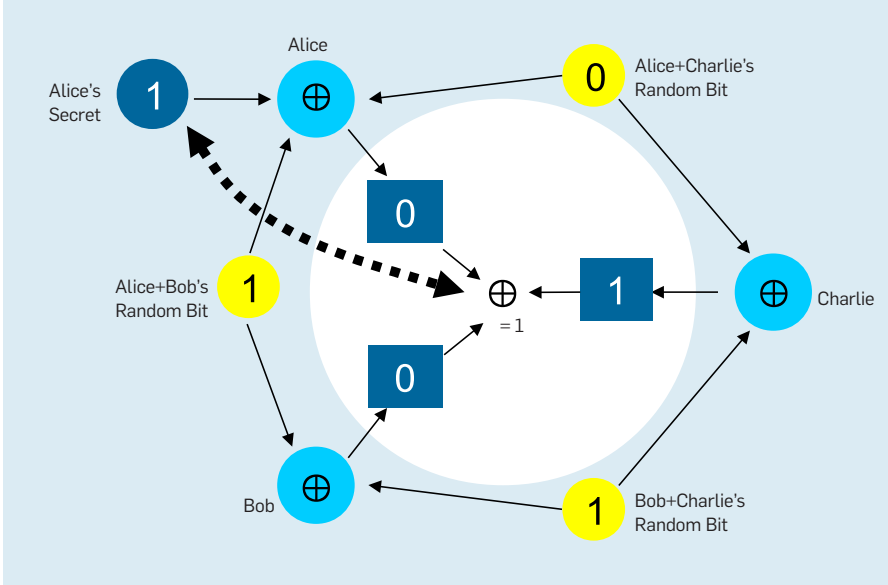
in the shuffled output. Shuffling has the practical disadvantage that the level of security achievable against potentially compromised shufflers depends on the number of shufflers in the path, and multiple shufflers must inherently be placed in sequence to improve security; in essence, latency is inversely proportional to security. The typical cascade arrangement, where all clients send their messages through the same sequence of shufflers at the same time, is most amenable to formal anonymity proofs but exacerbates the performance problem by creating the "worst possible congestion" at each shuffler in succession instead of randomly distributing load across many shufflers as an ad hoc, individualistic onion router network would.

For these reasons, verifiable shuffles may be practical only when high latencies are tolerable and shufflers are well provisioned. One relevant application is electronic voting, for which some shuffle schemes were specifically intended and which might readily tolerate minutes or hours of latency. A second application that arguably fits this model is "anonymous remailers,"⁵ which was popular before onion routing. Practical remailer systems have never, to our knowledge, employed state-of-the-art verifiable shuffles featuring anonymity proofs and were and remain vulnerable to active attacks analogous to the message-duplication attack described earlier.

Dining cryptographers. The only well-studied foundation for anonymity not based on sequential relaying is "dining cryptographers," or DC-nets, invented by Chaum³ in the late 1980s but never used in practical systems until two decades later by Herbiore.¹⁸ Instead of multi-hop message or packet relaying, DC-nets build on information-coding methods.

To illustrate how DC-nets operates, consider Chaum's classic scenario (see Figure 6), in which three cryptographers are dining at a restaurant when the waiter says their meal has been paid for. Suspicious, they wish to learn whether one of their group paid the bill anonymously or NSA agents at the next table paid it. So each adjacent pair of cryptographers flips a coin only the two can see. Each cryptographer XORs the coins to his left and right and writes the

Figure 6. The dining-cryptographers approach to anonymous communication; Alice reveals a one-bit secret to the group, but neither Bob nor Charlie learn which of the other two members sent the message.



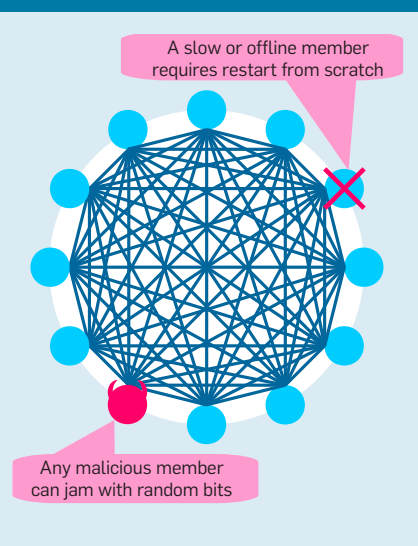
result on a napkin everyone can see—except any cryptographer who paid the bill (Alice in this case), who flips the result of the XOR. The cryptographers then XOR together the values written on all the napkins. Because each coin toss affects the values of exactly two napkins, the effects of the coins cancel out and have no effect on the final result, leaving a 1 if any cryptographer paid the bill (and lied about the XOR) or a 0 if no cryptographer paid. However, a 1 outcome provably reveals no information about which cryptographer paid the bill; Bob and Charlie cannot tell which of the other two cryptographers paid it, unless of course they collude against Alice.

DC-nets generalize to support larger groups and transmission of longer messages. Each pair of cryptographers typically uses Diffie-Hellman key exchange to agree on a shared seed for a standard pseudorandom-bit generator that efficiently produces the many “coin flips” needed to anonymize multi-bit messages. However, while theoretically appealing, DC-nets have not been perceived by anonymous communication tool developers as practical, for at least three reasons (see Figure 7). First, in groups of size N , optimal security normally requires all pairs of cryptographers share coins, yielding complexity $\Omega(N^2)$, both computational and communication. Second, large networks of “peer-to-peer” clients invariably exhibit high churn, with clients going offline at inopportune times; if a DC-nets group member disappears during a round, the results of the round become unusable and must be restarted from scratch. And third, large groups are more likely to be infiltrated by misbehaving members who might wish to block communication, and any member of a basic DC-nets group can trivially—and anonymously—jam all communication simply by transmitting a constant stream of random bits.

Practical dining cryptographers. Utilizing the DC-nets foundation in practical systems requires solving two main challenges: jamming and scalability. Herbivore¹⁸ pioneered exploration of practical solutions to both problems, and the Dissent project continues this work.

The jamming problem. Both Chaum’s original paper³ and many follow-up

Figure 7. Why scaling DC-nets is difficult in practice: worst case $N \times N$ coin-sharing matrix; network churn requires communications rounds to start over; and malicious members can anonymously jam the group.



works studied theoretical solutions to the jamming problem but were complex and to our knowledge never put into practice. Herbivore sidestepped the jamming problem by securely dividing a large peer-to-peer network into many smaller DC-nets groups, enabling participants who find themselves in an unreliable or jammed group to switch groups until they find a functioning one. This design has the advantage of scaling to support arbitrary-size networks, with the downside that participants obtain provable anonymity only within their own group—typically tens of nodes at most—and not guaranteeing anonymity within the larger network. Switching groups to avoid jamming can also introduce weaknesses to more intelligent attackers, who might run many Sybil nodes and selectively jam only groups they cannot compromise completely, all while offering good service in groups in which they have isolated a single “victim” node. The active attacker can thereby “prod” potential victims to switch groups until they land in a completely compromised group.¹

Dissent, the only system since Herbivore to put DC-nets into practice, explores different solutions to these challenges. First, it addresses the jamming problem by implementing accountability mechanisms, allowing the group to revoke the anonymity of any peer found to be attempting to jam commu-

nication maliciously while preserving strong anonymity protection for peers who “play by the rules.” Dissent’s first publicly available version introduced a conceptually simple and clean accountability mechanism that leveraged the verifiable-shuffle primitive discussed earlier, at the cost of requiring a high-latency shuffle between each round of (otherwise more efficient) DC-nets communication. The next version²³ in 2012 introduced a more efficient but complex retroactive-blame mechanism, allowing lower-latency DC-nets rounds to be performed “back-to-back” in the absence of jamming and requiring an expensive shuffle only once per detected jamming attempt.

However, an adversary who manages to infiltrate a group with many malicious nodes could still “sacrifice” them one-by-one to create extended denial-of-service attacks. Addressing this risk, a more recent incarnation of Dissent⁴ replaces the “coins” of classic DC-nets with pseudorandom elliptic-curve group elements, replaces the XOR combining operator with group multiplication, and requires clients to prove their DC-nets ciphertexts correct on submission, using zero-knowledge proofs. To avoid the costs of using elliptic-curve cryptography all the time, Dissent implements a hybrid mode that uses XOR-based DC-nets unless jamming is detected, at which point the system switches to elliptic-curve DC-nets briefly to enable the jamming victim to broadcast an accusation, yielding a more efficient retroactive-blame mechanism.

Scaling and network churn. Even with multiple realistic solutions to the jamming problem now available, DC-nets cannot offer useful anonymity if tools built using DC-nets can guarantee only anonymity-set size of at most tens of members. Herbivore addressed the $N \times N$ communication-complexity problem through a star topology in which a designated member of each group collects other members’ ciphertexts, XORs them together, and broadcasts the results to all members. However, without a general solution to the network churn and jamming problems, both Herbivore and the first version of Dissent were limited in practice to small anonymity sets comprising at most tens of nodes.

Addressing churn and scaling DC-nets further, Dissent now adopts a client/multi-server model with trust split across multiple servers, preferably administered independently. No single server is trusted; in fact, Dissent preserves full security provided only that not all of a group's servers maliciously collude against their clients. The clients need not know or guess which server is trustworthy but must trust only that at least one trustworthy server exists.

When a Dissent group is formed, the group's creator defines both the set of servers to support the group and the client-admission policy; in the simplest case, the policy is simply a list of public keys representing group members. Dissent servers thus play a role analogous to relays in Tor, serving to support the anonymity needs of many different clients and groups. Like Tor relays, the Dissent servers supporting a new group might be chosen automatically from a public directory of available servers to balance load. Choosing the servers for each group from a larger "cloud" of available servers in this way enables, in principle, Dissent's design to support an arbitrary number of groups, though the degree to which an individual group scales may be more limited. If a particular logical group becomes extremely popular, Herbivore's technique of splitting a large group into multiple smaller groups may be applicable. Our current Dissent prototype does not yet implement either a directory service or Herbivore-style subdivision of large networks.

While individual groups do not scale indefinitely, Dissent exploits its client/multi-server architecture to make groups scale two orders of magnitude beyond prior DC-nets designs.²³ Clients no longer share secret "coins" directly with other clients but only with each of the group's servers, as in Figure 8. Since the number of servers in each group is typically small (such as three to five, comparable to the number of Tor relays supporting a circuit), the number of pseudorandom strings each client must compute is substantially reduced. However, this change does not reduce anonymity, subject to Dissent's assumption that at least one server is honest. Chaum's DC-nets security proof³ ensures ideal anonymity, provided all



Public demand for anonymity online may intensify as a result of the ongoing surveillance scandal, thereby providing an opportunity to deploy new anonymity tools.



honest nodes are connected through the coin-sharing graph; Dissent satisfies this requirement, as the one honest server assumed to exist shares coins directly with all honest clients.

More important in practice, Dissent's client/multi-server coin-sharing design addresses network churn by making the composition of client ciphertexts independent of the set of other clients online in a given round. The servers set a deadline, and all clients currently online must submit their ciphertexts by that deadline or risk being "left out" of the round. Unlike prior DC-nets designs, if some Dissent clients miss the deadline, the other clients' ciphertexts remain usable. The servers merely adjust the set of client/server-shared secrets they use to compute their server-side DC-net ciphertexts.

Because each client's ciphertext depends on secrets it shares with all servers, no client's ciphertext can be used or decrypted unless all servers agree on the same set of online clients in the round and produce correct server-side ciphertexts based on that agreement. Malicious servers can do no worse than corrupt a round, and cannot de-anonymize clients except by colluding with all other servers.

How Dissent addresses attacks. Here, we outline how Dissent addresses the types of attacks discussed earlier.

Global traffic analysis. Dissent builds on anonymity primitives that have formal security proofs in a model where the attacker is assumed to monitor all network traffic sent among all participating nodes but cannot break the encryption. We have extended these formal security proofs to cover the first version of the full Dissent protocol;¹⁹ formal analysis of subsequent versions is in progress. Although verifiable shuffles differ from DC-nets in their details, both approaches share one key property that enables formal anonymity proofs: All participants act collectively under a common "control plane" rather than individually as in an ad hoc onion routing system; for example, they send identical amounts of network traffic in each round, though amounts and allocations may vary from round to round.

Active attacks. One countermeasure to traffic analysis in an onion router is

to “pad” connections to a common bit rate. While padding may limit passive traffic analysis, it often fails against active attacks, for reasons outlined in Figure 9. Suppose a set of onion router users pad the traffic they send to a common rate, but a compromised upstream ISP wishes to “mark” or “stain” each client’s traffic by delaying packets with a distinctive timing pattern. An onion router network that handles each client’s circuit individually preserves this recognizable timing pattern (with some noise) as it passes through the relays, at which point the attacker might recognize the timing pattern at the egress more readily than would be feasible with a traffic-confirmation attack alone. Active attacks also need not mark circuits solely through timing. A sustained attack deployed against Tor starting in January 2014 exploited another subtle protocol side-channel to mark and correlate circuits, going undetected for five months before being discovered by Tor project members on July 4, 2014 and subsequently thwarted (<https://blog.torproject.org/blog/tor-security-advisory-relay-early-traffic-confirmation-attack>).

In contrast, the collective-anonymity primitives underlying Herbivore and Dissent structurally keep the clients comprising an anonymity set in “lock-step” under the direction of a common, collective control plane. As in the popu-

lar children’s game “Simon Says,” participants transmit when and how much the collective control plane tells them to transmit. A client’s network-visible communication behavior does not leave a trackable fingerprint or stain, even under active attacks, because the client’s network-visible behavior depends only on this anonymized, collective control state; that is, a client’s visible behavior never depends directly on individual client state. Further, the Dissent servers implementing this collective control plane do not know which user owns which pseudonym or DC-nets transmission slot and thus cannot leak that information through their decisions, even accidentally.

Contrary to the intuition that defense against global traffic analysis and active attacks requires padding

traffic to a constant rate, Dissent’s control plane can adapt flow rates to client demand by scheduling future rounds based on (public) results from prior rounds. For example, the control-plane scheduler dynamically allocates DC-nets transmission bandwidth to pseudonyms that in prior rounds anonymously indicated a desire to transmit and hence avoids wasting network bandwidth or computation effort when no one has anything useful to say. Aqua, a project launched in 2013 at the Max Planck Institute for Software Systems in Germany to strengthen onion router security, employs a similar collective-control philosophy to normalize flow rates dynamically across an anonymity set.¹³ In this way, a collective control plane can in principle not only protect against

Figure 8. Improving scalability and churn resistance through asymmetric, client/server DC-nets architecture.

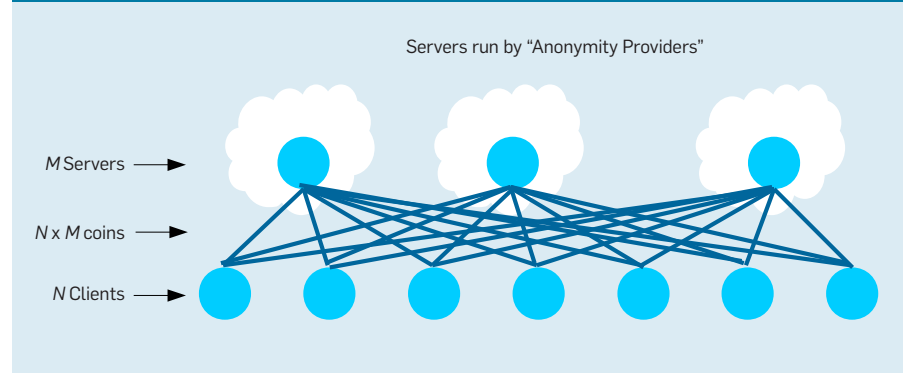
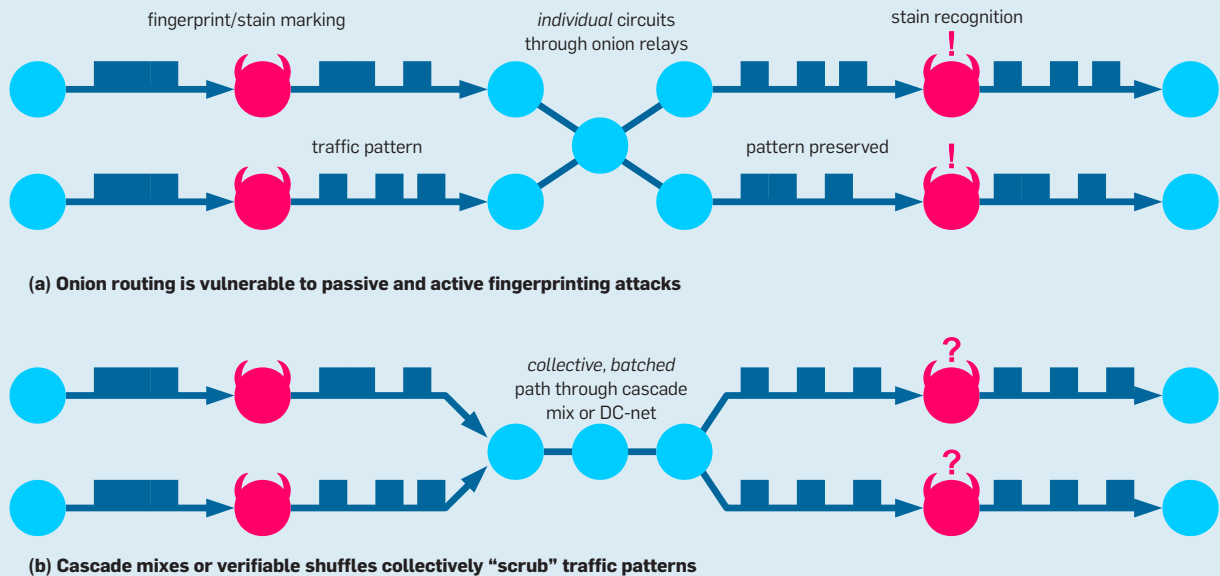


Figure 9. Fingerprinting or staining attacks.



both passive and active attacks but ironically also improve efficiency over padding traffic to a constant bit rate.

Intersection attacks. While the power and general applicability of intersection attacks have been studied extensively over the past decade, there is scant work on actually building mechanisms to protect users of practical systems against intersection attacks. The nearest precedents we are aware of suggest only that traffic padding may make intersection attacks more difficult, falling short of quantifying or controlling the effectiveness of such attacks.¹⁴ To the best of our knowledge, traffic padding proposals have never been implemented in deployed tools, in part because there is no obvious way to measure how much protection against intersection attacks a given padding scheme will provide in a real environment.

Dissent is the first anonymity system designed with mechanisms to measure potential vulnerability to intersection attacks, using formally grounded but plausibly realistic metrics, and offers users active control over anonymity loss under intersection attacks.²⁵ Dissent implements two different anonymity metrics: “possinymity,” a probabilistic measurement of anonymity-set size motivated by “plausible-denial-

ity” arguments, and “indinymity,” an indistinguishability metric effective against stronger adversaries that may make probabilistic “guesses” via statistical disclosure.¹⁴

Users may set policies for long-lived pseudonyms, limiting the rate measured possinymity or indinymity may be lost or setting a threshold below which these metrics are not allowed to fall. Dissent’s collective control plane enforces these policies in essence by detecting when allowing a communication round to proceed might reduce a pseudonym’s possinymity or indinymity “too much” and in response suppressing or delaying communication temporarily.

The control plane can compute these metrics and enforce these policies even though its logic does not “know” which user actually owns each pseudonym. The downside is that employing these controls to resist intersection attacks can reduce the responsiveness, availability, and/or lifetime of a pseudonym. This cost reflects a fundamental trade-off between anonymity and availability.

Software exploits and self-identification. No anonymity protocol can by itself prevent de-anonymization through software exploits or user self-identification. Nevertheless, the Dis-

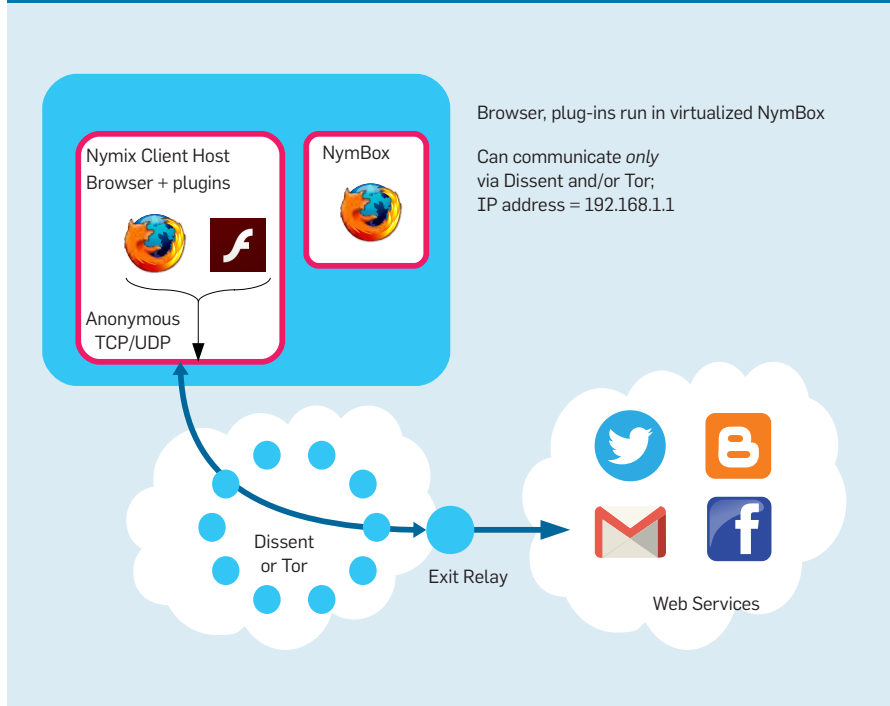
sent project is exploring system-level solutions through Nymix, a prototype USB-bootable Linux distribution that employs virtual machines (VMs) to improve resistance to exploits.²⁴

Nymix runs anonymity-client software (either Tor or Dissent) in the platform’s host operating system but isolates the browser and any plug-ins and other extensions it may depend on in a separate “guest VM,” as in Figure 10. No software in this guest VM is given access to information about the physical host OS or its network configuration; for example, the guest VM sees only a standard private (NATted) IP address (such as 192.168.1.1) and the fake MAC address of a virtual device. Even native code injected by a browser exploit (such as the one detected in August 2013 affecting the Windows version of the Tor Browser Bundle) would thus not be able to “leak” the client’s IP address without also breaking out of the VM. Escaping the VM as well may be possible, but the additional barrier increases attack difficulty.

Nymix binds guest-VM state instances to pseudonyms managed by the anonymity layer, enabling users to launch multiple simultaneous pseudonyms in different VMs, or “NymBoxes,” as in Figure 10. Nymix securely discards all pseudonym state embodied in a NymBox when appropriate to minimize the user’s long-term exposure to intersection attacks. This binding of pseudonyms to VMs makes it easy for the user to maintain state related to the context of one logical pseudonym (such as Web cookies and open logins) while offering stronger protection against the user’s accidentally linking different pseudonym VMs, because they appear as entirely separate OS environments, not just as different browser windows or tabs.

To reduce the risk of self-identification, Nymix allows the user to “move” data between non-anonymous contexts (such as personal .jpg photos stored on the host OS) and pseudonym-VM contexts only through a quarantine file system “drop box.” All files the user moves across browsing contexts in this way undergo a suite of tests to identify possibly compromising information (such as “exchangeable image file format,” or Exif, meta-

Figure 10. Using per-pseudonym virtual machines, or NymBoxes, to harden the client operating system against software exploits, staining, and self-identification.



data within .jpg files). The quarantine system alerts users of any detected compromise risks, giving them the opportunity to scrub the file or decide not to transfer it at all. While all these defenses are inherently “soft” because there is only so much privacy-tool developers can do to prevent users from shooting themselves in the foot, Nymix combines these VM-based isolation and structuring principles to make it easier for users to make appropriate and well-informed uses of today’s, as well as tomorrow’s, anonymity tools.

Challenges and Future Work

Dissent takes a few important steps toward developing a collective approach to anonymous communication, but many practical challenges remain.

First, while DC-nets now scale to thousands of users, to support a global user population DC-nets must scale to hundreds of thousands of users or more. One approach is to combine Dissent’s scaling techniques with those of HerbiVore¹⁸ by dividing large anonymity networks into manageable anonymity sets (such as hundreds or thousands of nodes), balancing performance against anonymity guarantees. A second approach is to use small, localized Dissent clusters that already offer performance adequate for interactive Web browsing^{23,24} as a decentralized implementation for the crucial entry-relay role in a Tor circuit.²⁰ Much of a Tor user’s security depends on the user’s entry relay’s being uncompromised;¹² replacing this single point of failure with a Dissent group could distribute the user’s trust among the members of the group and further protect traffic between the user and the Tor relays from traffic analysis by “last mile” ISP adversaries.

Second, while Dissent can measure vulnerability to intersection attack and control anonymity loss,²⁵ it cannot also ensure availability if users exhibit high churn and individualistic “every user for themselves” behavior. Securing long-lived pseudonyms may be feasible only in applications that incentivize users to keep communication devices online constantly, even if at low rates of activity, to reduce anonymity decay caused by churn. Further, robust intersection-attack resistance may be practical only in applications designed to encourage users to act collectively

rather than individually and optimized for these collective uses.

Applications in which users cooperatively produce collective information “feeds” consumed by many other users may be well suited to Dissent’s collective anonymity model, including the interaction models of Internet relay chat, forums like Slashdot and Twitter, and applications supporting voting, deliberating, or “town hall” meetings. Given the close relationship between collective deliberation and the foundations of democracy and freedom of speech, such applications may also represent some of the most socially important use cases for online anonymity. But how best to support and incentivize cooperative behavior remains an important open problem.

Finally, large anonymity sets clearly require widespread public demand for anonymity. Tor’s two-million daily users are dwarfed in number by the number of users of Google, Facebook, Yahoo!, and other services that do not provide anonymity—and cannot provide it, because their business models depend crucially on exploiting personal information. Public demand for anonymity online may intensify as a result of the ongoing surveillance scandal, thereby providing an opportunity to deploy new anonymity tools.

Acknowledgments

This material is based on work supported by the Defense Advanced Research Projects Agency and SPAWAR Systems Center Pacific, contract no. N66001-11-C-4018. 

References

1. Borisov, N., Danezis, G., Mittal, P., and Tabriz, P. Denial of service or denial of security? How attacks on reliability can compromise anonymity. In *Proceedings of the 14th ACM Conference on Computer and Communications Security* (Alexandria, VA, Oct. 29–Nov. 2). ACM Press, New York, 2007.
2. Brickell, J. and Shmatikov, V. Efficient anonymity-preserving data collection. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Philadelphia, PA, Aug. 20–23). ACM Press, New York, 2006.
3. Chaum, D. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology* 1, 1 (1988), 65–75.
4. Corrigan-Gibbs, H., Wolinsky, D.I., and Ford, B. Proactively accountable anonymous messaging in Verdict. In *Proceedings of the 22nd USENIX Security Symposium* (Washington, D.C., Aug. 14–16). USENIX Association, Berkeley, CA, 2013.
5. Danezis, G., Dingleline, R., and Mathewson, N. Mixinion: Design of a type III anonymous remailer protocol. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy* (Oakland, CA, May 11–14). IEEE Computer Society Press, Los Alamitos, CA, 2003.
6. Dingleline, R. and Murdoch, S.J. Performance improvements on Tor, or why Tor is slow and what

we’re going to do about it. Presented at DEFCON 17 (Las Vegas, NV, July 30–Aug. 2, 2009); <https://svn.torproject.org/svn/projects/roadmaps/2009-03-11-performance.pdf>

7. Evans, N.S., Dingleline, R., and Grothoff, C. A practical congestion attack on Tor using long paths. In *Proceedings of the 18th USENIX Security Symposium* (Montreal, Canada, Aug. 10–14). USENIX Association, Berkeley, CA, 2009.
8. Feigenbaum, J., Johnson, A., and Syverson, P. Probabilistic analysis of onion routing in a black-box model. *ACM Transactions on Information and System Security* 15, 3 (2012), article 14.
9. Gallagher, R. New Snowden documents show NSA deemed Google networks a ‘target.’ *Slate* (Sept. 9, 2013).
10. Gellman, B., Timberg, C., and Rich, S. Secret NSA documents show campaign against Tor encrypted network. *The Washington Post* (Oct. 4, 2013).
11. Goldschlag, D.M., Reed, M.G., and Syverson, P.F. Hiding routing information. In *Proceedings of the First International Workshop on Information Hiding* (Cambridge, U.K., May 30–June 1). Springer, Berlin, 1996.
12. Johnson, A., Wacek, C., Jansen, R., Sherr, M., and Syverson, P. Users get routed: Traffic correlation on Tor by realistic adversaries. In *Proceedings of the 20th ACM Conference on Computer and Communications Security* (Berlin, Germany, Nov. 4–8). ACM Press, New York, 2013.
13. Le Blond, S., Choffnes, D., Zhou, W., Druschel, P., Ballani, H., and Francis, P. Towards efficient traffic-analysis resistant anonymity networks. In *Proceedings of ACM SIGCOMM 2013* (Hong Kong, China, Aug. 12–16). ACM Press, New York, 2013.
14. Mathewson, N. and Dingleline, R. Practical traffic analysis: Extending and resisting statistical disclosure. In *Proceedings of the Fourth Workshop on Privacy Enhancing Technologies* (Toronto, Canada, May 24–26). Springer, Berlin, 2004.
15. Neff, C.A. A verifiable secret shuffle and its application to e-voting. In *Proceedings of the Eighth ACM Conference on Computer and Communications Security* (Philadelphia, PA, Nov. 6–8). ACM Press, New York, 2001.
16. Risen, J. and Poitras, L. NSA report outlined goals for more power. *The New York Times* (Nov. 22, 2013).
17. Segal, A., Ford, B., and Feigenbaum, J. Catching bandits and only bandits: Privacy-preserving intersection warrants for lawful surveillance. In *Proceedings of the Fourth USENIX Workshop on Free and Open Communications on the Internet* (San Diego, CA, Aug. 18). USENIX Association, Berkeley, CA, 2014.
18. Sirer, E.G., Goel, S., Robson, M., and Engin, D. Eluding carnivores: File sharing with strong anonymity. In *Proceedings of the 11th ACM SIGOPS European Workshop* (Leuven, Belgium, Sept. 19–22). ACM Press, New York, 2004.
19. Syta, E., Johnson, A., Corrigan-Gibbs, H., Weng, S.-H., Wolinsky, D.I., and Ford, B. Security analysis of accountable anonymity in Dissent. *ACM Transactions on Information and System Security* 17, 1 (2014), article 4.
20. Tor. *Anonymity Online*; <https://www.torproject.org>
21. Tor. Metrics portal; <http://metrics.torproject.org>
22. Watts, R. JK Rowling unmasked as author of acclaimed detective novel. *The Telegraph* (July 13, 2013).
23. Wolinsky, D.I., Corrigan-Gibbs, H., Johnson, A., and Ford, B. Dissent in numbers: Making strong anonymity scale. In *Proceedings of the 10th USENIX Symposium on Operating Systems Design and Implementation* (Hollywood, CA, Oct. 8–10). USENIX Association, Berkeley, CA, 2012.
24. Wolinsky, D.I., Jackowitz, D., and Ford, B. Managing NymBoxes for identity and tracking protection. In *Proceedings of the 2014 Conference on Timely Results in Operating Systems* (Broomfield, CO, Oct. 5). USENIX Association, Berkeley, CA, 2014.
25. Wolinsky, D.I., Syta, E., and Ford, B. Hang with your buddies to resist intersection attacks. In *Proceedings of the 20th ACM Conference on Computer and Communications Security* (Berlin, Germany, Nov. 4–8). ACM Press, New York, 2013.

Joan Feigenbaum (joan.feigenbaum@yale.edu) is the department chair and Grace Murray Hopper Professor of Computer Science at Yale University, New Haven, CT.

Bryan Ford (bryan.ford@epfl.ch) is an associate professor of computer and communication sciences at the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland.

Copyright held by authors.

DOI:10.1145/2714560

This framework addresses the environmental dimension of software performance, as applied here by a paper mill and a car-sharing service.

**BY PATRICIA LAGO, SEDEF AKINLI KOÇAK,
IVICA CRNKOVIC, AND BIRGIT PENZENSTADLER**

Framing Sustainability as a Property of Software Quality

SUSTAINABILITY IS DEFINED as the “capacity to endure”³⁴ and “preserve the function of a system over an extended period of time.”¹³ Discussing sustainability consequently requires a concrete system (such as a specific software system) or a specific software-intensive system. Analysis of the sustainability of a specific software system requires software developers weigh four major dimensions of sustainability—economic, social, environmental, and technical—affecting their related trade-offs.³²

The first three stem from the Brundtland report,⁴ whereas technical is added for software-intensive systems²⁷ at a level of abstraction closer to implementation. The economic dimension is concerned with preserving

capital and value. The social dimension is concerned with maintaining communities. The environmental dimension seeks to improve human welfare by protecting natural resources. And the technical dimension is concerned with supporting long-term use and evolution of software-intensive systems. Sustainability is achievable only when accounting for all dimensions. Including the environmental dimension makes it possible to aim at dematerializing production and consumption processes to save natural resources.¹² Connections among the four dimensions involve different dependencies and stakeholders.^{28,31} Potential conflicts among stakeholder interests means software developers must understand the relationships among goals of the four dimensions.

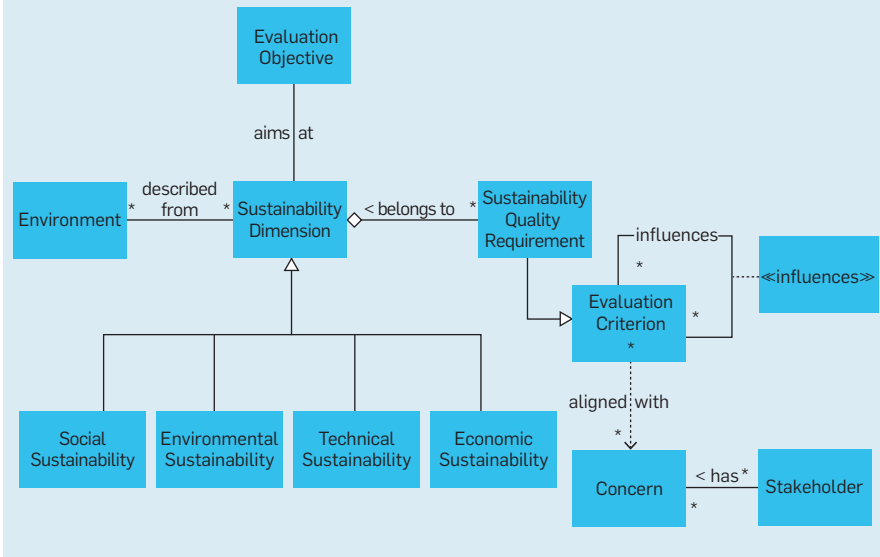
The shortcoming of current software engineering practice with regard to sustainability is that the technical and economic dimensions are taken into account while the environmental and social dimensions are not. The question we address here is how these concepts relate to software and how to break down the respective concerns into software-quality requirements. We focus on the (currently neglected) environmental dimension and its relation to the other dimensions. While most efforts in environmental sustainability through software have focused on energy efficiency, we tie the concept of environmental sustainability to other sustainability dimensions of a software system, particularly to ad-

» key insights

- The sustainability analysis framework enables software developers to specifically consider environmental and social dimensions relative to technical and economic dimensions.
- Sustainability requirements and concerns will increase system scope, requiring extended analysis during requirements engineering.
- The framework helps draw a more comprehensive picture of the relevant quality dimensions and, as a result, improve decision making.



Figure 1. Framework for sustainability software-quality requirements.



dress second-order effects,¹³ or those of a software system in its operational context, as with, say, how a car-sharing service used by many users over a number of years affects the surrounding environment.

Our contribution is a sustainability analysis framework that aids practitioners exploring software qualities related to the four dimensions and explicitly representing dependencies among the dimensions. To illustrate the application of this framework we offer two case-study examples from different domains.

Sustainability Analysis Framework

The framework aims to capture the relevant qualities that characterize sustainability concerns of software systems, helping identify how these qualities influence each other with respect to the different aspects of sustainability (see the sidebar “Software Sustainability”). Software qualities as nonfunctional properties have been studied and adopted in software engineering. In particular, various methods for quality evaluation in software architecture have been defined to support holistic reasoning and decision making that involve software, hardware, human, and system elements. We exploited this holistic approach, defining our framework by extending an existing model, the *Third Working Draft of ISO/IEC 42030 Architecture Evaluation*,¹⁴ as outlined in Figure 1. The blue boxes denote generalized pre-

existing components from the working draft. While the draft specifically targets evaluations, the potential context of the framework is broader, embracing any activity that relies on a sound representation of qualities, including requirements engineering, design decision making, trade-off analyses, and quality assessment.

The following paragraphs describe the dimensions used in the framework to characterize sustainability in the context of software-intensive systems:

Social sustainability. Social sustainability focuses on ensuring current and future generations have the same or greater access to social resources by pursuing generational equity. For software-intensive systems, it encompasses the direct support of social communities in any domain, as well as activities or processes that indirectly create benefits for social communities;

Environmental sustainability. Environmental sustainability aims to improve human welfare while protecting natural resources; for software-intensive systems, this means addressing ecological requirements, including energy efficiency and creation of ecological awareness; and

Technical sustainability. Technical sustainability addresses the long-term use of software-intensive systems and their appropriate evolution in a constantly changing execution environment; and

Economic sustainability. Economic sustainability focuses on preserving

capital and financial value.

An evaluation criterion can be a quality requirement, as in Figure 1. In particular, as we focus on characterizing sustainability-related software qualities, we address how quality requirements relate to sustainability, or “sustainability quality requirements.” In this context, requirements could include both traditional quality requirements (such as performance, usability, security, and maintainability) and sustainability-related requirements (such as energy efficiency).

Whenever we specifically target sustainability, as in Figure 1, where the association aims to link the evaluation objective to the sustainability dimension, software developers must resolve trade-offs among the various qualities classified as belonging to each of the four dimensions. In particular, traditional software decision making considers trade-offs either between different technical sustainability criteria (such as performance versus availability) or between technical sustainability criteria and economic sustainability criteria (such as performance versus costs). In contrast, sustainability-related software decision making involves trade-offs between environmental sustainability criteria (such as energy efficiency) and social, economic, and technical sustainability criteria.

To frame software qualities this way we position them in the four sustainability dimensions and relate them to the concerns of the relevant stakeholders. For the sake of simplicity, this information is not included in the case-study examples, though the description of a paper-mill control system refers to three main stakeholders: surrounding community and society at large (concerned about environmental sustainability like forest sustainability); customers (concerned about economic sustainability like production savings expressing productivity and economic value creation); and producing organization, including managers and engineers (concerned about technical sustainability like optimization of configurability and performance).

Moreover, interdependent quality requirements may influence one another, as in association/association-class influences among sustainability quality requirements; for example, in

the paper-mill control system (see Figure 2), performance and energy savings could influence each other, while increasing performance could demand more resources that consume more power and ultimately have a negative effect on energy savings. Using our framework to make these influences explicit helps designers of software-intensive systems appreciate the importance of the various qualities.

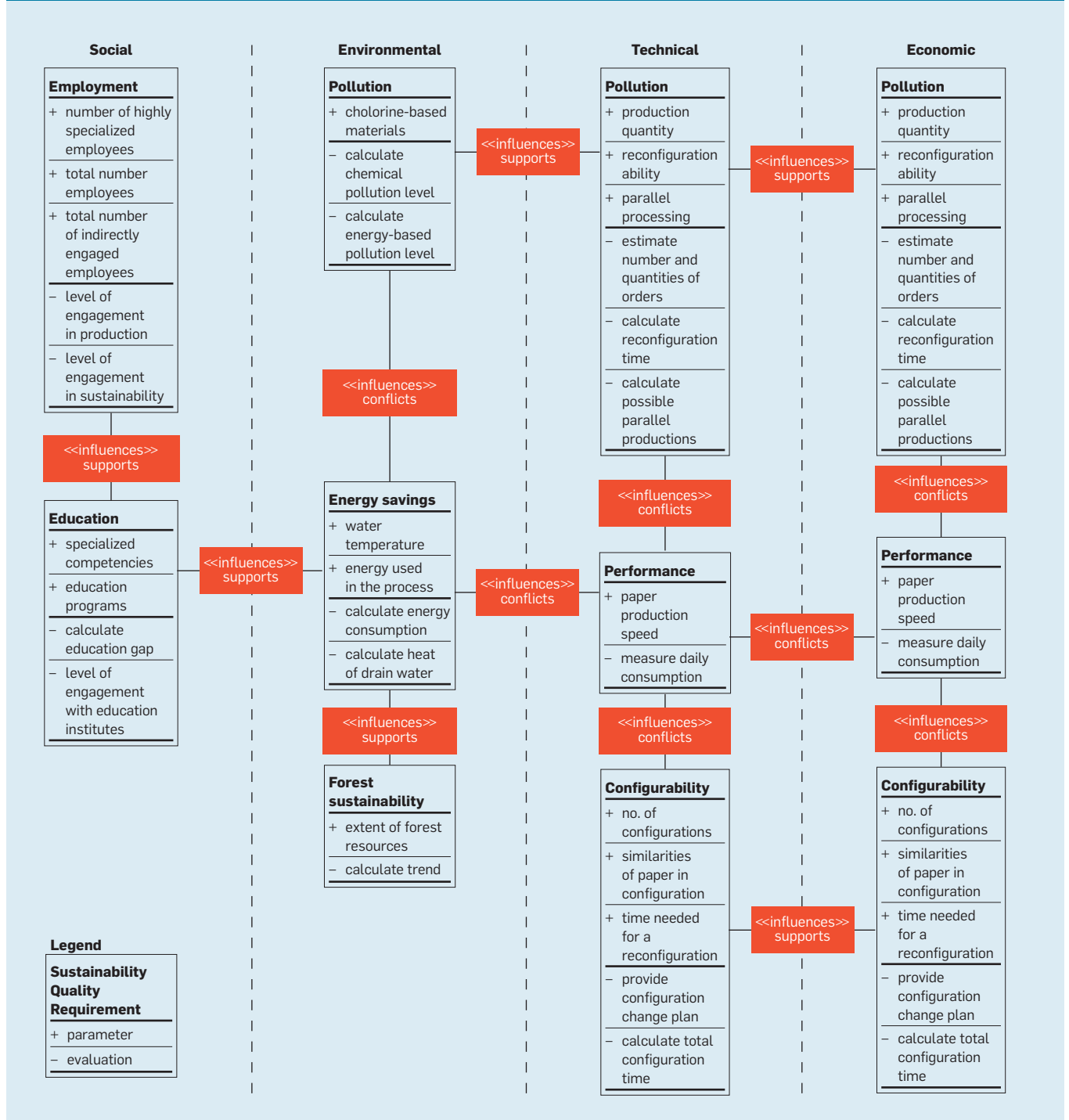
In addition, the trade-offs software developers make among qualities change depending on stakeholders (such as business customers, vendors, and society at large). If a company's main stakeholder is a vendor, performance probably wins over energy savings; the opposite is probably true if the stakeholders are consumers. Keeping track of the elements captured by the framework is thus crucial for rea-

soning about the trade-offs to be made among the various qualities in the four sustainability dimensions.

Examples

We show the applicability of the sustainability analysis framework through examples. For each, we briefly introduce the domain, then discuss its sustainability qualities and their interdependencies. We illustrate the

Figure 2. Sustainability quality requirements: Paper-mill control system.



Software Sustainability

The past few years has seen the transformation of the role of IT in sustainability due to rising demand for energy and increasing use of IT systems and potentially negative effects on the environment. As outlined by Gartner analyst Tratz-Ryan,³⁶ industry is moving toward sustainability to enhance compliance, operational efficiency, and performance, suggesting achieving sustainability objectives should involve IT integration, providing efficiency, performance, and business processes. While industries are integrating IT-enabled solutions, they must also integrate sustainability programs, addressing lack of awareness of emissions reduction and potential financial savings through IT, lack of robust policies for addressing climate change, and lack of frameworks, systems, tools, and practices for decision support and connecting sustainability performance to economic performance.⁹

As the IT industry becomes aware of sustainability, the software-engineering research community has begun paying attention to sustainability, as demonstrated by an increasing number of publications, empirical studies, and conferences. Surveys of published studies^{25,29} show over 50% of those on sustainability in software engineering were published between 2010 and 2012, indicating the emergence of the topic in the software-development community. Software technology can help systems improve their energy efficiency, streamline processes, and adapt to changes in the environment. There is a rich body of knowledge regarding energy estimation¹¹ and optimization (such as efficient algorithms) and tools and methods to measure energy efficiency,^{15,21} particularly for mobile devices.⁷

Researchers often rely on estimates or focus on hardware rather than on software. They increasingly focus on energy efficiency as an objective of the software-development life cycle and related development tools and methodologies. In 2014, Kalaitzoglou et al.¹⁶ developed a practical evaluation model that could serve as a method for evaluating the energy efficiency of software applications.

These energy-related studies emphasize the environmental dimension of sustainability. The other dimensions, as related to software, are also being discussed; for example in 2005, Tate³⁵ characterized sustainable software engineering as “the ability to react rapidly to any change in the business or technical environment” but considered only financial aspects of sustainability. Mahaux et al.²² analyzed the use processes of a software system with respect to social and environmental aspects of sustainability. Naumann et al.²⁴ identified a lack of models and descriptions covering the spectrum of software aspects of sustainability. Razavian et al.³² applied the four-dimensional sustainability model to the services and conflicts among dimensions. More concrete initiatives are emerging in industrial practice.¹⁰

All related studies help build awareness of sustainability in software engineering. Our own next step is to create best practices and guidance by applying definitions, frameworks, and models to case studies. Our framework is thus a means for developing software sustainability by including all four dimensions of sustainability—economic, social, environmental, and technical—while our case studies could help software developers address the challenges of sustainability practices in software engineering.

Software quality and sustainability. Various systems, including energy, management, and computer, target sustainability as a quality objective. Models, tools, and metrics/indicators have been developed to instrument systems for sustainability assessment. A 2013 survey by Lago et al.¹⁸ on green software metrics found metrics are limited to energy consumption, while models to assess green software qualities are lacking. Mocigemba²³ defined a sustainable computing model focusing on product, production, and consumption-process assessments for both hardware and software. And Afgan¹ introduced a multi-criteria assessment method, with economic, environmental, and social indicators, as a way to assess energy systems as proxy for sustainable development. Other preliminary initiatives have investigated how to define, measure, and assess sustainability as an attribute of software quality.^{2,18,26} In general, these efforts point to the multidimensional nature of sustainability and the need for an interdisciplinary approach.

The quality models introduced by the International Organization for Standardization (<http://www.iso.org>)—ISO/9126 and ISO/IEC 25010—do not (yet) consider sustainability a quality property of software development. However, the working group on software architecture (WG42, working on ISO/IEC 42030) is considering including Kern et al.¹⁷ who developed a quality model for green software that refers to quality factors from ISO/IEC 25000 based on direct and indirect software-related criteria. Calero et al.,⁵ who considered sustainability in 2013 as a new factor affecting software quality, presented a quality model based on ISO/25010. In a 2014 study, Akinli Kocak et al.³ evaluated product quality and environmental criteria within a decision framework, providing a trade-off analysis among the criteria. Studies from before Akinli Kocak et al.³ covered the relations between software quality and sustainability, highlighting that both product and use qualities should be considered when assessing software sustainability. However, no study has specifically investigated the multidimensionality of sustainability and the trade-off among the dimensions in software engineering practice. Sustainability-analysis frameworks are beginning to appear in software-engineering research.^{30,31} Our work, as discussed here, is a first step toward emphasizing the environmental dimension generally neglected in other studies.

framework’s added value with various aspects of business sustainability: stakeholders (in the first case) and specialized influences relations between qualities (in the second case). The granularity of requirements ranges from coarse-grain high-level goals to fine-grain detailed system requirements. These case-study examples are at the high-level end of this spectrum (see van Lamsweerde²⁰). Figures 2 and 3 emphasize several framework elements: sustainability quality requirements (for which we detail parameters and metrics to capture quality levels); their influences and interdependencies; and the sustainability dimension they belong to (represented as “swimlanes”). In the figures we do not propose a new notation but the approach we suggest for capturing the relations among the four sustainability dimensions. For formalizing and modeling in more detail, the notations proposed by Chung et al.⁶ are also useful. Here, we use a simple notation based on Unified Modeling Language class diagrams.

Paper-mill control system. The worldwide paper-production industry is an example of successful sustainability improvement through advances in technical and economic solutions.⁸ A typical plant control system (PCS) some 30 years ago would have involved a paper-production cycle of several days. The energy consumption would have been very high (though the cost of electricity was lower, the energy costs were three times more per ton of pulp than today); so was the pollution, mostly in a form of water polluted by chlorine compounds (water pollution at the time had just started to be a public-policy issue). A PCS would manage the entire process through a few hundred sensors and actuators. A typical plant would employ from 2,000–3,000 people, with a considerable number of them relatively uneducated, and several tens of experts who would optimize the process with respect to production quality through their experience. A PCS today can handle several hundred thousand signals while reducing the production cycle to an hour while lowering the environmental impact significantly; for example, water consumption of 200–300 cubic meters

per ton of pulp in 1970 decreased to less than 50 cubic meters per ton and in some mills below even 10 cubic meters per ton. The number of employees in Swedish plants (Sweden is a major pulp and paper producer) decreased over 75%, though their qualifications increased; today, over 50% of employees are highly qualified engineers and technical specialists. Production in such plants has increased dramatically, by at least 10 times in the past 30 years.^a The main concern for mill owners today is energy savings, including energy for the technological process (such as in cooking paper pulp) and energy for the PCS. This gives environmentally sustainable software a double role: decrease energy consumption of the PCS itself, which is distributed and complex, with many devices, and decrease energy consumption of the en-

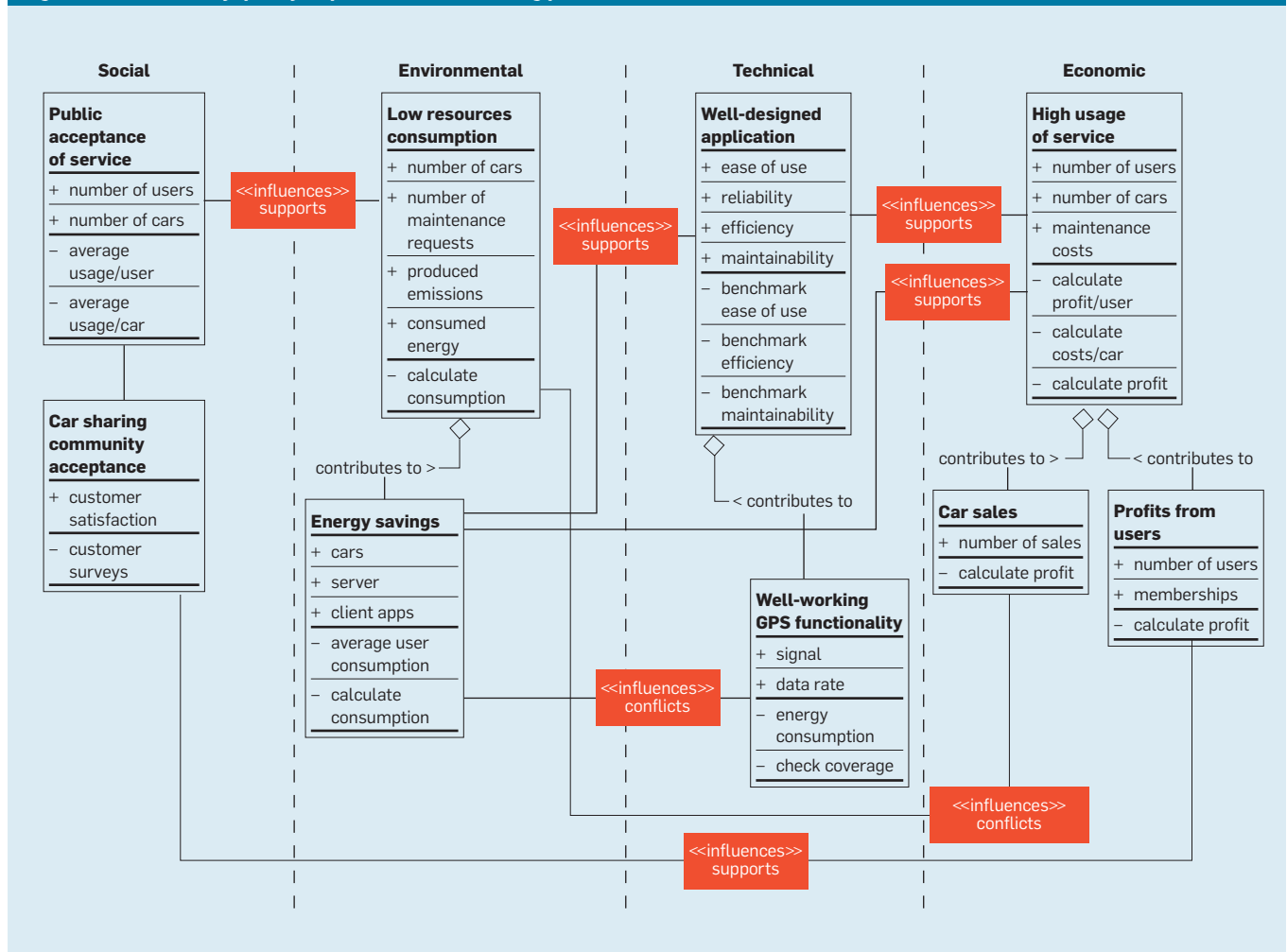
tire production system through smart algorithms and energy-efficient technologies controlled by software. Consequently, the survival of paper-mill companies in Sweden (and worldwide) depends on all four sustainability dimensions, driven primarily by customers and competitors but also by society, including towns, cities, and municipalities, as well as the entire country.

Figure 2 includes example sustainability quality requirements, sorted by sustainability dimensions and the relations among them. We distinguish between vertical (within a dimension) and horizontal (between dimensions) relations. The social dimension refers to the changes in the infrastructure in the companies and in society needed to support requirements for employee skills. A company would need highly educated people, putting demand on their supply from society. The company would need

to make a short- and long-term plan for requalification of employees, and the local society (typically a municipality or county) would need to take responsibility for retraining people. Increased education level would improve environmental sustainability awareness. Such awareness is an example of a horizontal relation. An example of a vertical relation in the environmental dimension involves the following operating environment. A company might deploy new technologies that leads to less water pollution and greater effectiveness of the process that leads to increased environment sustainability (in terms of cleaner water, less energy, reduced forest resources, and forest regeneration). However, such results would require a wise trade-off between increased production, in terms of scalability, performance, and configurability, and economic and environmental requirements; for example, increased

^a According to an internal ABB report, 2007.

Figure 3. Sustainability quality requirements: car-sharing platform.



productivity could undermine environmental demands, and addressing them would require new technologies, as well as changes in the process, including direct and indirect changes (such as selective tree cutting, paper recycling, and planting new trees) requiring changes in the technology of the control system.

The horizontal relations also reflect a balancing of stakeholder interests; trade-offs are typically required between economic and social sustainability requirements or between economic and environmental sustainability requirements. In contrast, technical requirements provide the solutions that improve economic and environmental dimensions.

This case example illustrates how the sustainability analysis framework can be applied in development processes of large, long-lived systems that require public investment and feature significant profit margins. Economic and technical sustainability are customer-driven. The environmental and social sustainability requirements do not come from the customers of the paper mill but from the surrounding community and society at large, including region and state. Due to the large public investment, society can impose requirements. Since environmental and social sustainability requirements do not come from customers, they tend to be overlooked by managers and engineers. Integrating our four-dimensional sustainability analysis framework into the engineering processes of such long-lived industrial systems provides valuable support to managers and engineers trying to satisfy not only economic and technical but also environmental and social sustainability requirements.

Car-sharing platform. In a 2013 study, we analyzed the sustainability impact of DriveNow, a München-based car-sharing platform²⁷ created to serve users who do not otherwise have access to a car for short-distance inner-city trips (see Figure 3). The primary quality requirement is significant use of the platform in the economic sustainability dimension. It is supported by a well-designed application that in turn supports (in the social sustainability dimension)

strong public acceptance of the application. The focus was on the different types of influences affecting framework relations. As with any kind of requirement or goal, sustainability can be linked through various types of influence relationships, as in goals.²⁰ We focus here on support and conflict. In the following paragraphs, we discuss one requirement and its interrelations, illustrating outcomes due to direct and indirect effects on quality requirements. Environmental sustainability, in terms of energy savings, is affected in at least three ways:

GPS. For a well-designed application, reliable GPS functionality is needed, and adding it will, in turn, negatively affect energy savings in the application;

Energy. DriveNow aims to get people to share cars, leading to reduced car production, hence energy savings in production; and

Marketing. DriveNow generates revenue not only through the platform itself but also through the marketing value created by driving new cars around the city; they will be seen by potential customers who may be motivated to buy them, leading in turn to more emissions and less energy savings due to increased car production.

The result is a well-known phenomenon known as first-, second-, and third-order effects.¹³ While use of the app leads to more energy consumption due to GPS use, or a first-order effect (the direct effect of a software system), it also facilitates sharing more cars and thus reduces total energy use, or a second-order effect, the indirect effects triggered by use of a software system in its operational context. On a larger scale, the effect might turn around yet again and lead to a completely different result, or a third-order effect, systemic effects triggered by long-term, widespread use.

The original development of DriveNow did not consider all four dimensions or all these effects. The primary dimension was economic, and the secondary dimension was technical. Both social and environmental were not considered, yielding several consequences:

Social. When the service was available for only a few months and ana-

lyzed by the project team, it turned out a user community was developing in which individual users had established an understanding of themselves as part of a larger community supporting shared mobility services. Had the company's founders and developers considered the social dimension in advance, the system's user interface could have been developed to make it easier to form carpools among users;

Environmental. DriveNow uses mostly environmentally friendly hybrid and electric cars, providing a good basis for environmental sustainability. However, as the company's founders and developers did not consider the environmental aspect of the service during their initial business case analysis, no green IT options were explored for the server side. Likewise, they did not do a comparative (simulation) study of how the long-term widespread use of the service would affect München traffic and parking. Consequently, the environmental sustainability of the system still needs improvement; and

Interrelation of dimensions. One example of often-underestimated relations among the dimensions our framework helps analyze is the use of electric cars, which must be driven in the right way to ensure they produce less pollution (environmental dimension). There is thus a need to offer training (social dimension) for that type of driving, including or leading to further investment (economic dimension).

While simplified, this case illustrates the importance of understanding the interdependencies among qualities by business managers and software developers alike. Our framework is useful for understanding and addressing them, avoiding dangerous pitfalls like negative business consequences and environmental effects.

Observations

These case studies illustrate how our approach to sustainability analysis links the four sustainability dimensions that are seemingly unrelated to software qualities. Determining and analyzing the relations among the qualities, as outlined in Figure 2 and Figure 3, give decision makers a blueprint for analyzing sustain-

ability qualities and gaining insight into sustainability stewardship. By addressing all four dimensions, the framework enables software practitioners to make trade-offs across different dimensions; for example, in the case of the paper-mill control system, a manager using the framework can easily identify not only technical and environmental but also social and economic trade-offs. The framework also helps capture the influence of various stakeholders on the various qualities regarding the four dimensions. Both studies show sustainability quality relations potentially carry positive or negative influences. Moreover, they reveal that when evaluating a system's sustainability quality, all aspects of the system's performance should be taken into consideration; for example, in the case of DriveNow, environmental and social dimensions were originally not included, hindering potential positive effects on the environment. The framework allows management to draw a more comprehensive picture of the relevant quality aspects and help make more-informed decisions.

Figure 2 and Figure 3 are snapshots at the time of the case studies and do not characterize the systems' overall life cycles. The case studies, four dimensions, and relations have their own life cycles. In particular, the relations and their quantification will likely change over time; the initial deployment of infrastructure for a PCS requires a substantial energy investment up front, but situation-aware systems accrue significant benefits over time. While first- and second-order effects could indicate one trajectory in the assessment of sustainability, the effects on global goals can change or even reverse the trend. Moreover, the effect of software systems on the environment could differ dramatically depending on the framework conditions. Any concerns related to sustainability requirements must be prioritized and traded off against business requirements and financial constraints.

The notion of sustainability entails a long chain of (possibly circular) consequences across all the dimensions. When identifying the concerns pertaining to a software system, man-



Due to the large public investment in such an industry [paper production], society can impose requirements.



agement must define the sustainability concerns directly influencing the system, the boundaries outside the scope (but that could be useful for decision making), and the boundaries too remote to be considered. The ISO/IEC 42030 working draft models the environment in which a system is situated. In our understanding of the draft, part of such an environment is within the system's scope, while part is outside it. However, sustainability requirements and concerns likely increase system scope.

There are also limitations as to what the sustainability-analysis framework can provide. The influences among the sustainability quality requirements must be determined by developers and/or stakeholders, as the framework can provide only the means for linking them but not the analysis itself. Constraints and parameters must be chosen by the developers, as it is not possible to list them in a way that is generic enough to be applicable in all circumstances and at the same time specific enough to be useful. The best guidance we can provide with this framework is through examples showing how to apply it and its potential benefits. Part of our own future work is to extend this guidance with further examples.

Conclusion

This article has presented a framework for trading off sustainability quality requirements from the various dimensions of sustainability. It is based on the *Third Working Draft of ISO/IEC 42030 Systems and Software Engineering Architecture Evaluation*¹⁴ and a first attempt at understanding the multidimensional effect of software on its environment. It can assist software practitioners in making trade-offs, not only among technical and economic aspects of business sustainability but also in relation to society and the environment. We focus on classifying sustainability quality requirements as the first step toward sound decision making, trade-off analyses, and quality evaluation. Applying the framework enables software developers to specifically consider the neglected environmental and social dimensions in relation to the technical and economic dimen-

sions. Using the framework, practitioners are better able to determine their sustainability goals and see the potential outcomes of the criteria.

We hope to help provide new research directions and a foundation for discussing the integration of the various ISO quality models. Our own future research will focus on how the framework's sustainability quality requirements can be systematically deduced from a goal model while considering the effects of software on its environment. These requirements include how to refine such information in the form of constraints on design and implementation. Moreover, the resulting models could be useful for cost estimation, specifically in terms of how software design decisions affect architecture and infrastructure. Another open challenge we hope to address is "scoping," or distinguishing sustainability concerns outside the software system but directly influencing it, so the information about such concerns could help take optimal decisions. Finally, as there are no standardized metrics for software sustainability, applying the framework can help establish sound metrics that would serve as a basis for building satisfactory tool support.

Acknowledgments

This work was partially sponsored by the European Fund for Regional Development under project RAAK MKB Greening the Cloud, the Deutsche Forschungsgemeinschaft under project EnviroSiSE (grant PE2044/1-1) and the Swedish Foundation for Strategic Research via project RALF3. Thanks, too, to the participants of the GREENS Workshop at the 35th International Conference on Software Engineering in San Francisco, CA, in 2013 who contributed thoughts and ideas, especially Henning Femmer and Hausi Muller. 

References

1. Afgan, N.H. Sustainability paradigm: Intelligent energy system. *Sustainability* 2, 12 (Dec. 2010), 3812–3830.
2. Akinli Kocak, S., Calienes, G.G., Işıkalar Alptekin, G., and Başar Bener, A. Requirements prioritization framework for developing green and sustainable software using ANP-based decision making. In *Proceedings of the EnviroInformatics Conference* (Hamburg, Germany, Sept. 2–4, 2013), 327–335.
3. Akinli Kocak, S., Işıkalar Alptekin, G., and Başar Bener, A. Evaluation of software product quality attributes and environmental attributes using ANP decision framework. In *Proceedings of the Third International Workshop on Requirement Engineering for Sustainability* (Karlskrona, Sweden, Aug. 26, 2014), 37–44.
4. Brundtland, G. et al. *Our Common Future* (Brundtland Report). United Nations World Commission on Environment and Development, 1987; <http://www.un-documents.net/our-common-future.pdf>
5. Calero, C., Bertoa, M., and Angeles Moraga, M. Sustainability and quality: Icing on the cake. In *Proceedings of the 2013 Workshop on Requirements Engineering for Sustainable Systems* (Rio de Janeiro, Brazil, July 15, 2013), 50–59.
6. Chung, L., Nixon, B.A., Yu, E., and Mylopoulos, J. *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishers, 1992.
7. Corral, L., Georgiev, A.B., Sillitti, A., and Succì, G. A method for characterizing energy consumption in Android smartphones. In *Proceedings of the Second International Workshop on Green and Sustainable Software* (San Francisco, CA, May 20). IEEE, Piscataway, NJ, 2013, 38–45.
8. Crnkovic, I. Are ultra-large systems systems of systems? In *Proceedings of the Second International Workshop on Ultra-Large-Scale Software-Intensive Systems* (Leipzig, Germany, May 10–11). ACM Press, New York, 2008, 57–60.
9. Global e-Sustainability Initiative. *GeSI SMARTer 2020: The Role of ICT in Driving a Sustainable Future*. Global e-Sustainability Initiative, Brussels, Belgium, 2012; <http://gesi.org/portfolio/report/72>
10. Gu, Q. and Lago, P. *An Open Online Library of Green ICT Practices*; www.greenpractice.few.vu.nl
11. Hao, S., Li, D., Halfond, W. G. J., and Govindan, R. Estimating Android applications CPU energy usage via bytecode profiling. In *Proceedings of the First International Workshop on Green and Sustainable Software* (Zürich, Switzerland, June 3). IEEE Press, Piscataway, NJ, 2012, 1–7.
12. Hilty, L.M. and Ruddy, T.F. Sustainable development and ICT interpreted in a natural science context. *Information, Communication & Society* 13, 1 (Feb. 2010) 7–22.
13. Hilty, L.M., Arnalk, P., Erdmann, L., Goodman, J., Lehmann, M., and Wäger, P.A. The relevance of information and communication technologies for environmental sustainability: A prospective simulation study. *Environmental Modelling & Software* 21, 11 (Nov. 2006) 1618–1629.
14. International Organization for Standardization and International Electrotechnical Commission. *42030, Systems and Software Engineering, Architecture Evaluation. Technical Report WD3*. ISO/IEC, New York, 2013.
15. Johann, T., Dick, M., Naumann, S., and Kern, E. How to measure energy efficiency of software: Metrics and measurement results. In *Proceedings of the First International Workshop on Green and Sustainable Software* (Zürich, Switzerland, June 3). IEEE Press, Piscataway, NJ, 2012, 51–54.
16. Kalaitzoglou, G., Bruntink, M., and Visser, J. A practical model for evaluating the energy efficiency of software applications. In *Proceedings of the International Conference of ICT for Sustainability* (Stockholm, Sweden, Aug. 24–27). Atlantis Press, Amsterdam, the Netherlands, 2014.
17. Kern, E., Dick, M., Naumann, S., Guldner, A., and Johann, T. Green software and green software engineering: Definitions, measurements, and quality aspects. In *Proceedings of the First International Conference of ICT for Sustainability* (Zürich, Switzerland, Feb. 14–16, 2013), 87–94.
18. Lago, P., Gu, Q., and Bozzelli, P. A Systematic Literature Review of Green Software Metrics. *Technical Report*. University of Tampere, Finland, 2013; http://www.sis.uta.fi/~pt/TIEA5_Thesis_Course/Session_10_2013_02_18/SLR-GreenMetrics.pdf
19. Lago, P., Jansen, T., and Jansen, M. The service greenery: Integrating sustainability in service-oriented software. In *Proceedings of the Second International IEEE Workshop on Software Research and Climate Change* (Cape Town, South Africa, May 3, 2010).
20. Lamsweerde, A.V. *Requirements Engineering*. John Wiley & Sons, New York, 2007.
21. Li, D., Sahin, C., Clause, J., and Halfond, W. G. J. Energy-directed test suite optimization. In *Proceedings of the Second International Workshop on Green and Sustainable Software* (San Francisco, CA, May 20). IEEE Press, Piscataway, NJ, 2013, 62–69.
22. Mahaux, M., Heymans, P., and Saval, G. Discovering sustainability requirements: An experience report. In *Proceedings of the International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer, Heidelberg, Germany, 2011, 19–33.
23. Moigemba, D. Sustainable computing. *Poiesis & Praxis* 4, 3 (Dec. 2006) 163–184.
24. Naumann, S., Dick, M., Kern, E., and Johann, T. The GREENSOFT model: A reference model for green and sustainable software and its engineering. *Sustainable Computing: Informatics and System* 1, 4 (Dec. 2011) 294–304.
25. Penzenstadler, B., Bauer, V., Calero, C., and Franch, X. Sustainability in software engineering: A systematic literature review. In *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering* (Ciudad Real, Spain, May 14–15). IET, Wales, U.K., 2012, 32–41.
26. Penzenstadler, B., Tomlinson, B., and Richardson, D. RE4ES: Support environmental sustainability by requirements engineering. In *Proceedings of the First International Workshop on Requirements Engineering for Sustainable Systems* (Essen, Germany, Mar. 19, 2012), 34–39.
27. Penzenstadler, B. and Femmer, H. A generic model for sustainability with process- and product-specific instances. In *Proceedings of the 2013 Workshop on Green in/by Software Engineering* (Fukuoka, Japan, Mar. 26). ACM Press, New York, 2013, 3–8.
28. Penzenstadler, B. and Femmer, H., and Richardson, D. Who is the advocate? Stakeholders for sustainability. In *Proceedings of the Second International Workshop on Green and Sustainable Software at the 35th International Conference on Software Engineering* (San Francisco, CA, May 20). IEEE Press, Piscataway, NJ, 2013, 70–77.
29. Penzenstadler, B., Raturi, A., Richardson, D., Calero, C., Femmer, H., and Franch, X. Systematic mapping study on software engineering for sustainability (SE4S). In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering* (London, U.K., May 13–14). ACM Press, New York, 2014, article 14.
30. Penzenstadler, B., Raturi, A., Richardson, D., and Tomlinson, B. Safety, security, now sustainability: The non-functional requirement for the 21st century. *IEEE Software* 31, 3 (May–June 2014), 40–47.
31. Procaccianti, G., Lago, P. and Bevin, S. A systematic literature review on energy efficiency in cloud software architectures. *Sustainable Computing: Informatics and Systems* 4 (Nov. 2014).
32. Razavian, M., Procaccianti, G., and Tamburri, D.A. Four-dimensional sustainable e-services. In *Proceedings of the International Conference on Informatics for Environmental Protection* (Oldenburg, Germany, Sept. 10–12, 2014), 221–228.
33. Razavian, M., Lago, P., and Gordijn, J. Why is aligning economic and IT services so difficult? Chapter in *Exploring Services Science*. Springer, 2014, 92–107.
34. Sustainability. *Sustainability: Can our society endure?*; <http://www.sustainability.com/sustainability>
35. Tate, K. *Sustainable Software Development: An Agile Perspective*. Addison-Wesley Professional, Boston, MA, 2005.
36. Tratz-Ryan, B. *Sustainability Innovation Key Initiative Overview*. Gartner RAS Research Note G00251246, June 14, 2013; <https://www.gartner.com/doc/2516916/sustainability-innovation-key-initiative-overview>

Patricia Lago (p.lago@vu.nl) is a professor of software engineering and leader of the Software and Services Research Group at VU University Amsterdam, the Netherlands.

Sedef Akinli Kocak (Sedef.akinlikocak@ryerson.ca) is a researcher at Environmental Applied Science and Management Data Science Lab at Ryerson University, Toronto, Canada.

Ivica Crnkovic (crnkovic@chalmers.se) is a professor of software engineering and a director of the ICT Area of Advance at Chalmers University of Technology, Gothenburg, Sweden.

Birgit Penzenstadler (birgit.penzenstadler@csulb.edu) is a professor of software engineering and leader of the Software Engineering for Sustainability Lab at the California State University, Long Beach.

Call for Nominations

The ACM Doctoral Dissertation Competition

Rules of the Competition

ACM established the Doctoral Dissertation Award program to recognize and encourage superior research and writing by doctoral candidates in computer science and engineering. These awards are presented annually at the ACM Awards Banquet.

Submissions

Nominations are limited to one per university or college, from any country, unless more than 10 Ph.D.'s are granted in one year, in which case two may be nominated.

Eligibility

Please see our website for exact eligibility rules. Only English language versions will be accepted. Please send a copy of the thesis in PDF format to emily.eng@hq.acm.org.

Sponsorship

Each nomination shall be forwarded by the thesis advisor and must include the endorsement of the department head. A one-page summary of the significance of the dissertation written by the advisor must accompany the transmittal.

Deadline

Submissions must be received by **October 31, 2015** to qualify for consideration.

Publication Rights

Each nomination must be accompanied by an assignment to ACM by the author of exclusive publication rights. (Copyright reverts to author if not selected for publication.)

Publication

Winning dissertations will be published by ACM in the ACM Books Program and appear in the ACM Digital Library. Honorable mention dissertations will appear in the ACM Digital Library.

Selection Procedure

Dissertations will be reviewed for technical depth and significance of the research contribution, potential impact on theory and practice, and quality of presentation. A committee of individuals serving staggered five-year terms performs an initial screening to generate a short list, followed by an in-depth evaluation to determine the winning dissertation.

The selection committee will select the winning dissertation in early 2016.

Award

The Doctoral Dissertation Award is accompanied by a prize of \$20,000 and the Honorable Mention Award is accompanied by a prize of \$10,000. Financial sponsorship of the award is provided by Google.

For Submission Procedure

http://awards.acm.org/doctoral_dissertation/



Association for
Computing Machinery

The challenge of missing heritability offers great contribution options for computer scientists.

BY ELEAZAR ESKIN

Discovering Genes Involved in Disease and the Mystery of Missing Heritability

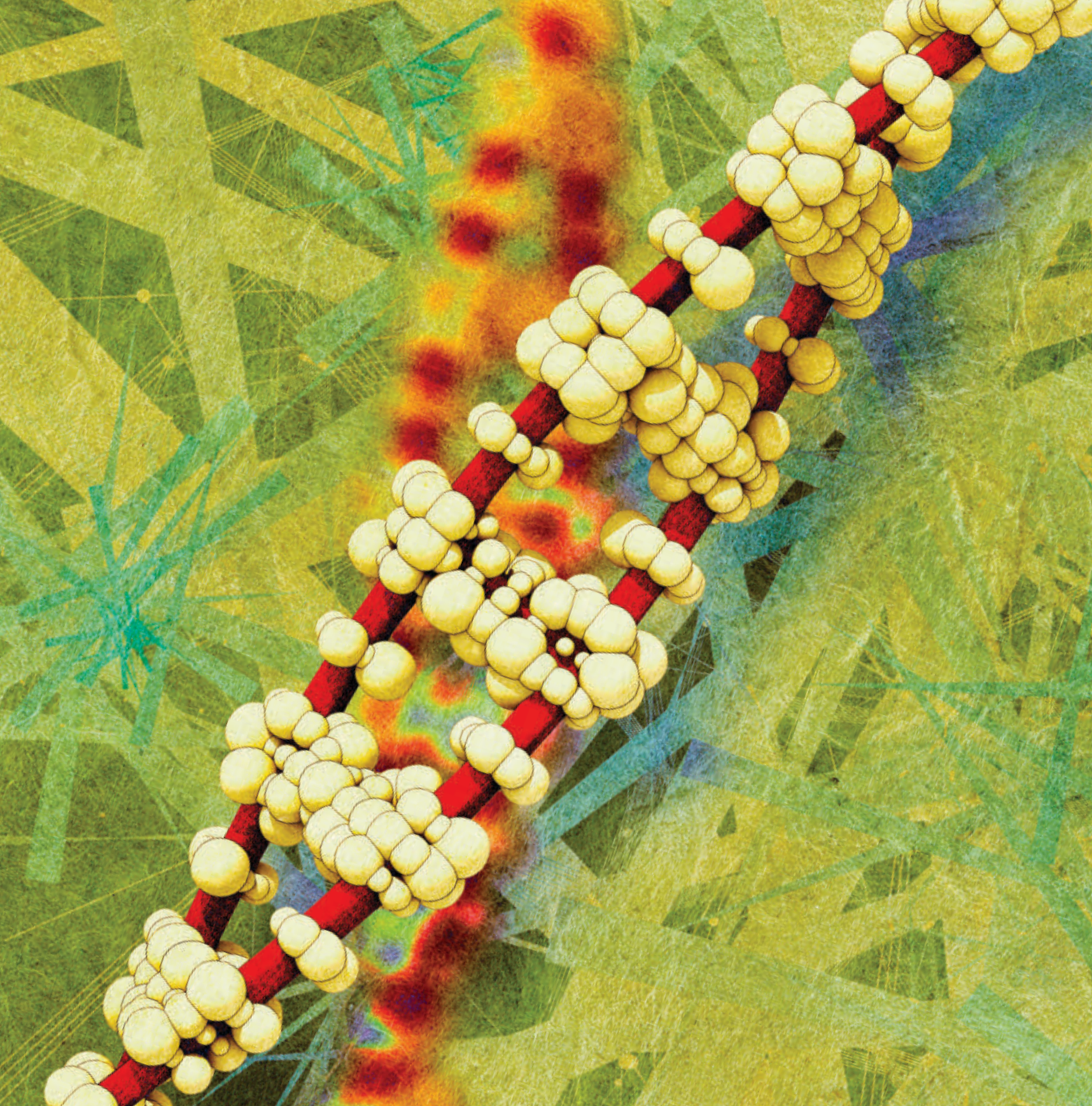
WE LIVE IN a remarkable time for the study of human genetics. Nearly 150 years ago, Gregor Mendel published his laws of inheritance, which lay the foundation for understanding how the information that determines traits is passed from one generation to the next. Over 50 years ago, Watson and Crick discovered the structure of DNA, which is the molecule that encodes this genetic information. All humans share the same three billion-length DNA sequence at more than 99% of the

positions. Almost 100 years ago, the first twin studies showed this small fraction of genetic differences in the sequence accounts for a substantial fraction of the diversity of human traits. These studies estimate the contribution of the genetic sequence to a trait by comparing the relative correlation of traits between pairs of maternal twins (which inherit identical DNA sequences from their parents) and pairs of fraternal twins (which inherit a different mix of the genetic sequence from each parent).^{5,29} This contribution is referred to as the “heritability” of a trait. For example, twin studies have shown that genetic variation accounts for 80% of the variability of height in the population.^{5,15,26} The amount of information about a trait encoded in the genetic sequence suggests it is possible to predict the trait directly from the genetic sequence and this is a central goal of human genetics.

Only in the past decade has technology developed to be able to cost effectively obtain DNA sequence information from individuals and a large number of the actual genetic differences have been identified and implicated in having an effect on traits. On the average, individuals who carry such a genetic difference, often referred to as a genetic variant, will have a different value for a trait compared to individuals who do not carry the variant. For example, a recently published paper reporting on a large study to identify the genetic differences that affect height

» key insights

- Over the past several years, thousands of genetic variants that have been implicated in dozens of common diseases have been discovered.
- Despite this progress, only a fraction of the variants involved in disease have been discovered—a phenomenon referred to as “missing heritability.”
- Many challenges related to understanding the mystery of missing heritability and discovering the variants involved in human disease require analysis of large datasets that present opportunities for computer scientists.



reported hundreds of variants in the DNA sequence that either increase or decrease an individual's height if the individual carries the variant.^{2,23} Knowing these variants and their effects allows us to take the first steps in predicting traits only using genetic information. For example, if an individual carried many variants that increased height, we would predict the individual's height is higher than the population average. While predicting

an easily measured trait such as height from genetic information seems like an academic exercise, the same ideas can be used to predict disease-related traits such as risk of heart attack or response to a certain drug. These predictions can help guide selecting the best treatment options. Over 1,000 genetic variants have been implicated in hundreds of traits including many human disease-related traits of great medical importance.^{16,31}

A majority of these discoveries were made using a type of genetic study called a genome-wide association study (GWAS). In a GWAS, data from a large number of individuals is collected, including both a measurement of the disease-related trait as well as information on genetic variants from the individual. GWAS estimate the correlation between the collected disease trait and the collected genetic variants to identify genetic variants

that are “associated” with disease.²⁷ These associated variants are genetic variations that may have an effect on the disease risk of an individual.

While GWAS have been extremely successful in identifying variants involved in disease, the results of GWASs have also raised a host of questions. Even though hundreds of variants have been implicated to be involved in some traits, their total contribution only explains a small fraction of the total genetic contribution that is known from twin studies. For example, the combined contributions of the 50 genes discovered to have an effect on height using GWASs through 2009 with over tens of thousands individuals only account for ~5% of the phenotypic variation, which is a far cry from the 80% heritability previously estimated from twin studies.³² The gap between the known heritability and the total genetic contribution from all variants implicated in genome studies is referred to as “missing heritability.”¹⁷

After the first wave of GWAS results reported in 2007 through 2009, it became very clear the discovered variants were not going to explain a significant portion of the expected heritability. This observation was widely referred to as the “mystery of missing heritability.” A large number of possible explanations for the “missing heritability” were presented, including interactions between variants, interactions between variants and the environments, and rare variants.¹⁷ Missing heritability has very important implications for human health. A key challenge in personalized medicine is how to use an individual’s genomes to predict disease risk. The genetic variants discovered from GWASs up to this point only utilize a fraction of the predictive information we know is present in the genome. In 2009 and 2010, a pair of papers shook the field by suggesting the missing heritability was not really “missing,” but actually accounted for in the common variants,^{21,32} which had very small effects. This was consistent with the results of the larger GWAS studies performed in 2011 and 2012, which analyzed tens of thousands of individuals and reported even more variants involved in disease, many of them with very small

effects as postulated. The results of these later studies provide a clearer picture of the genetic architecture of disease and motivate great opportunities for predicting disease risk for an individual using their genetic information. This article traces the history of the GWAS era from the first studies, through the mystery of missing heritability and to the current understanding of what GWAS has discovered.

What is exciting about the area of genetics is that many of these questions and challenges are “quantitative” in nature. Quantitative genetics is a field with a long and rich history dating back to the works of R.A. Fisher, Sewall Wright, and J.B.S. Haldane, which are deeply intertwined with the development of modern statistics. With the availability of large-scale genetic datasets^{1,28} including virtually all data from published GWASes, the critical challenges involve many computationally intensive data analysis problems. There are great opportunities for contributions to these important challenges from computer scientists.

The Relation between Genotypes and Phenotypes

The genomes of any two humans are approximately 99.9% identical and the small amount of differences in the remaining genomic sequence accounts for the full range of phenotypic diversity we observe in the human population. A genetic variant is a position in the human genome where individuals in the population have different genetic content. The most common type of genetic variation is referred to as a single nucleotide polymorphism (SNP). For example, the SNP rs9939609 refers to the position 53820527 on chromosome 16, which is in the FTO gene and was implicated in Type 2 diabetes in one of the first genome-wide studies performed.³⁰ For this SNP, 45% of the chromosomes in the European population have an “A” in that position while 55% have the “T” in that position.²⁸ The occurring genomic content (“A” or “T”) is referred to as the “allele” of the variant and the frequency of the rarer allele of the variant (0.45) is referred to as the minor allele frequency (MAF). The less common allele (in this case “A”) is referred

to as the minor allele and the more common allele (in this case “T”) is referred to as the major allele. The specific allele present in an individual is referred to as the genotype. Because mutations occur rarely in human history, for the vast majority of SNPs, only two alleles are present in the human population. Since humans are diploid—each individual has two copies of each chromosome—the possible genotypes are “TT,” “AT,” and “AA” typically encoded “0,” “1,” and “2” corresponding to the number of minor alleles the individual carries.

There are many kinds of genetic variation that are present in addition to SNPs such as single position insertion and deletions, referred to as indels, or even larger variants, referred to as structural variants, encompassing such phenomenon as duplications or deletions of stretches of the genome or even inversions or other rearrangements of the genome. Virtually all GWASes collect SNP information because SNPs are by far the most common form of genetic variation in the genome and are present in virtually every region in the genome as well as amenable to experimental techniques that allow for large-scale collection of SNP information.^{7,18} While other types of genetic variation may be important in disease, since SNPs are so common in the genome, virtually every other type of genetic variant occurs near a SNP that is highly correlated with that variant. Thus genetic studies collecting SNPs can capture the genetic effects of both the SNPs they collect as well as the other genetic variants that are correlated with these SNPs.

Genetic variation can be approximately viewed as falling into one of two categories: common and rare variation. The minor allele frequency threshold separating common and rare variation is obviously subjective and the threshold is usually defined in the range of 1%–5% depending on the context. Variants that are more common tend to be more strongly “correlated” to other variants in the region. The genetics community, for historical reasons, refers to this correlation by “linkage disequilibrium.” Two variants are “correlated” if whether or not an individual carries the minor allele at one variant provides

information on carrying the minor allele at another variant. This correlation structure between neighboring variants is a result of human population history and the biological processes that pass variation from one generation to the next. The study of these processes and how they shape genetic variation is the rich field of population genetics.⁸

The field of genetics assumes a standard mathematical model for the relationship between genetic variation and traits or phenotypes. This model is called the polygenic model. Despite its simplicity, the model is a reasonable approximation of how genetic variation affects traits and provides a rich starting point for understanding genetic studies. Here, we describe a variant of the classic polygenic model.

We assume our genetic study collects N individuals and the phenotype of individual j is denoted y_j . We assume a genetic study collects M variants and for simplicity, we assume all of the variants are independent of each other (not correlated). We denote the frequency of variant i in the population as p_i . We denote the genotype of the i th variant in the j th individual as $g_{ij} \in \{0, 1, 2\}$, which encodes the number of minor alleles for that variant present in the individual. In order to simplify the formulas later in this article, without loss of generality, we normalize the genotype values such that

$$x_{ij} \in \left\{ \frac{-2p_i}{\sqrt{2p_i(1-p_i)}}, \frac{1-2p_i}{\sqrt{2p_i(1-p_i)}}, \frac{2-2p_i}{\sqrt{2p_i(1-p_i)}} \right\}$$

since the mean and variance of the column vector of genotypes (g_i) is $2p_i$ and $2p_i(1-p_i)$, respectively. Because of the normalization, the mean and variance of the vector of genotypes at a specific variant i denoted X_i is 0 and 1, respectively.

The phenotype can then be modeled using

$$y_j = \mu + \sum_{i=1}^M \beta_i x_{ij} + e_j \quad (1)$$

where the effect of each variant on the phenotype is β_i , the model mean is μ and e_j is the contribution of the environment on the phenotype is assumed to be normally distributed with variance σ_e^2 , denoted $e_j \sim \mathcal{N}(0, \sigma_e^2)$. We note that inherent to this model is

Missing heritability has very important implications for human health.

the “additive” assumption in that the variants all contribute linearly to the phenotype value. More sophisticated models, which include nonadditive effects or gene-by-gene interactions, are an active area of research.

If we denote the vector of phenotypes y and vector of effect sizes β , the matrix of normalized genotypes X and the vector of environmental contributions e , then the model for the study population can be denoted

$$y = \mu \mathbf{1} + \sum_i \beta_i X_i + e \quad (2)$$

where $\mathbf{1}$ is a column vector of 1s, and e is a random vector drawn from the multivariate normal distribution with mean 0 and covariance matrix $\sigma_e^2 \mathbf{I}$, denoted as $e \sim \mathcal{N}(0, \sigma_e^2 \mathbf{I})$.

Genome-Wide Association Studies

Genome-wide association studies (GWAS) collect disease trait information, referred to as phenotypes, and genetic information, referred to as genotypes, from a set of individuals. The phenotype is either a binary indicator of disease status or a quantitative measure of a disease-related trait such as an individual's cholesterol level. Studies that collect binary trait information are referred to as case/control studies and typically collect an equal number of individuals with and without the disease. Studies that collect quantitative measures are often performed on a representative sample of a population, referred to as a population cohort, and collect individuals using a criteria designed to be representative of a larger population (for example, all individuals who were born in a specific location in a specific year²⁵).

GWASes focus on discovering the common variation involved in disease traits. Because of the correlation structure of the genome, GWASes only collect a subset of the common variation typically in the range of 500,000 variants. Studies have shown that collecting only this fraction of the common variants “captures” the full set of common variants in the genome. For the vast majority of common variants in the genome, at least 1 of the 500,000 variants that is collected is correlated with the variant. GWASes typically collect genotype information on these

variants in thousands of individuals along with phenotypic information.

The general analysis strategy of GWAS is motivated by the assumptions of the polygenic model (Equation 1). In a GWAS, genotypes and phenotypes are collected from a set of individuals with the goal of discovering the associated variants. Intuitively, a GWAS identifies a variant involved in disease by splitting the set of individuals based on their genotype (“0,” “1,” or “2”) and computing the mean of the disease-related trait in each group. If the means are significantly different, then this variant is declared associated and maybe involved in the disease. More formally, the analysis of GWAS data in the context of the model in Equation (1) corresponds to estimating the vector β from the data and we refer to the estimated vector as $\hat{\beta}$ following the convention that estimates of unknown parameters from data are denoted with the “hat” over the parameter. Since the number of individuals is at least an order of magnitude smaller than the number of variants, it is impossible to simultaneously estimate all of the components of β . Instead, in a typical GWAS, the effect size for each variant is estimated one at a time and a statistical test is performed to determine whether or not the variant has a significant effect on the phenotype. This is done by estimating the maximum likelihood parameters of the following equation

$$y = \mu \mathbf{1} + \beta_k X_k + \mathbf{e} \quad (3)$$

which results in estimates of $\hat{\mu}$ and $\hat{\beta}_k$ and performs a statistical test to see if the estimated value of $\hat{\beta}_k$ is non-zero. (See Appendix 1, available with this article in the ACM Digital Library, for more details on association statistics.) The results of an association study is then the set of significantly associated variants, which we denote using the set A , and their corresponding effect size estimates $\hat{\beta}_i$.

The results of GWASes can be directly utilized for personalized medicine. In personalized medicine, one of the challenges is to identify individuals that have high genetic risk for a particular disease. In our model from

In the genetics community, how much genetics influences a trait is quantified using “heritability,” which is the proportion of disease phenotypic variance explained by the genetics.

Equation (1), each individual’s phenotype can be decomposed into a genetic mean ($\mu + \sum_{i=1}^M \beta_i x_{ij}$) and an environmental component (e_j). The genetic mean, which is unique to each individual and a function of the effect sizes and the individual’s genotypes, can be thought of as a measure of the individual’s genetic risk. Thus, inferring this genetic mean is closely related to identifying individuals at risk for a disease and since the environmental contribution has mean 0, predicting the genetic mean and the phenotype are closely related problems.

Knowing nothing about an individual’s genotypes or the effect sizes, the best prediction for an individual’s phenotype would be the prediction of the phenotypic mean of $\hat{\mu}$. The more information we have on an individual’s genotypes and the effects sizes, the more closely our phenotype prediction is to the true phenotype. Using the results of a GWAS and the genotypes of a new individual x^* , we can use the discovered associated loci to make a phenotype prediction, y^* , for the individual using $y^* = \hat{\mu} + \sum_{i \in A} \hat{\beta}_i x_i^*$. As we discuss here, while the prediction of a trait from GWAS is more informative than just using the mean, unfortunately, the predictions are not accurate enough to be clinically useful.

What GWAS Has Discovered and the Mystery of Missing Heritability

In the genetics community, how much genetics influences a trait is quantified using “heritability,” which is the proportion of disease phenotypic variance explained by the genetics. The heritability of a trait can be measured using other approaches taking advantage of related individuals. One approach for measuring heritability is taking advantage of twin studies. Twin studies measure the same trait in many pairs of twins. Some of these pairs of twins are monozygotic (MZ) twins, often referred to as maternal twins and some of the pairs are dizygotic (DZ) twins, often referred to as fraternal twins. The difference between MZ twins and DZ twins is that MZ twins have virtually identical genomes, while DZ twins only share about 50% of their genomes. By computing the relative correlation between trait values of MZ twins

versus DZ twins, heritability of the trait can be estimated.²⁹ Intuitively, if the MZ twins within a pair have very similar trait values while DZ twins within a pair have different trait values, then the trait is very heritable. If the difference in trait values with pairs of MZ twins is approximately the same as the difference between values within pairs of DZ twins, then the trait is not very heritable.

In our model, the total phenotypic variance $\text{Var}(y)$ can be decomposed into a genetic component and environmental component. In our context, heritability refers to the proportion the variance of the genetic component ($\sum_i \beta_i X_i$) contributes to the overall variance. The variance corresponding to the environment is σ_e^2 . Since the genotypes are normalized, the phenotypic variance accounted for by each variant is β_i^2 , thus the total genetic variance is $\sum_{i=1}^M \beta_i^2$. The heritability, which is denoted h^2 for historical reasons, is then

$$h^2 = \frac{\sum_{i=1}^M \beta_i^2}{\sum_{i=1}^M \beta_i^2 + \sigma_e^2} \quad (4)$$

Unfortunately, we do not know the true values of β_i or σ_e^2 . The studies using twins have been shown to closely approximate the heritability as defined in Equation (4).

GWASes have been tremendously successful in discovering variation involved in traits. The initial studies found a few variants in disease. For example, one of the first GWASes was the Wellcome Trust Case Control Consortium study, which used 3,000 healthy individuals and 2,000 individuals from each of seven diseases.³⁰ They found 24 associations. As sample sizes increased, more discoveries were found particularly because many smaller GWASes were combined to enable a meta-analysis of a larger population. The results of all GWASes are catalogued at the National Human Genome Research Institute (<http://www.genome.gov/gwastudies>) and as of November 2013, GWASes have identified 11,996 variants associated with 17 disease categories.¹⁰

While the large number of associations discovered can lead to new insights about the genetic basis of common diseases, the vast majority of

discovered loci have very small effect sizes. Yet it is well known that genetics plays a large role in disease risk. For example, for many diseases, it is known that parental disease history is a strong predictor of disease risk.

Now let us use the results of GWAS to estimate the heritability. We can also estimate the total phenotypic variance by estimating the variance of our phenotypes directly, $\text{Var}(y)$, which is a reasonable approximation for the true phenotypic variance ($\sum_{i=1}^M \beta_i^2 + \sigma_e^2$). Let A be the set of associated variants and for these variants, the estimate $\hat{\beta}_i$ is a reasonable estimate for β_i . We can use them to estimate the heritability explained by GWAS which we denote $\hat{h}_G^2$

$$\hat{h}_G^2 = \frac{\sum_{i \in A} \hat{\beta}_i^2}{\text{Var}(y)} \quad (5)$$

We note the main difference between $\hat{h}_G^2$ and h^2 is there are only $|A|$ terms in the numerator of $\hat{h}_G^2$ while there are M terms in h^2 . For this reason, $\hat{h}_G^2 < h^2$. Intuitively, the difference between $\hat{h}_G^2$ and h^2 is the gap between the contribution of the variants that have been discovered by GWAS and the contribution of all variants to the genetic effect.

A landmark study in 2009 compared the heritability estimates from twin studies to the heritability explained by GWAS results.¹⁷ In this study, they showed that the 18 variants implicated by GWAS in Type 2 Diabetes only explained 6% of the known heritability. Similarly, the 40 variants implicated to be involved in height at that time only explained 5% of the heritability. The large gap between the heritability is referred to as the “missing heritability” and a large amount of effort has gone into finding this missing heritability.

Part of the picture of missing heritability can be explained by analyzing the statistical power of GWASes. An analysis of the statistical power shows that even very large GWAS studies often fail to detect trait-affecting variants that have low minor allele frequencies (see Appendix 1, available online, for a discussion and definition of statistical power). Thus, a possible explanation for missing heritability is that a very large number of variants with very small effects are present throughout the genome accounting

for the remaining heritability and simply could not be discovered by GWAS due to power considerations. If this is the case, as study samples increase, more and more of these variants will be discovered and the amount of heritability explained by the GWAS results will slowly approach the total heritability of the trait. Unfortunately, there is a practical limit to how large GWASes can become due to cost considerations. Even putting cost aside, for some diseases, there are simply not enough individuals with the disease on the planet to perform large enough GWASes to discover all of the variants involved with the disease.

Without the ability to perform even larger GWASes, it was not clear if we could identify whether there are enough small effect size variants in the genome corresponding to the missing heritability or the missing heritability was due to some other reasons such as interactions between variants, structural variation, rare variants, or interactions between genetics and environment.

Mixed Models for Population Structure and Missing Heritability

Another insight into missing heritability emerged from what initially seemed like an unrelated development addressing an orthogonal problem in association studies. GWAS statistics (Appendix 1, available online) make the same assumptions as linear regression, which assumes the phenotype of each individual is independently distributed. Unfortunately, this is not always the case. The reason is due to the discrepancy the statistical model that actually generated the data (Equation 2) and the statistical model that is assumed when performing a GWAS (Equation 3). The term that is missing from the testing model, $\sum_{i=k} \beta_i x_{ij}$, is referred to as an unmodeled factor. This unmodeled factor corresponds to the effect of variants in the genome other than the variant being tested in the statistical test.

If the values for the unmodeled factor are independently distributed among individuals, then the factor will increase the amount of variance, but not violate the independently distributed assumption of the statistics. The effect of the unmodeled factor is it

will increase the variance estimate of $\hat{\sigma}_e^2$ in Equation (3) compared to the true environmental variance σ_e^2 in Equation (2). However, if the unmodeled factor is not independently distributed, then this will violate the assumptions of the statistical test in Equation (3).

Unfortunately, in association studies, the effect of the rest of the genome on a trait is not independent when individuals who are related are present in the association studies. Consider a pair of siblings who are present in an association study as well as a pair of unrelated individuals. Since siblings share about half of their genome, for half of the genome, they will have identical genotypes. Many of these variants will have an effect on the phenotype. The values of $\sum_{i \neq k} \beta_i x_{ij}$ will be much closer to each other for siblings compared to a pair of unrelated individuals. This applies for more distant relationships as well. This problem is referred to as “population structure” where differing degrees of relatedness between individuals in the GWAS cause an inflation of the values of the association statistics leading to false positives. Many methods for addressing population structure have been presented over the years including genomic control⁴ that scales the statistics to avoid inflation, principal component based methods,²⁰ and most recently mixed model methods.^{11,12,14,34}

The basis of the mixed model approach to population structure is the insight the proportion of the genome shared corresponds to the expected similarity in the values of the unmodeled factors. In fact, the amount of similarity between the unmodeled factors in association studies will be proportional to the amount of the genome shared between individuals, particularly under some standard assumptions made about the effect sizes of the variants and the assumption that each variant has equal likelihood of being causal. More precisely, the covariance of the unmodeled factors is proportional to the amount of the genome shared. The amount of genome shared is referred to as the “kinship matrix” and since the genotypes are normalized, the kinship is simply $\mathbf{K} = \mathbf{X}\mathbf{X}^T/M$ where $\mathbf{X}$ is the $N \times M$

matrix of the normalized genotypes. We then add a term to the statistical model to capture these unmodeled factors resulting in the statistical model

$$y = \mu \mathbf{1} + \beta_k x_k + \mathbf{u} + \mathbf{e} \quad (6)$$

where x_k is a column vector of normalized genotypes for variant k , $e \sim \mathcal{N}(0, \sigma_e^2 \mathbf{I})$, and $\mathbf{u} \sim \mathcal{N}(0, \sigma_g^2 \mathbf{K})$ represents the contributions of the unmodeled factors. When performing an association, mixed model methods estimate the maximum likelihood for parameters μ , β_k , σ_g^2 , and σ_e^2 using the likelihood $L(N, y, x_k, \mu, \beta_k, \sigma_g^2, \sigma_e^2, \mathbf{K})$

$$(2\pi)^{-N/2} \left| \sigma_g^2 \mathbf{K} + \sigma_e^2 \mathbf{I} \right|^{-1/2} e^{-\frac{1}{2} (y - \mu - \beta_k x_k)^T (\sigma_g^2 \mathbf{K} + \sigma_e^2 \mathbf{I})^{-1} (y - \mu - \beta_k x_k)} \quad (7)$$

and compare this maximum likelihood to the maximum likelihood when β_k is restricted to 0. By comparing these likelihoods, mixed model methods can obtain a significance for the association at variant k correcting for population structure. Mixed models were shown to perform well for a wide variety of population structure scenarios and correct for the structure in studies involving closely related individuals¹³ to studies with more distant relationships.¹¹

A major development related to the mystery of missing heritability was when the connection was made between the mixed model estimates of σ_g^2 and σ_e^2 . In a seminal paper, it was pointed out that these estimates from GWAS data for a population cohort can be used to estimate the heritability.³² We refer to this estimate as h_M^2 where

$$h_M^2 = \frac{\sigma_g^2}{\sigma_g^2 + \sigma_e^2} \quad (8)$$

This method was applied to estimate the heritability of height from the full set of GWAS data and obtained an estimate of 0.45, which is dramatically higher than the estimate from the results of the association studies (h_G^2), which was 0.05. This study suggests the common variants capture a much larger portion of the heritability than just the associated variants, which provides strong support that the main cause of missing heritability is simply many variants with very small effects spread throughout the genome.

Around the same time, another study showed if the criterion for including variants in a prediction model is not as stringent as standard GWAS, but instead, the significance threshold is reduced, the predictions of the model are more accurate.²¹ In this study, not only significant associated variants, but variants that had weaker effects were included in the model and the resulting predictive model showed better performance when it was evaluated using cross-validation. This further suggests many weakly associated variants are contributing the missing heritability. This concept of including more variants in the predictive model is analogous to the trade-off related to prediction accuracy and overfitting when including more features in a machine learning classifier.

While mixed model approaches are a step toward understanding the mystery of missing heritability, there are still many unanswered questions. There is still a significant discrepancy between estimates from related individuals and mixed model estimates. For example, height is estimated to have a heritability of 0.8, while mixed models only explain 0.45. One theory for the remaining heritability argues the remaining portion of the missing heritability can be explained by rare variants having small effects that are poorly correlated with the common variants used to compute kinship matrices.³² Other theories postulate that interactions between variants account for the remaining missing heritability.^{3,35} Additional questions are related to the fact the interpretation of the mixed model estimate of heritability is only correct under the assumption that only the causal variants are used for estimating the kinship matrices.³⁵ Unfortunately, which variants are causal is unknown and various approaches have been proposed to address this issue.⁶

The developments in mixed models provide interesting opportunities for phenotype prediction, which is a problem with a rich history in genetics, particularly in the literature on the best linear unbiased predictor (BLUP).^{9,19,24} Consider the scenario where we have a population of individuals with known

phenotypes y and genotypes X . Given a new individual's genome x^* , we can predict the individual's phenotype y^* using mixed models. In order to make predictions, we first estimate the parameters of the mixed model σ_g^2 and σ_e^2 . We then compute the kinship values between the new individual and the set of individuals with known genotypes and phenotypes. We can then treat the new individual's phenotype as missing and compute the most likely value for this phenotype value given the mixed model likelihood value.

The Future of Phenotype Prediction

Phenotype prediction from genetic information is currently an active area of research. Clearly phenotype prediction using only associated variants ignores the information from the polygenic score obtained from mixed models and only leverages the information from the portion of the heritability that is accounted for in GWASes. However, using only the polygenic score from mixed models ignores variants that are clearly involved in the trait. Several strategies are utilizing both types of information by first utilizing the associated SNPs and then using a polygenic score from the rest of the genome.^{22,33} However, even these combined strategies seem to be missing out on information because variants that are just below the significance threshold have a higher chance of having an effect on the phenotype than other variants, yet all variants are grouped together when estimating the kinship matrix and the polygenic score from variants that are not associated. This problem is closely related to the standard classification problem widely investigated in the machine learning community.

Phenotype and genotype data for massive numbers of individuals is widely available. The actual disease study datasets are available through a National Center for Biotechnology Information database called the database of Genotypes and Phenotypes (dbGaP) available at <http://www.ncbi.nlm.nih.gov/gap>. Virtually all U.S. government-funded GWASes are required to submit their data into the dbGaP database. A similar project,

the European Genome-Phenome Archive (EGA) hosted by the European Bioinformatics Institute (EBI) is another repository of genome wide association study data available at <https://www.ebi.ac.uk/ega/>. For both of these databases, investigators must apply for the data in order to guarantee they comply with certain restrictions on the use of the data due to the inherent privacy and ethical concerns. Hundreds of large datasets are available through both of these resources.

This computational challenge (as well as other computational challenges in human genetics listed in Appendix 2, available online) will have a great impact on human health and provide tremendous opportunities for important contributions from computer scientists. **□**

References

- Abecasis, G.R., Auton, A., Brooks, L.D., DePristo, M.A., et al. An integrated map of genetic variation from 1,092 human genomes. *Nature* 491, 7422 (2012), 56–65.
- Berndt, S.I., Gustafsson, S., Mägi, R., Ganna, A., et al. Genome-wide meta-analysis identifies 11 new loci for anthropometric traits and provides insights into genetic architecture. *Nat. Genet.* 45, 5 (2013), 501–512.
- Bloom, J.S., Ehrenreich, I.M., Loo, W.T., Lite, T.-L.V.L., et al. Finding the sources of missing heritability in a yeast cross. *Nature* 494, 7436 (2013), 234–237.
- Devlin, B., Roeder, K. Genomic control for association studies. *Biometrics* 55, 4 (1999), 997–1004.
- Fisher, R.A. The correlation between relatives on the supposition of Mendelian inheritance. *Trans. R. Soc. Edinb.* 52 (1918), 399–433.
- Golan, D., Rosset, S. Accurate estimation of heritability in genome wide studies using random effects models. *Bioinformatics* 27, 13 (2011), i317–i323.
- Gunderson, K.L., Steemers, F.J., Lee, G., Mendoza, L.G., et al. A genome-wide scalable SNP genotyping assay using microarray technology. *Nat. Genet.* 37, 5 (2005), 549–554.
- Hartl, D.L., Clark, A.G. Sunderland, MA: Sinauer Associates, 2007.
- Henderson, C.R. Best linear unbiased estimation and prediction under a selection model. *Biometrics* 31, 2 (1975), 423–447.
- Hindorf, L.A., Sethupathy, P., Junkins, H.A., Ramos, E.M., et al. Potential etiologic and functional implications of genome-wide association loci for human diseases and traits. *Proc. Natl. Acad. Sci. USA* 106, 23 (2009), 9362–9367.
- Kang, H.M., Sul, J.H., Service, S.K., Zaitlen, N.A., Kong, S.-Y.Y., Freimer, N.B., Sabatti, C., Eskin, E. Variance component model to account for sample structure in genome-wide association studies. *Nat. Genet.* 42, 4 (2010), 348–354.
- Kang, H.M., Zaitlen, N.A., Wade, C.M., Kirby, A., Heckerman, D., Daly, M.J., Eskin, E. Efficient control of population structure in model organism association mapping. *Genetics* 178, 3 (2008), 1709–1723.
- Kenny, E.E., Kim, M., Gusev, A., Lowe, J.K., Salit, J., Smith, J.G., Kovvali, S., Kang, H.M., Newton-Cheh, C., Daly, M.J., Stoffel, M., Altshuler, D.M., Friedman, J.M., Eskin, E., Breslow, J.L., Pe'er, I. Increased power of mixed models facilitates association mapping of 10 loci for metabolic traits in an isolated population. *Hum. Mol. Genet.* 20, 4 (2010), 827–839.
- Lippert, C., Listgarten, J., Liu, Y., Kadie, C.M., Davidson, R.I., Heckerman, D. Fast linear mixed models for genome-wide association studies. *Nat. Methods* 8, 10 (2011), 833–835.
- Macgregor, S., Cornes, B.K., Martin, N.G., Visscher, P.M. Bias, precision and heritability of self-reported and clinically measured height in Australian twins. *Hum. Genet.* 120, 4 (2006), 571–580.
- Manolio, T.A., Brooks, L.D., Collins, F.S. A HapMap harvest of insights into the genetics of common disease. *J. Clin. Invest.* 118, 5 (2008), 1590–1605.
- Manolio, T.A., Collins, F.S., Cox, N.J., Goldstein, D.B., et al. Finding the missing heritability of complex diseases. *Nature* 461, 7265 (2009), 747–753.
- Matsuzaki, H., Dong, S., Loi, H., Di, X., Liu, G., et al. Genotyping over 100,000 SNPs on a pair of oligonucleotide arrays. *Nat. Methods* 1, 2 (2004), 109–111.
- Meuwissen, T.H., Hayes, B.J., Goddard, M.E. Prediction of total genetic value using genome-wide dense marker maps. *Genetics* 157, 4 (2001), 1819–1829.
- Price, A.L., Patterson, N.J., Plenge, R.M., Weinblatt, M.E., et al. Principal components analysis corrects for stratification in genome-wide association studies. *Nat. Genet.* 38, 8 (2006), 904–909.
- Purcell, S.M., Wray, N.R., Stone, J.L., Visscher, P.M., et al. Common polygenic variation contributes to risk of schizophrenia and bipolar disorder. *Nature* 460, 7256 (2009), 748–752.
- Rakitsch, B., Lippert, C., Stegle, O., Borgwardt, K. A lasso multi-marker mixed model for association mapping with population structure correction. *Bioinformatics* 29, 2 (2012), 206–214.
- Randall, J.C., Winkler, T.W., Kutalik, Z., Berndt, S.I., et al. Sex-stratified genome-wide association studies including 270,000 individuals show sexual dimorphism in genetic loci for anthropometric traits. *PLoS Genet.* 9, 6 (2013), e1003500.
- Robinson, G.K. That BLUP is a good thing: The estimation of random effects. *Stat. Sci.* 6, 1 (1991), 15–32.
- Sabatti, C., Service, S.K., Hartikainen, A.-L.L., Pouta, A., et al. Genome-wide association analysis of metabolic traits in a birth cohort from a founder population. *Nat. Genet.* 41, 1 (2009), 35–46.
- Silventoinen, K., Sammalisto, S., Perola, M., Boomsma, D.I., et al. Heritability of adult body height: A comparative study of twin cohorts in eight countries. *Twin Res.* 6, 5 (2003), 399–408.
- Stram, D.O. *Design, Analysis, and Interpretation of Genome-Wide Association Scans*. Springer, 2013.
- The International HapMap Consortium. A haplotype map of the human genome. *Nature* 437, 7063 (2005), 1299.
- van Dongen, J., Slagboom, P.E., Draisma, H.H.M., et al. The continuing value of twin studies in the omics era. *Nat. Rev. Genet.* 7 (2012).
- Wellcome Trust Case Control Consortium. Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. *Nature* 447, 7145 (2007), 661–678.
- Welter, D., MacArthur, J., Morales, J., Burdett, T., et al. The NHGRI GWAS catalog, a curated resource of SNP-trait associations. *Nucl. Acids Res.* 42, Database issue (2014), D1001–D1006.
- Yang, J., Benyamin, B., McEvoy, B.P., Gordon, S., et al. Common SNPs explain a large proportion of the heritability for human height. *Nat. Genet.* 42, 7 (2010), 565–569.
- Zhou, X., Carbonetto, P., Stephens, M. Polygenic modeling with Bayesian sparse linear mixed models. *PLoS Genet.* 9, 2 (2013), e1003264.
- Zhou, X., Stephens, M. Genome-wide efficient mixed-model analysis for association studies. *Nat. Genet.* 44, 7 (2012), 821–824.
- Zuk, O., Hechter, E., Sunyaev, S.R., Lander, E.S. The mystery of missing heritability: Genetic interactions create phantom heritability. *Proc. Natl. Acad. Sci. USA* 109, 4 (2012), 1193–1198.

Eleazar Eskin (eeskin@cs.ucla.edu) is a professor in the Department of Computer Science and the Department of Human Genetics at the University of California, Los Angeles.

© 2015 ACM 00010782/15/10 \$15.00



Watch the author discuss his work in this exclusive *Communications* video. <http://cacm.acm.org/videos/discovering-genes-involved-in-disease-and-the-mystery-of-missing-heritability>

Are you looking for your next IT job?

Do you need Career Advice?

The **ACM Career & Job Center** offers ACM members a host of career-enhancing benefits:

- A **highly targeted focus** on job opportunities in the computing industry
- **Access to hundreds** of industry job postings
- Resume posting **keeping you connected** to the employment market while letting you maintain full control over your confidential information
- **Job Alert system** that notifies you of new opportunities matching your criteria
- **Career coaching** and guidance available from trained experts dedicated to your success
- **Free access** to a content library of the best career articles compiled from hundreds of sources, and much more!



Visit **ACM's Career & Job Center** at:
<http://jobs.acm.org>



Association for
Computing Machinery

Advancing Computing as a Science & Profession

The **ACM Career & Job Center** is the perfect place to begin searching for your next employment opportunity!

Visit today at **<http://jobs.acm.org>**

research highlights

P. 90

**Technical
Perspective**
**Not Just a Matrix
Laboratory Anymore**

By Cleve Moler

P. 91

**Computing Numerically with
Functions Instead of Numbers**

By Lloyd N. Trefethen

Technical Perspective

Not Just a Matrix Laboratory Anymore

By Cleve Moler

I NEVER DREAMED MATLAB would become what it is now. It began almost 40 years ago as a simple calculator my students could use for matrix operations. Today, extended by dozens of specialized toolboxes and Simulink (a block diagram environment for simulation and model-based design), MATLAB has evolved into a mature programming language supporting a rich technical computing environment. Its use has spread in sometimes surprising ways far beyond the original context of academia to a wide variety of applications in industry and business. So MATLAB has come a long way from being just a “matrix laboratory.”

As chief mathematician for MathWorks, I love to see the mathematics that underlies all these applications and ties everything together. The mathematics may be invisible to the user, since one of the objectives of some tools is to hide the math, but it can be found by poking deeply enough.

Mathematics is not difficult to find in Chebfun, the subject of the following paper, which began life in the early 2000s as an extension of MATLAB’s operations for discrete vectors and matrices to continuous functions. Before Chebfun, there were two different ways of computing with functions, meaning structures of the form “ $f(x)$.”

- Symbolic, exemplified by Mathematica and Maple. A function is represented by a list or a string; think of text.

- Numeric, exemplified by MATLAB. A function is represented by a finite-dimensional vector of floating-point values; think of a table.

The separation between these two representations is not clear-cut, since Mathematica and Maple can do purely numerical computation, and MATLAB has an add-on toolbox for symbolic computation.

Symbolic computation gives answers in the form you came to expect in your calculus class. But it soon suf-

MATLAB
has evolved into
a mature
programming
language
supporting
a rich technical
computing
environment.

fers from combinatorial explosion in both time and space as the complexity of the representation grows. (A telling example of this appears early in the paper.) And symbolic computation simply cannot solve most scientific and engineering problems because they do not have “closed form” answers. On the other hand, numerical computation suffers from many difficulties that stem from approximating continuous processes by discrete ones.

Chebfun combines the best of both worlds. It represents a function as a piecewise Chebyshev expansion, allowing Chebfun to appear to be doing (nearly exact) symbolic computation, but with the nimbleness and speed of numerical computation. Chebfun automatically chooses the number of interpolation points so the function is represented to roughly machine precision (IEEE double, approximately 15 decimal digits of relative accuracy). As in MATLAB, the underlying mathematics in Chebfun ties together all of the computations.

If you already know MATLAB, you

“know” Chebfun, whose guiding principle is to overload the definition of operations defined in MATLAB for vectors and matrices so they make sense when applied to chebfun, where a chebfun with a lowercase *c* denotes an object in the Chebfun system. For example, if *v* is a finite-dimensional vector in MATLAB, then `sum(v)` is the sum of the elements of *v*. Extension of this idea to functions means that, if *f* is a chebfun, `sum(f)` is the definite integral of $f(x)$ over its specified range of definition. Full details and a host of examples are given at <http://www.chebfun.org/about>.

The Chebfun project has made enormous progress for the one-dimensional case, when singularities and discontinuities can be detected automatically, intervals can be broken into subintervals, and piecewise expansions are available in which the breakpoints are specified scalar points. But in two dimensions, matters are much more complicated, as Nick Trefethen describes in his paper, and will be the subject of continuing activity by an expanding group of researchers. The success of Chebfun has already inspired further applications. Version 5 of Chebfun^a was released in June 2014 and is posted on GitHub.

Chebfun is a remarkable example of what mathematical research combined with software development, supported by systems like MATLAB, can produce. 

^a The history of Chebfun can be found at <http://www.chebfun.org/about/history.html>.

Cleve Moler (moler@mathworks.com) is the chief mathematician for MathWorks, Natick, MA.

Computing Numerically with Functions Instead of Numbers

By Lloyd N. Trefethen

Abstract

Science and engineering depend upon computation of functions such as flow fields, charge distributions, and quantum states. Ultimately, such computations require some kind of discretization, but in recent years, it has become possible in many cases to hide the discretizations from the user. We present the Chebfun system for numerical computation with functions, which is based on a key idea: an analogy of floating-point arithmetic for functions rather than numbers.

1. INTRODUCTION

The oldest problem of computing is, how can we calculate mathematical quantities? As other aspects of computing have entered into every corner of our lives, mathematical computation has become a less conspicuous part of computer science, but it has not gone away. On the contrary, it is bigger than ever, the basis of much of science and engineering.

The mathematical objects of interest in science and engineering are not just individual numbers but *functions*. To make weather predictions, we simulate velocity, pressure, and temperature distributions, which are multidimensional functions evolving in time. To design electronic devices, we compute electric and magnetic fields, which are also functions. Sometimes the physics of a problem is described by long-established differential equations such as the Maxwell or Schrödinger equations, but just because the equations are understood does not mean the problem is finished. It may still be a great challenge to solve the equations.

How do we calculate functions? The almost unavoidable answer is that they must be discretized in one way or another, so that derivatives, for example, may be replaced by finite differences. Numerical analysts and computational engineers are the experts at handling these discretizations.

As computers grow more powerful, however, a new possibility has come into play: hiding the discretizations away so that the scientist does not have to see them. This is not feasible yet for weather prediction, but for certain kinds of desktop computing, it is becoming a reality. This paper introduces the Chebfun software system, which has followed this vision from its inception in 2002. For functions of one variable, $f(x)$, the aim has been largely achieved, and progress is well underway for functions of two variables, $f(x, y)$.

Chebfun is built on an analogy. To work with real numbers on a computer, we typically approximate them to 16 digits by finite bit strings: *floating-point numbers*, with an associated concept of *rounding* at each step of a calculation. To work with functions, Chebfun approximates them to 16 digits by polynomials (or piecewise polynomials) of finite degree: Chebyshev expansions, again with an associated concept of rounding.

Thus the key to numerical computation with functions is the generalization of the ideas of floating-point approximation and rounding from numbers to functions.

2. A COMBINATORIAL EXPLOSION

Have not discretizations in general, and floating-point numbers in particular, been rendered superfluous by the introduction of symbolic systems like Mathematica or Maple? It is worth taking a moment to explain why the answer is no, for this will help elucidate the basis of our algorithms for numerical computing with functions.

We begin with what looks like an encouraging observation: if x and y are rational numbers, then so are $x + y$, $x - y$, xy , and x/y (assuming $y \neq 0$). Since rational numbers can readily be represented on computers, this might seem to suggest that there is no need for floating-point arithmetic with its inexact process of rounding. If a computer works in rational arithmetic, no error is ever made, so it might seem that, in principle, much of numerical computation could be carried out exactly.

The first obstacle we encounter is that not every interesting real number x is rational (think of the hypotenuse of a triangle). However, this alone is not a serious problem, as x can be approximated arbitrarily closely by rationals.

The bigger problem is that when we try to construct such approximations by practical algorithms, we run into combinatorial or exponential explosions. For example, suppose we wish to find a root of the polynomial

$$p(x) = x^5 - 2x^4 - 3x^3 + 3x^2 - 2x - 1.$$

We can approximate an answer to great accuracy by rational numbers if we take a few steps of Newton's method, taught in any introductory numerical analysis course. Let us do this, beginning from the initial guess $x^{(0)} = 0$. The startling result is shown in Table 1.

There is a problem here! As approximations to an exact root of p , the rational numbers displayed in the table are accurate to approximately 0, 0, 1, 3, 6, and 12 digits, respectively; the number of useful digits doubles at each step thanks to the quadratic convergence of Newton's method. Yet the lengths of the numerators are 1, 1, 2, 10, 53, and 265 digits, expanding by a factor of about 5 at each step since the degree of p is 5. After three more steps, we will have an answer $x^{(8)}$ accurate to 100 digits, but represented by numerator and denominator each about 33,125 digits long, and storing it will require 66 kB.

The original version of this paper was published with the same title in *Mathematics in Computer Science* 1 (2007), 9–19.

Table 1. Five steps of Newton's method in rational arithmetic to find a root of a quintic polynomial.

$$\begin{aligned}
x^{(0)} &= 0 \\
x^{(1)} &= -\frac{1}{2} \\
x^{(2)} &= -\frac{22}{95} \\
x^{(3)} &= -\frac{11414146527}{36151783550} \\
x^{(4)} &= -\frac{43711566319307638440325676490949986758792998960085536}{138634332790087616118408127558389003321268966090918625} \\
x^{(5)} &= -\frac{7243914791768201761290013818789259730350038836047543931178041194343579260105802744696299228820641845856700177035519963166516115634363}{22974602373157587333399081666432003514775984720802108866006687478324948875098845198224797582289844718084679832592257179299176857894449} \\
&\quad \frac{4562735299921308664663139405767412052875538201240642484300698212354536105198706894715223176068754569028985198376505504345452967921}{1536221568972260935865495519518216876316931568370465908144002495419674804116675018139752278347161906687414800535564210785107754250}
\end{aligned}$$

If we were so foolish as to try to take 20 steps of Newton's method in this mode, we would need 16 TB to store the result.

Such difficulties are ubiquitous. Rational computations, and symbolic computations in general, have a way of expanding exponentially. If nothing is done to counter this effect, computations grind to a halt because of excessive demands on computing time and memory. This is ultimately the reason why symbolic computing, though powerful when it works, plays such a circumscribed role in computational science. As an example with more of a flavor of functions rather than numbers, suppose we want to know the indefinite integral of the function

$$f(x) = e^x \cos^5(6x) \sin^6(5x).$$

This happens to be a function that can be integrated analytically, but the result is not simple. The Wolfram Mathematica Online Integrator produces an answer that consists of the expression

$$\frac{5e^x(24 \sin(24x) + \cos(24x))}{295424}$$

plus 20 other terms of similar form, with denominators ranging from 512 to 3,687,424. Working with such expressions is unwieldy when it is possible at all. An indication of their curious status is that if I wanted to be confident that this long formula was right, the first thing I would do would be to see if it matched results from a numerical computation.

3. FLOATING-POINT ARITHMETIC

It is in the light of such examples that I would like to consider the standard alternative to rational arithmetic, namely floating-point arithmetic. As is well known, this is the idea of representing numbers on computers by, for example, 64-bit binary words containing 53 bits (≈ 16 digits) for a fraction and 11 for an exponent. (These parameters correspond to the IEEE double precision standard.) Konrad Zuse invented floating-point arithmetic in Germany before World War II, and the idea was developed by IBM and other manufacturers a few years later. The IEEE standardization came in the

mid-1980s and is beautifully summarized in the book by Overton.¹⁵ For more up-to-date details, see Muller et al.¹⁴

There are two aspects to floating-point technology: a *representation* of real (and complex) numbers via a subset of the rationals and a prescription for *rounded arithmetic*. These principles combine to halt the combinatorial explosion. Thus, for example, if two 53-bit numbers are multiplied, the mathematically exact result would require about 106 bits to be represented. Instead of accepting this, we round the result down to 53 bits again. More generally, most floating-point arithmetic systems adhere to the following principle: when an operation $+$, $-$, $\times$, $/$ is performed on two floating-point numbers, the output is the exactly correct result rounded to the nearest floating-point number, with ties broken by a well-defined rule. This implies that every floating-point operation is exact except for a small relative error:

$$\text{computed}(x * y) = (x * y)(1 + \varepsilon), \quad |\varepsilon| \leq \varepsilon_{\text{mach}}. \quad (1)$$

Here $*$ denotes one of the operations $+$, $-$, $\times$, $/$, and we are ignoring the possibilities of underflow or overflow. The IEEE double precision value of “machine epsilon” is $\varepsilon_{\text{mach}} = 2^{-53} \approx 1.1 \times 10^{-16}$.

Equation (1) implies an important corollary: when two floating-point numbers x and y are combined on the computer by an operation $*$, the result $\text{computed}(x * y)$ is exactly equal to $\tilde{x} * \tilde{y}$ for some two numbers $\tilde{x}$ and $\tilde{y}$ that are close to x and y in a relative sense:

$$\text{computed}(x * y) = \tilde{x} * \tilde{y}, \quad \frac{|x - \tilde{x}|}{|x|}, \frac{|y - \tilde{y}|}{|y|} \leq \varepsilon_{\text{mach}}. \quad (2)$$

Numerical analysts say that the operations $+$, $-$, $\times$, $/$ are *backward stable*, delivering the exactly correct results for inputs that are slightly perturbed from their correct values in a relative sense. The same conclusion holds or nearly holds for good implementations of other elementary operations, often unary instead of binary, such as $\sqrt{}$, $\exp$, or $\sin$.¹⁴

Floating-point arithmetic is not widely regarded as one of computer science's sexier topics. A common view is that it is an ugly but necessary engineering compromise. We cannot do arithmetic honestly, the idea goes, so we cheat a bit—unfortunate, but unavoidable, or as some have called it, a

“Faustian bargain.” In abandoning exact computation, we sell our souls, and in return, we get some numbers.

I think one can take a more positive view. Floating-point arithmetic is an *algorithm*, no less than a general procedure for containing the combinatorial explosion. Consider the Newton iteration of Table 1 again, but now carried out in IEEE 16-digit arithmetic:

```
x(0) = 0.000000000000000,
x(1) = -0.500000000000000,
x(2) = -0.33684210526316,
x(3) = -0.31572844839629,
x(4) = -0.31530116270328,
x(5) = -0.31530098645936,
x(6) = -0.31530098645933,
x(7) = -0.31530098645933,
x(8) = -0.31530098645933.
```

It is the same process as before, less startling without the exponential explosion, but far more useful. Of course, though these numbers are printed in decimal, what is really going on in the computer is binary. The exact value at the end, for example, is not the decimal number printed but

```
x(8) = -0.010100001011011110010000 ...
      11000001001111010100011110001binary.
```

Abstractly speaking, when we compute with rational numbers, we might proceed like this:

*Compute an exact result,
then round it to a certain number of bits.*

The problem is that the exact result is often exponentially lengthy. Floating-point arithmetic represents an alternative idea:

*Round the computation at every step,
not just at the end.*

This strategy has proved spectacularly successful. At a stroke, combinatorial explosion ceases to be an issue. Moreover, so long as the computation is not numerically unstable in a sense understood thoroughly by numerical analysts, the final result will be accurate. This is what one observes in practice, and it is also the rigorous conclusion of theoretical analysis of thousands of algorithms investigated by generations of numerical analysts.¹²

4. CHEBFUN

Chebfun is an open-source software system developed over the past decade at Oxford by myself and a succession of students and postdocs including Zachary Battles, Ásgeir Birkisson, Nick Hale, and Alex Townsend, as well as Toby Driscoll at the University of Delaware (a full list can be found in the Acknowledgments and at www.chebfun.org). The aim of Chebfun is to extend the ideas we have just discussed from numbers to functions. Specifically, Chebfun works with piecewise smooth real or complex functions defined on an interval $[a, b]$, which by default is $[-1, 1]$. A function is represented by an object known as a *chebfun*. (We write “Chebfun” as the name of the system and “chebfun” for the representation of an individual function.) If f and g are chebfuns, we can

perform operations on them such as $+$, $-$, $\times$, $/$, as well as other operations like $\exp$ or $\sin$. The intention is not that such computations will be exact. Instead, the aim is to achieve an analogue of Equation (2) for functions,

$$\text{computed}(f * g) = \tilde{f} * \tilde{g}, \quad \frac{\|f - \tilde{f}\|}{\|f\|}, \frac{\|g - \tilde{g}\|}{\|g\|} \leq C \varepsilon_{\text{mach}} \quad (3)$$

(again ignoring underflow and overflow), where C is a small constant, with a similar property for unary operations. Here $\|\cdot\|$ is a suitable norm such as $\|\cdot\|_{\infty}$. Thus the aim of Chebfun is *normwise backward stable computation of functions*. We shall say more about the significance of (3) in Section 6.

Chebfun is implemented in MATLAB, a language whose object-oriented capabilities enable one to overload operations such as $+$, $-$, $\times$, $/$, $\sin$, and $\exp$ with appropriate alternatives. Some of the methods defined for chebfuns are as follows (this list is about one-third of the total):

abs	csc	kron	real
acos	cumprod	legpoly	remez
airy	cumsum	length	roots
angle	diff	log	round
arclength	dirac	max	sec
asin	eq	mean	semilogy
atan	erf	min	sign
atanh	exp	minus	sin
besselj	feval	mod	sinh
bvp4c	find	norm	spline
ceil	floor	null	sqrt
chebpade	gmres	ode45	std
chebpoly	heaviside	pinv	sum
chebpolyplot	imag	plot	svd
cond	integral	plus	tanh
conj	interp1	poly	times
conv	inv	polyfit	transpose
cos	isequal	prod	var
cosh	isinf	qr	waterfall
cot	isnan	rank	
coth	jacpoly	rank	

MATLAB (or Python) programmers will recognize many of these as standard commands. In MATLAB, such commands apply to discrete vectors, or sometimes matrices, but in Chebfun, they perform continuous analogues of the operations on chebfuns. Thus, for example, `log(f)` and `sinh(f)` deliver the logarithm and the hyperbolic sine of a chebfun f , respectively. More interestingly, `sum(f)` produces the definite integral of f from a to b (a scalar), the analogue for continuous functions of the sum of entries of a vector. Similarly, `cumsum(f)` produces the indefinite integral of f (a chebfun), `diff(f)` computes the derivative (another chebfun), and `roots(f)` finds the roots in the interval $[a, b]$ (a vector of length equal to the number of roots).

Mathematically, the basis of Chebfun—and the origin of its name—is piecewise Chebyshev expansions. Let T_j denote the Chebyshev polynomial $T_j(x) = \cos(j \cos^{-1} x)$, of degree j , which equioscillates between $j+1$ extrema ± 1 on $[-1, 1]$. The Chebyshev series for any Hölder continuous $f \in C[-1, 1]$ is defined by²²

$$f(x) = \sum_{j=0}^{\infty} a_j T_j(x), \quad a_j = \frac{2}{\pi} \int_{-1}^1 \frac{f(x) T_j(x)}{\sqrt{1-x^2}} dx, \quad (4)$$

where the prime indicates that the term with $j = 0$ is multiplied by $1/2$. (These formulas can be derived using the change of variables $x = \cos \theta$ from the Fourier series for the 2π -periodic even function $f(\cos \theta)$. Chebyshev series are essentially the same as Fourier series, but for nonperiodic functions.) Chebfun is based on storing and manipulating coefficients $\{a_j\}$ for such expansions. Many of the algorithms make use of the equivalent information of samples $f(x_j)$ at *Chebyshev points*,

$$x_j = \cos \frac{j\pi}{n}, \quad 0 \leq j \leq n \quad (5)$$

and one can go back and forth to the representation of Equation (4) as needed by means of the Fast Fourier Transform (FFT). Each chebfun has a fixed finite n chosen to be large enough for the representation, according to our best estimate, to be accurate in the local sense (Equation (3)) to 16 digits. Given data $f_j = f(x_j)$ at the Chebyshev points (Equation (5)), other values can be determined by the *barycentric interpolation formula*,¹⁸

$$f(x) = \sum_{j=0}^n \frac{w_j}{x - x_j} f_j \bigg/ \sum_{j=0}^n \frac{w_j}{x - x_j}, \quad (6)$$

where the weights $\{w_j\}$ are defined by

$$w_j = (-1)^j \delta_j, \quad \delta_j = \begin{cases} 1/2, & j=0 \text{ or } j=n, \\ 1, & \text{otherwise.} \end{cases} \quad (7)$$

(If x happens to be exactly equal to some x_j , one bypasses Equation (6) and sets $f(x) = f(x_j)$.) This method is known to be numerically stable, even for polynomial interpolation in millions of points.¹³

If f is analytic on $[-1, 1]$, its Chebyshev coefficients $\{a_j\}$ decrease exponentially.²² If f is not analytic but still several times differentiable, they decrease at an algebraic rate determined by the number of derivatives. It is these properties of rapid convergence that Chebfun exploits to be a practical computational tool. Suppose a chebfun is to be constructed, for example, by the statement

```
f = chebfun(@(x) sin(x)).
```

What happens when this command is executed is that the system performs adaptive calculations to determine what degree of polynomial approximation is needed to represent $\sin(x)$ to about 15 digits of accuracy. The answer in this case turns out to be 13, so that our 15-digit approximation is actually

$$\begin{aligned} f(x) = & 0.88010117148987T_1(x) - 0.03912670796534T_3(x) \\ & + 0.00049951546042T_5(x) - 0.00000300465163T_7(x) \\ & + 0.00000001049850T_9(x) - 0.00000000002396T_{11}(x) \\ & + 0.00000000000004T_{13}(x), \end{aligned}$$

when represented in the well-behaved basis of Chebyshev polynomials $\{T_k\}$, or

$$\begin{aligned} f(x) = & 1.00000000000000x - 0.16666666666665x^3 \\ & + 0.008333333333314x^5 - 0.00019841269737x^7 \\ & + 0.0000027572913x^9 - 0.00000002504820x^{11} \\ & + 0.00000000015785x^{13} \end{aligned}$$

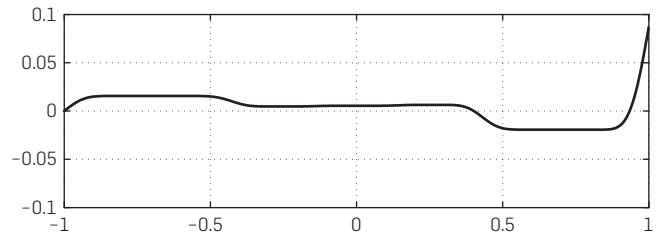
in the badly behaved but more familiar basis of monomials. This is a rather short chebfun; more typically, the length might be 50 or 200. For example, `chebfun(@(x) sin(50*x))` has length 90, and `chebfun(@(x) exp(-1./x.^2))` has length 219.

Having settled on representing functions by Chebyshev expansions and interpolants, we next face the question of how to implement mathematical operations such as those in the list above. This is a very interesting matter, and details of the many algorithms used in Chebfun can be found in Trefethen²² and the other references. For example, zeros of chebfuns are found by `roots` by a recursive subdivision of the interval combined with eigenvalue computations for Chebyshev “colleague matrices,”⁴ and global maxima and minima are determined by `max` and `min` by first finding zeros of the derivative. All these computations are fast and accurate even when the underlying polynomial representations have degrees in the thousands.

At the end of Section 2, we considered an indefinite integral. In Chebfun indefinite integration is carried out by the command `cumsum`, as mentioned above, and that example on the interval $[-1, 1]$ could go like this:

```
x = chebfun(@(x) x);
f = exp(x) .* cos(6*x) .^5 .* sin(5*x) .^6;
g = cumsum(f);
```

The chebfun g is produced in about 0.02 s on a desktop machine, a polynomial of degree 94 accurate to about 16 digits. Here is a plot:



5. TAMING THE EXPLOSION

As mentioned earlier, when two 53-bit numbers are multiplied, an exact result would normally require 106 bits, but floating-point arithmetic rounds this to 53. Chebfun implements an analogous compression for polynomial approximations of functions as opposed to binary approximations of numbers. For example, suppose x is the chebfun corresponding to the linear function x on $[-1, 1]$. If we execute the commands

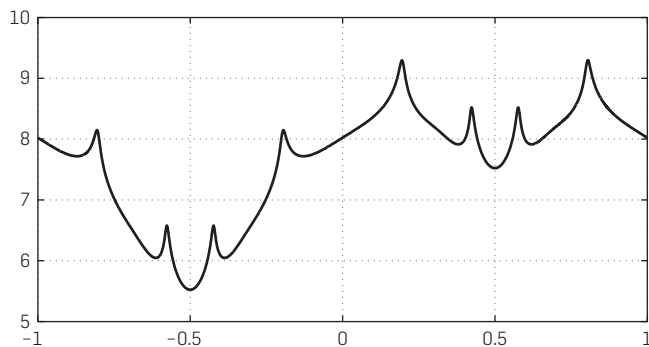
```
f = sin(x), g = cos(x), h = f.*g,
```

we find that the chebfuns f and g have degrees 13 and 14, respectively. One might expect their product to have degree 27, but in fact, h has degree only 17. This happens because at every step, the system automatically discards Chebyshev coefficients that are below machine precision—just as floating-point arithmetic discards bits below the 53rd. The degree grows only as the complexity of the functions involved genuinely grows, as measured on the scale of machine epsilon.

Here is an example to illustrate how this process contains the explosion of polynomial degrees. The program

```
f = chebfun(@(x) sin(pi*x));
s = f;
for j = 1:15
    f = (3/4)*(1 - 2*f.^4);
    s = s + f;
end
plot(s)
```

begins by constructing a chebfun f corresponding to the function $\sin(\pi x)$ on the interval $[-1, 1]$, with degree 19. Then it takes 15 steps of an iteration that raises the current f to the fourth power at each step. The result after a fraction of a second on a desktop computer is a rather complicated chebfun, of degree 3378, which looks like this:



The degree 3378 may seem high, but it is very low compared to what it would be if the fourth powers were computed without dropping small coefficients, namely $19 \times 4^{15} = 20,401,094,656!$ Thus the complexity has been curtailed by a factor of millions, yet with little loss of accuracy. For example, the command `roots(s-8)` now takes less than a second to compute the 12 points $x \in [-1, 1]$ with $s(x) = 8$:

```
-0.99293210741191
-0.81624993429017
-0.79888672972343
-0.20111327027657
-0.18375006570983
-0.00706789258810
0.34669612041826
0.40161707348210
0.44226948963247
0.55773051036753
0.59838292651791
0.65330387958174
```

These results are all correct except in the last digit.

Once one has a chebfun representation, further computations are easy. For example, `sum(s)` returns the definite integral 15.26548382582674 in a few thousands of a second. The exact value is 15.26548382582674700943...

6. NORMWISE BACKWARD STABILITY

Does Chebfun live up to the vision of an analogue for functions of floating-point arithmetic for numbers? While considering

this question, a good starting point is the normwise backward stability condition Equation (3), and in particular, it is productive to focus on two questions:

- (I) How close does Chebfun come to achieving Equation (3)?
- (II) What are the implications of this condition?

The answer to (I) appears to be that Chebfun does satisfy Equation (3), at least for the basic operations $+$, $-$, $\times$, $/$. This has not been proved formally, and it is a project for the future to develop a rigorous theory. To explain how Equation (3) can hold, let us consider the mode in which each chebfun is represented precisely by a finite Chebyshev series with floating-point coefficients (instead of values at Chebyshev points). The property of Equation (3) for $+$ and $-$ stems from the corresponding properties for addition and subtraction of floating-point numbers, together with the numerical stability of barycentric interpolation.¹³ For multiplication, the argument is only slightly more complicated, since again the operation comes down to one of Chebyshev coefficients. The more challenging fundamental operation is division, for this case, the quotient f/g is sampled pointwise at various Chebyshev points and then a new Chebyshev series is constructed by the adaptive process used generally for chebfun construction. It is not known whether the current code contains safeguards enough to give a guarantee of Equation (3), and this is a subject for investigation. In addition, it will be necessary to consider analogues of Equation (3) for other Chebfun operations besides $+$, $-$, $\times$, $/$.

This brings us to (II), the question of the implications of Equation (3). The easier part of the answer, at least for numerical analysts familiar with backward error analysis, is to understand exactly what the property of Equation (3) does and does not assert about numerical accuracy. A crucial fact is that the bound involves the global norms of the function f and g , not their values at particular points. For example, we may note that if two chebfuns f and g give $(f-g)(x) < 0$ at a point x , then from Equation (3), we cannot conclude that $f(x) < g(x)$. We can conclude, however, that there are nearby chebfuns $\tilde{f}$ and $\tilde{g}$ with $\tilde{f}(x) < \tilde{g}(x)$. This is related to the “zero problem” that comes up in the theory of real computation.²⁴ It is well known that the problem of determining the sign of a difference of real numbers with guaranteed accuracy poses difficulties. However, Chebfun makes no claim to overcome these difficulties: the normwise condition of Equation (3) promises less.

Does it promise enough to be useful? What strings of computations in a system satisfying Equation 3 at each step can be expected to be satisfactory? This is nothing less than the problem of *stability of Chebfun algorithms*, and it is a major topic for future research. Certainly, there may be applications where Equation (3) is not enough to imply what one would like typically for reasons related to the zero problem. For example, this may happen in some problems of geometry, where arbitrarily small coordinate errors may make the difference between two bodies intersecting or not intersecting or between convex and concave. On the other hand, generations of numerical analysts have found that such difficulties are by no means universal, that the backward stability condition of Equation (2) for floating-point arithmetic is sufficient to ensure success for many scientific computations. An aim of ours

for the future will be to determine how far this conclusion carries over to the condition of Equation (3) for chebfuns.

7. CHEBFUN SOFTWARE PROJECT

Chebfun began in 2002 as a few hundred lines of MATLAB code, written by Zachary Battles, for computing with global polynomial representations of smooth functions on $[-1, 1]$, and this “core Chebfun” framework has been the setting for the discussion in this article. But in fact, the project has expanded greatly in the decade since then, both as a software effort and in its computational capabilities.

In terms of software, we have grown to an open-source project hosted on GitHub with currently about a dozen developers, most but not all based at Oxford. The code is written in MATLAB, which is a natural choice for this kind of work because of its vector and matrix operations, although implementations of parts of core Chebfun have been produced by various people in other languages including Python, C, Julia, Maxima, and Octave. To date, there have been about 20,000 Chebfun downloads. We interact regularly with users through bug reports, help requests by email, and other communications, but we believe we are not alone among software projects in feeling that we have an inadequate understanding of who our users are and what they are doing.

In terms of capabilities, here are some of the developments beyond the core ideas emphasized in this article. The abbreviations ODE and PDE stand for ordinary and partial differential equations.

- piecewise smooth functions¹⁶
- periodic functions (Fourier not Chebyshev)⁷
- fast edge detection for determining breakpoints¹⁶
- infinite intervals $[a, \infty)$, $(-\infty, b]$, $(-\infty, \infty)$
- functions with poles and other singularities
- delta functions of arbitrary order
- Padé, Remez, CF rational approximations^{8, 17, 23}
- fast Gauss and Gauss–Jacobi quadrature^{9, 11}
- fast Chebyshev $\leftrightarrow$ Legendre conversions¹⁰
- continuous QR factorization, SVD, least-squares^{1, 21}
- representation of linear operators⁶
- solution of linear ODEs⁶
- solution of integral equations⁵
- solution of eigenvalue problems⁶
- exponentials of linear operators⁶
- Fréchet derivatives via automatic differentiation²
- solution of nonlinear ODEs²
- PDEs in one space variable plus time
- Chebgui interface to ODE/PDE capabilities
- Chebfun2 extension to rectangles in 2D^{19, 20}

We shall not attempt to describe these developments, but here are a few comments. For solving ODE boundary value problems, whether scalars or systems and smooth or just piecewise smooth, Chebfun and its interface Chebgui have emerged as the most convenient and flexible tool in existence, making it possible to solve all kinds of problems with minimal effort with accuracy close to machine precision (these developments are due especially to Ásgeir Birkisson, Toby Driscoll, and Nick Hale).² For computing quadrature

nodes and weights, convolution, and conversion between Legendre and Chebyshev coefficient representations, Chebfun contains codes implementing new algorithms that represent the state of the art, enabling machine accuracy for even millions of points in seconds (these developments are due to Nick Hale, Alex Townsend, and Ignace Bogaert^{3, 9, 10}). Extensions to multiple dimensions have begun with Alex Townsend’s Chebfun2 code initially released in 2013.^{19, 20}

The best way to get a sense of the wide range of problems that can be solved by this kind of computing is to look at the collection of Chebfun Examples available online at the web site www.chebfun.org. Approaching 200 in number, the examples are organized under headings that look like chapters of a numerical analysis textbook (optimization, quadrature, linear algebra, geometry, ...), with dozens of short discussions in each category of problems ranging from elementary to advanced.

Here is an example that gives a taste of Chebfun’s ability to work with functions that are only piecewise smooth and to solve ODE eigenvalue problems. The sequence

```
x = chebfun(@(x) x, [-2, 2]);
V = max(x.^2/2, 1-2*abs(x));
quantumstates(V),
```

produces the plot shown in Figure 1 as well as associated numerical output. The figure shows the first 10 eigenmodes of a Schrödinger operator $-h^2 \partial^2 u / \partial x^2 + V(x)u(x)$ with the default value of Planck’s constant $h = 0.1$. The potential function $V(x)$ consists of the parabola $x^2/2$ over the interval $[-2, 2]$ maximized with a triangular barrier around $x = 0$, and it is represented by a piecewise-smooth chebfun with four pieces. This kind of mathematics arises in any introductory quantum mechanics course; Chebfun makes exploring the dependence of eigenstates on potential functions almost effortless, yet with accuracy close to machine precision.

And here is an example that gives a taste of Chebfun-like computing on rectangles in 2D as implemented by Townsend’s extension Chebfun2. The sequence

Figure 1. Schrödinger eigenstates computed by `quantumstates(V)`, where `V` is a chebfun representing a piecewise smooth potential function.

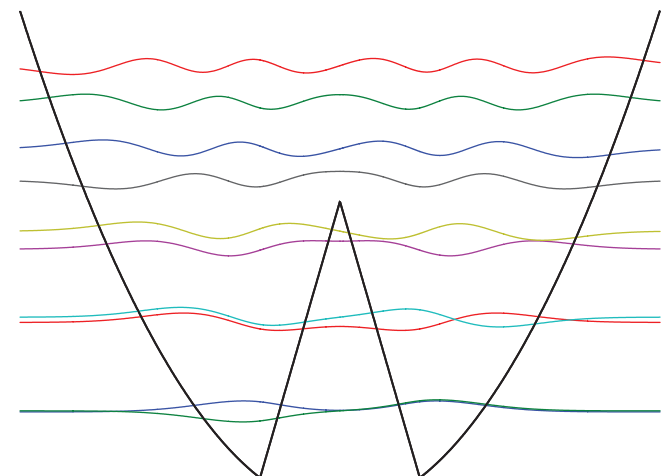
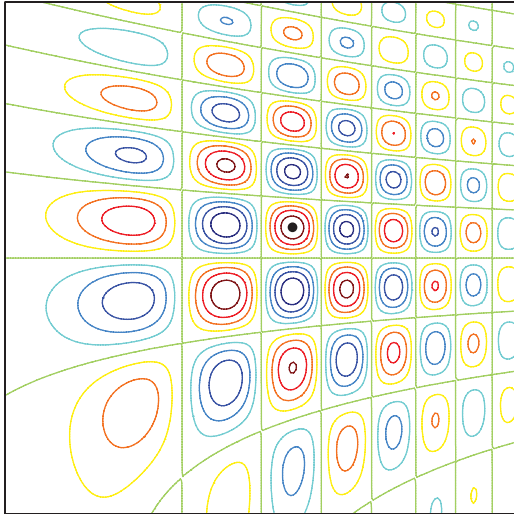


Figure 2. Two-dimensional extension of Chebfun: an oscillatory function represented by a chebfun2, with its maximum shown as a black dot.



```
f = chebfun2(@(x,y) exp(-(x.^2+y.^2))...
    .*sin(6*(2+x).*x).*sin(4*(3+x+y).*y));
contour(f),
```

defines and plots a chebfun2 representing an oscillatory function of x and y on the unit square $[-1, 1]^2$, as shown in Figure 2. The command `max2` tells us its global maximum in a fraction of a second:

```
max2(f)
ans = 0.970892994917307.
```

The algorithms underlying Chebfun2 are described in Townsend and Trefethen.^{19,20}

8. CONCLUSION

Chebfun is being used by scientists and engineers around the world to solve one-dimensional and two-dimensional numerical problems without having to think about the underlying discretizations. The Chebyshev technology it is built on is powerful, and it is hard to see any serious competition for this kind of high-accuracy representation of functions in 1D.

At the same time, the deeper point of this article has been to put forward a vision that is not tied specifically to Chebyshev expansions or to other details of Chebfun. The vision is that by the use of adaptive high-accuracy numerical approximations of functions, computational systems can be built that “feel symbolic but run at the speed of numerics.”

Acknowledgments

In addition to the leaders mentioned at the beginning of Section 4, other contributors to the Chebfun project have included: Anthony Austin, Folkmar Bornemann, Filomena di Tommaso, Pedro Gonnet, Stefan Güttel, Hrothgar, Mohsin Javed, Georges Klein, Hadrien Montanelli, Sheehan Olver, Ricardo Pachón, Rodrigo Platte, Mark Richardson, Joris Van Deun, Grady Wright, and Kuan Xu. It has been a fascinating experience working with these people over the

past decade to rethink so much of discrete numerical mathematics in a continuous mode.

During 2008–2011, the Chebfun project was supported by the UK Engineering and Physical Sciences Council. Currently, we are supported by MathWorks, Inc. and by the European Research Council under the European Union’s Seventh Framework Programme (FP7/2007–2013)/ERC grant agreement no. 291068. The views expressed in this article are not those of the ERC or the European Commission, and the European Union is not liable for any use that may be made of the information contained here. G

References

- Battles, Z., Trefethen, L.N. An extension of MATLAB to continuous functions and operators. *SIAM J. Sci. Comput.* 25 (2004), 1743–1770.
- Birkisson, Á., Driscoll, T.A. Automatic Fréchet differentiation for the numerical solution of boundary-value problems. *ACM Trans. Math. Softw.* 38, 26 (2012), 1–28.
- Bogaert, I. Iteration-free computation of Gauss–Legendre quadrature nodes and weights. *SIAM J. Sci. Comput.* 36 (2014), A1008–A1026.
- Boyd, J.A. Computing zeros on a real interval through Chebyshev expansion and polynomial rootfinding. *SIAM J. Numer. Anal.* 40 (2002), 1666–1682.
- Driscoll, T.A. Automatic spectral collocation for integral, integro-differential, and integrally reformulated differential equations. *J. Comput. Phys.* 229 (2010), 5980–5998.
- Driscoll, T.A., Bornemann, F., Trefethen, L.N. The Chebop system for automatic solution of differential equations. *BIT Numer. Math.* 48 (2008), 701–723.
- Driscoll, T.A., Hale, N., Trefethen, L.N. *Chebfun Guide*. Pafnuty Publications, Oxford, UK, 2014 (freely available at www.chebfun.org).
- Gonnet, P., Pachón, R., Trefethen, L.N. Robust rational interpolation and least-squares. *Elect. Trans. Numer. Anal.* 38 (2011), 146–167.
- Hale, N., Townsend, A. Fast and accurate computation of Gauss–Legendre and Gauss–Jacobi quadrature nodes and weights. *SIAM J. Sci. Comput.* 35 (2013), A652–A674.
- Hale, N., Townsend, A. A fast, simple, and stable Chebyshev–Legendre transform using an asymptotic formula. *SIAM J. Sci. Comput.* 36 (2014), A148–A167.
- Hale, N., Trefethen, L.N. Chebfun and numerical quadrature. *Sci. China Math.* 55 (2012), 1749–1760.
- Higham, N.J. *Accuracy and Stability of Numerical Algorithms*, 2nd edn. SIAM, Philadelphia, PA, 2002.
- Higham, N.J. The numerical stability of barycentric Lagrange interpolation. *IMA J. Numer. Anal.* 24 (2004), 547–556.
- Muller, J.-M., et al. *Handbook of Floating-Point Arithmetic*. Birkhäuser, Boston, 2010.
- Overton, M.L. *Numerical Computing with IEEE Floating Point Arithmetic*. SIAM, Philadelphia, PA, 2001.
- Pachón, R., Platte, R., Trefethen, L.N. Piecewise-smooth chebfuns. *IMA J. Numer. Anal.* 30 (2010), 898–916.
- Pachón, R., Trefethen, L.N. Barycentric-Remez algorithms for best polynomial approximation in the chebfun system. *BIT Numer. Math.* 49 (2009), 721–741.
- Salzer, H.E. Lagrangian interpolation at the Chebyshev points $x_{n,v} = \cos(v\pi/n)$, $v = 0(1)n$; some unnoted advantages. *Computer J.* 15 (1972), 156–159.
- Townsend, A., Trefethen, L.N. An extension of Chebfun to two dimensions. *SIAM J. Sci. Comput.* 35 (2013), C495–C518.
- Townsend, A., Trefethen, L.N. Continuous analogues of matrix factorizations. *Proc. Roy. Soc. Lond. A* 471 (2015), 20140585.
- Trefethen, L.N. Householder triangularization of a quasimatrix. *IMA J. Numer. Anal.* 30 (2010), 887–897.
- Trefethen, L.N. *Approximation Theory and Approximation Practice*. SIAM, Philadelphia, PA, 2013.
- Van Deun, J., Trefethen, L.N. A robust implementation of the Carathéodory–Fejér method for rational approximation. *BIT Numer. Math.* 51 (2011), 1039–1050.
- Yap, C.K. Theory of real computation according to EGC. In *Reliable Implementation of Real Number Algorithms: Theory and Practice*, Volume 5045 of Lecture Notes in Computer Science P. Hertling, C.M. Hoffmann, W. Luther, and N. Revol, eds. Springer-Verlag, Berlin Heidelberg, 2008, 193–237.

Lloyd N. Trefethen (trefethen@maths.ox.ac.uk), Mathematical Institute, University of Oxford, U.K.

CAREERS

Davidson College Assistant Professor in Computer Science

Davidson College invites applications for a tenure-track appointment at the **Assistant Professor level in Computer Science**, targeted to candidates with interest and expertise in systems topics such as operating systems, distributed systems, computer networks, database systems, or computer architecture. We seek faculty members with broad teaching and research interests who will support and enhance the computer science curriculum at all levels, and who can collaborate with colleagues and students across disciplinary lines in a liberal arts environment. Excellence in classroom teaching and an active research program in which undergraduate students can participate are essential. The ideal candidate will have an aptitude for and interest in helping guide the expansion of our existing computer science program into a major. The teaching load is four courses in the first year, and five courses per year thereafter. Davidson is strongly committed to achieving excellence and cultural diversity and welcomes applications from women, members of minority groups, and others who would bring additional dimensions to the college's mission. Consistently ranked among the nation's top liberal arts colleges, Davidson College is a highly selective, independent liberal arts college located in Davidson, North Carolina, close to the city of Charlotte. Davidson faculty enjoy a low student-faculty ratio, emphasis on and appreciation of excellence in teaching, and a collegial, respectful atmosphere that honors academic achievement and integrity. See www.davidson.edu/math for further information and jobs.davidson.edu to apply. Applications received by November 20, 2015, will receive fullest consideration.

Indiana University School of Informatics and Computing Faculty Positions in Computer Science and Informatics

The School of Informatics and Computing (SoIC) at Indiana University Bloomington invites applications for faculty positions in **computer science, health informatics, and security informatics**. Positions are open at all levels (assistant, associate, or full professor). Duties include teaching, research, and service.

Computer science applications are especially encouraged in the areas of databases, machine learning, and systems (particularly cyber-physical systems, parallelism, and networks).

Health informatics applications are especially encouraged in the areas of patient-facing technologies, including but not limited to novel technologies used by patients outside the clinical setting.

Security informatics applications are welcome from information and computer scientists in a wide range of areas including but not limited to us-

able security, human-centered design, identity, social informatics of security, and design for privacy.

Applicants should have an established record (for senior level) or demonstrable potential for excellence (for junior level) in research and teaching, and a PhD in a relevant area or (for junior level) expected before 8/16.

The SoIC is the first of its kind and among the largest in the country, with unsurpassed breadth. Its mission is to excel and lead in education, research, and outreach spanning and integrating the full breadth of computing and information technology. It includes Computer Science, Informatics, and Information and Library Science, with over 100 faculty, 900 graduate students, and 1500 undergraduate majors on the Bloomington Campus. It offers PhDs in Computer Science, Informatics, and Information Science.

Bloomington is a culturally thriving college town with a moderate cost of living and the amenities for an active lifestyle. Indiana University is renowned for its top-ranked music school, high-performance computing and networking facilities, and performing and fine arts.

All applicants should submit a CV, a statement of research and teaching, and names of 6 references (3 for junior level) using the links below (preferred) or to Faculty Search, SoIC, 919 E 10th St, Bloomington, IN 47408. Questions may be sent to hire@soic.indiana.edu. For full consideration applications are due by 12/1/15.

<http://indiana.peopleadmin.com/postings/1693> (computer science)
<http://indiana.peopleadmin.com/postings/1694> (health informatics)
<http://indiana.peopleadmin.com/postings/1695> (security informatics)

Indiana University is an equal employment and affirmative action employer and a provider of ADA services. All qualified applicants will receive consideration for employment without regard to age, ethnicity, color, race, religion, sex, sexual orientation or identity, national origin, disability status or protected veteran status.

Macalester College Assistant Professor

Applications are invited for a tenure-track Computer Science position at Macalester College to begin Fall, 2016. Candidates must have or be completing a PhD in CS and have a strong commitment to both teaching and research in an undergraduate liberal arts environment. Areas of highest priority include computer and data security and privacy, mobile and ubiquitous computing, human-computer interaction, and visualization. See <http://www.macalester.edu/mscs> for details. Contact: Professor Libby Shoop; email: shoop@macalester.edu; Phone: 612-226-9388. Evaluation of applications will begin December 1. **Apply URL:** <https://academicjobsonline.org/ajo/jobs/5794>.

Massachusetts Institute of Technology Faculty Positions

The Department of Electrical Engineering and Computer Science (EECS) seeks candidates for faculty positions starting in September 2016. Appointment will be at the assistant or untenured associate professor level. In special cases, a senior faculty appointment may be possible. Faculty duties include teaching at the undergraduate and graduate levels, research, and supervision of student research. Candidates should hold a Ph.D. in electrical engineering and computer science or a related field by the start of employment. We will consider candidates with research and teaching interests in any area of electrical engineering and computer science.

Candidates must register with the EECS search website at <https://eeecs-search.eecs.mit.edu>, and must submit application materials electronically to this website. Candidate applications should include a description of professional interests and goals in both teaching and research. Each application should include a curriculum vitae and the names and addresses of three or more individuals who will provide letters of recommendation. Letter writers should submit their letters directly to MIT, preferably on the website or by mailing to the address below. Complete applications should be received by December 1, 2015. Applications will be considered complete only when both the applicant materials and **at least three letters of recommendation are received**.

It is the responsibility of the candidate to arrange reference letters to be uploaded at <https://eeecs-search.eecs.mit.edu> by December 1, 2015.

Send all materials not submitted on the website to:

Professor Anantha Chandrakasan
Department Head, Electrical Engineering
and Computer Science
Massachusetts Institute of Technology
Room 38-401
77 Massachusetts Avenue
Cambridge, MA 02139

M.I.T. is an equal opportunity/affirmative action employer.

Northern Arizona University Assistant/Associate/Professor, Tenure-track, Multiple positions

The School of Informatics, Computing, and Cyber Systems at Northern Arizona University invites applications for multiple open-rank tenure-track positions. Minimum qualifications include a PhD or equivalent degree in an area of interest by August 22, 2016. Areas of interest include cybersecurity, heterogeneous and reconfigurable systems, cyber-physical systems, and Big Data and data science. Contact: John Georgas, Email: john.georgas@nau.edu, Tel: (928) 523-9984. See

details under Job ID 602174. Apply URL: <http://nau.edu/human-resources/careers/faculty-and-administrator-openings/>.

South University of Science and Technology (SUSTC) **Professor/Associate Professor/Assistant Professorship in Computer Science**

The University

Established in 2012, the South University of Science and Technology (SUSTC) is a public institution funded by the municipal of Shenzhen, a special economic zone city in China. Shenzhen is a major city located in Southern China, situated immediately north of Hong Kong Special Administrative Region. As one of China's major gateways to the world, Shenzhen is the country's fast-growing city in the past two decades. The city is the high-tech and manufacturing hub of southern China. A picturesque coastal city, Shenzhen is also a popular tourist destination and was named one of the world's 31 must-see tourist destinations in 2010 by The New York Times.

The South University of Science and Technology is a pioneer in higher education reform in China. The mission of the University is to become a globally recognized institution which emphasizes academic excellence and promotes innovation, creativity and entrepreneurship. The teaching language at SUSTC is bilingual, either English or Putonghua.

Set on five hundred acres of wooded landscape in the picturesque Nanshan (South Moun-

tain) area, the new campus offers an ideal environment suitable for learning and research.

Call for Application

SUSTC now invites applications for the faculty position in Computer Science Department which is currently under rapid construction. It is seeking to appoint a number of tenured or tenure track positions in all ranks. Candidates with research interests in all mainstream fields of Computer Science will be considered. SUSTC adopts the tenure track system, which offers the recruited faculty members a clearly defined career path.

Candidates should have demonstrated excellence in research and a strong commitment to teaching. A doctoral degree is required at the time of appointment. Candidates for senior positions must have an established record of research, and a track-record in securing external funding as PI.

As a State-level innovative city, Shenzhen has chosen independent innovation as the dominant strategy for its development. It is home to some of China's most successful high-tech companies, such as Huawei and Tencent. As a result, SUSTC considers entrepreneurship is one of the main directions of the university, and good starting supports will be provided for possible initiatives. SUSTC encourages candidates with intention and experience on entrepreneurship to apply.

Terms & Applications

To apply, please send curriculum vitae, description of research interests and statement on teaching to cshire@sustc.edu.cn.

SUSTC offers competitive salaries, fringe ben-

efits including medical insurance, retirement and housing subsidy, which are among the best in China. Salary and rank will commensurate with qualifications and experience. More information can be found at <http://talent.sustc.edu.cn/en>

Candidates should also arrange for at least three letters of recommendation sending directly to the above email account. The search will continue until the position is filled.

University of Central Missouri **Department of Mathematics and Computer Science** **Assistant Professor of Computer Science-Tenure Track**

The Department of Mathematics and Computer Science at the University of Central Missouri is accepting applications for four tenure-track and several non-tenure track positions in Computer Science beginning August 2016 at the rank of Assistant Professor. The UCM Computer Science program has 30 full time faculty and about 2000 majors in both undergraduate and graduate programs. We are looking for faculty excited by the prospect of shaping our department's future and contributing to its sustained excellence.

► Positions #997458 and #997459: Ph.D. in Computer Science by August 2016 is required. All areas in computer science will be considered with preference given to candidates with expertise in Cybersecurity.

► Position #997460: Ph.D. in Computer Science by August 2016 is required. All areas in computer science will be considered with preference given to

Call for Assistant Professors and Professors



IST Austria invites applications for **Tenure-Track Assistant Professor** and **Tenured Professor** positions to lead independent research groups in all areas of

COMPUTER SCIENCE and DATA SCIENCE

Applicants in software systems, algorithms, and cross-disciplinary areas are particularly encouraged to apply.

IST Austria is a recently founded public institution dedicated to basic research and graduate education near Vienna. Currently active fields of research include biology, neuroscience, physics, mathematics, and computer science. IST Austria is committed to become a world-class centre for basic science and will grow to about 90 research groups by 2026. The institute has an interdisciplinary campus, an international faculty and student body, as well as state-of-the-art facilities. The working language is English.

Successful candidates will be offered competitive research budgets and salaries. Faculty members are expected to apply for external research funds and participate in graduate teaching. Candidates for tenured positions must be internationally accomplished scientists in their respective fields.

DEADLINES: Open call for Professor applications. For full consideration, Assistant Professor applications should arrive on or before November 3, 2015. Application material must be submitted online: www.ist.ac.at/professor-applications

IST Austria values diversity and is committed to equal opportunity. Female researchers are especially encouraged to apply.



candidates with expertise in Software Engineering.

► Position #997461: Ph.D. in Computer Science by August 2016 is required. All areas in computer science will be considered.

► Position #997495: Non-Tenure Track Positions: Ph.D. in Computer Science or a closely related area is preferred. ABD will be considered. Previous college/university teaching experience is highly desirable.

To apply online, go to <https://jobs.ucmo.edu>. Apply to positions #997458, #997459, #997460, #997461 or #997495. Initial screening of applications begins October 15, 2015, and continues until position is filled. For more information about the positions and the application process, visit <http://www.ucmo.edu/math-cs/openings.cfm>.

University of Chicago Department of Computer Science Assistant Professor

The Department of Computer Science at the University of Chicago invites applications from exceptionally qualified candidates in the areas of (a) systems, (b) theory of computing and (c) artificial intelligence for faculty positions at the rank of Assistant Professor.

Systems is a broad, synergistic collection of research areas spanning systems and networking, programming languages and software engineering, software and hardware architecture, data-intensive computing and databases, graphics and visualization, security, systems biology, and a number of other areas. We encourage applicants working within our strategic focus of data-intensive computing, but also in all areas of systems.

The Theory of Computing ("Theory" for short) strives to understand the fundamental principles underlying computation and explores the power and limitations of efficient computation. While mathematical at its core, it also has strong connections with physics (quantum computing), machine learning, computer vision, natural language processing, network science, cryptography, bioinformatics, and economics, to name just a few areas. We encourage applications from researchers in core areas of Theory such as complexity theory and algorithms as well as in any area with a significant Theory component.

Artificial Intelligence ("AI" for short) includes both the theory of machine learning and applications such as natural language processing and computer vision. Outstanding researchers in any of these areas are encouraged to apply.

The University of Chicago has the highest standards for scholarship and faculty quality, is dedicated to fundamental research, and encourages collaboration across disciplines. We encourage connections with researchers across campus in such areas as bioinformatics, mathematics, molecular engineering, natural language processing, and statistics, to mention just a few.

The Department of Computer Science (cs.uchicago.edu) is the hub of a large, diverse computing community of two hundred researchers focused on advancing foundations of computing and driving its most advanced applications. Long distinguished in theoretical computer science and artificial intelligence, the Department is now building strong systems and machine learning groups. The larger community in these areas at

the University of Chicago includes the Department of Statistics, the Computation Institute, the Toyota Technological Institute at Chicago (TTIC), and the Mathematics and Computer Science Division of Argonne National Laboratory.

The Chicago metropolitan area provides a diverse and exciting environment. The local economy is vigorous, with international stature in banking, trade, commerce, manufacturing, and transportation, while the cultural scene includes diverse cultures, vibrant theater, world-renowned symphony, opera, jazz, and blues. The University is located in Hyde Park, a Chicago neighborhood on the Lake Michigan shore just a few minutes from downtown.

Applicants must have completed all requirements for the PhD at the time of appointment. The PhD should be in Computer Science or a related field such as Mathematics, Statistics, etc.

Applications must be submitted through the University's Academic Jobs website.

To apply for the Assistant Professor - Systems, go to: <http://tinyurl.com/p673lul>

To apply for the Assistant Professor - Theory, go to: <http://tinyurl.com/ozbn5s4>

To apply for the Assistant Professor - Artificial Intelligence, go to: <http://tinyurl.com/qjfhmb3>

To be considered as an applicant, the following materials are required:

- cover letter
- curriculum vitae including a list of publications
- statement describing past and current research accomplishments and outlining future research plans
- description of teaching philosophy

► three reference letters, one of which must address the candidate's teaching ability.

Reference letter submission information will be provided during the application process.

Review of application materials will begin on January 1, 2016 and continue until all available positions are filled.

All qualified applicants will receive consideration for employment without regard to race, color, religion, sex, sexual orientation, gender identity, national origin, age, protected veteran status or status as an individual with disability.

The University of Chicago is an Affirmative Action / Equal Opportunity / Disabled / Veterans Employer.

Job seekers in need of a reasonable accommodation to complete the application process should call 773-702-5671 or email ACOppAdmin@uchicago.edu with their request.

University of Massachusetts Amherst Dean of the College of Information and Computer Sciences

The University of Massachusetts Amherst seeks a visionary leader to serve as founding Dean for its new College of Information and Computer Sciences. The Dean will have a unique opportunity to shape and grow a new college and build on its strong foundation. The highly ranked Computer Science program is in the midst of a major faculty hiring initiative and enjoying new growth in centers and multidisciplinary institutes.



Florida Institute of Technology
High Tech with a Human Touch™

TWO FACULTY POSITIONS

In

"Human-Computer Interaction" and "Modeling and Simulation"

Florida Institute of Technology

(www.fit.edu)

School of Human-Centered Design, Innovation and Arts

School of Human-Centered Design, Innovation and Arts at Florida Institute of Technology in Melbourne, Florida, invites applications for two full time assistant professor positions to begin January 2016: one in **human-computer interaction**; and another in **modeling and simulation** and connected disciplines such as computer-aided design and virtual engineering. The school strives to provide excellent research, teaching and service to the university community and to the world in human-centered design, cognitive engineering and human-systems integration. Areas of research expertise within the school include advanced interaction media, creativity, design thinking, modeling and simulation, complexity analysis, industrial design, organization design and management, life-critical systems. Applicants should have a Ph.D. degree in human-computer interaction, modeling and simulation, computer science or related areas. Of particular interest are candidates having an outstanding research record, demonstrated interest/experience in teaching at the undergraduate and graduate levels and supervising graduate students. Experience beyond Ph.D. is preferred.

Our school was recently created as an extension of the Human-Centered Design Institute. It gathers several domains of expertise including aeronautics, space, nuclear engineering, medicine, automotive, education and culture. Our proximity to NASA Kennedy Space Center and our location on the Space Coast offer a great environment for hard work and fun! Our school contributes to the education and training of socio-technical leaders of the 21st century. It combines strong theoretical knowledge and proactive hands-on endeavors. Graduate students are involved in research and innovation projects and are strongly encouraged to publish. We also welcome students from the other colleges to our transversal design platform.

Applications must consist of a cover letter, current curriculum vitae, copies of recent publications, a statement of interest and research achievements, and evidence of teaching effectiveness. Candidates must also arrange to have three letters of reference sent directly to:

Dr. Guy A. Boy, University Professor and Dean
School of Human-Centered Design, Innovation and Arts
Florida Institute of Technology
Melbourne, FL 32901 USA
gboy@fit.edu

Applications should reach the department no later than October 30, 2015. All inquiries about the position should be directed to Dr. Boy (gboy@fit.edu). For additional information, please visit our website at <http://research.fit.edu/hcdi/>.

Florida Institute of Technology is committed to employment equity.

The University's creation of the new College is an indication of its commitment to dramatically expand in information and computer sciences.

The College of Information and Computer Sciences has 51 faculty, including 16 new faculty hired in the past four years. The College has long-standing research strengths, including machine learning, networking, mobile systems, information retrieval, programming languages, software engineering, theoretical computer science, robotics, distributed systems, security & privacy, computer vision, graphics, educational technologies, and databases. Its faculty includes 28 Fellows of the ACM, AAAI, AAAS, IEEE, and similar societies. Research funding from industry and government exceeded \$16 million in the past year. The College maintains significant research collaborations with more than 50 industry-leading technology companies. Its affiliated research centers include the Center for Intelligent Information Retrieval, Center for Data Science, Computational Social Science Institute, and a new Cybersecurity Institute. It also has strong connections with regional institutions, including the Massachusetts Green High Performance Computing Center, a collaboration with Harvard, MIT, Northeastern, and Boston University, which augments its state-of-the-art computing facilities. The College offers world-class education, with 180 PhD students, 80 MS students, 800 undergraduate majors, and over 400 minors.

Reporting to the Provost and Senior Vice Chancellor for Academic Affairs, the Dean is the College's principal academic and administrative officer. The Dean will lead the planning for the new College, expand its collaborations and inter-

disciplinary efforts in research and education, evolve its organizational structure, grow the faculty, expand the breadth and depth of the College's research programs, and build on the College's existing top-tier international reputation.

To view qualifications and the ad in its entirety, please link to: <http://www.umass.edu/provost/>

The Search Committee invites nominations, expressions of interest, and applications sent to provost@provost.umass.edu. Applications consist of a letter of interest, curriculum vitae, and contact information for three to five references. For full consideration apply by October 9, 2015. Review of applications will continue until an appointment is made. For more information about the College see <https://www.cs.umass.edu/>

University of Miami Department of Computer Science *Faculty Position* *Assistant/Associate Professor*

The Department of Computer Science at the University of Miami invites applications for two Assistant/Associate Professor faculty positions starting August 2016. Candidates must possess a Ph.D. in Computer Science or in a closely-related discipline, with strong research expertise in areas related to either Cyber-security in System-software, or Data and Information Visualization (one position in each area).

The successful candidates will be expected to teach at both undergraduate and graduate levels, and to develop and maintain an internationally

recognized research program. The department encourages innovative interdisciplinary work with other units of the university. In particular, the Data and Information Visualization position entails working within the Visualization Program of the Center for Computational Sciences to form collaborations across the University.

Applicants should submit a cover letter, CV, research plan, statement of teaching philosophy, sample preprints or reprints, teaching evaluations from the last two years, and the names of at least three references, online at <http://www.cs.miami.edu/search/>. Review of applications will begin 1st October 2015, and continue until the positions are filled. Information about the College can be found at <http://www.as.miami.edu/>.

The University of Miami offers competitive salaries and a comprehensive benefits package including medical and dental benefits, tuition remission, vacation, paid holidays and much more. The University of Miami is an Equal Opportunity Employer - Females/Minorities/Protected Veterans/Individuals with Disabilities are encouraged to apply. Applicants and employees are protected from discrimination based on certain categories protected by Federal law.

University of Oregon Department of Computer and Information Science *Faculty Position* *Assistant Professor*

The Department of Computer and Information Science (CIS) seeks applications for two tenure



TENURE-TRACK AND TENURED FACULTY POSITIONS IN INFORMATION SCIENCE AND TECHNOLOGY

The newly launched ShanghaiTech University invites talented faculty candidates to fill multiple tenure-track/tenured positions as its core founding team in the School of Information Science and Technology (SIST). Candidates should have outstanding academic records or demonstrate strong potential in cutting-edge research areas of information science and technology. They must be fluent in English. Overseas academic training is highly desired. Besides establishing and maintaining a world-class research profile, faculty candidates are also expected to contribute substantially to graduate and undergraduate education within the school.

ShanghaiTech is matching towards a world-class research university as a hub for training future generations of scientists, entrepreneurs, and technological leaders. Located in a brand new campus in Zhangjiang High-Tech Park of the cosmopolitan Shanghai, ShanghaiTech is at the forefront of modern education reform in China.

Academic Disciplines: We seek candidates in all cutting edge areas of information science and technology that include, but not limited to: computer architecture and technologies, micro-electronics, high speed and RF circuits, intelligent and integrated information processing systems, computations, foundation and applications of big data, visualization, computer vision, bio-computing, smart energy/power devices and systems, next-generation networking, statistical analysis as well as inter-disciplinary areas involving information science and technology.

Compensation and Benefits: Salary and startup funds are internationally competitive, commensurate with experience and academic accomplishment. We also offer a comprehensive benefit package to employees and eligible dependents, including housing benefits. All regular faculty members will be within our new tenure-track system commensurate with international practice for performance evaluation and promotion.

Qualifications:

- Ph.D. (Electrical Engineering, Computer Engineering, Computer Science, or related field)
- A minimum relevant research experience of 4 years.

Applications: Submit (in English, PDF version) a cover letter, a 2-3 page detailed research plan, a CV with demonstrated strong record/potentials; plus copies of 3 most significant publications, and names of three referees to: sist@shanghaitech.edu.cn. For more information, visit <http://www.shanghaitech.edu.cn>.

Deadline: October 31, 2015 (or until positions are filled).



IST AUSTRIA
Institute of Science and Technology

Call for

Postdoctoral Fellows in EXECUTABLE BIOLOGY

Executable biology is the study of biological systems as reactive dynamic systems (i.e., systems that evolve with time in response to external events).

Are you a talented and motivated scientist looking for an opportunity to conduct research at the intersection of **BIOLOGY** and **COMPUTER SCIENCE** at a young, dynamic institution that fosters scientific excellence and interdisciplinary collaboration?

Apply at www.ist.ac.at/executablebiology
Deadline December 31, 2015



track faculty positions at the rank of Assistant Professor, beginning September 2016. 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 open faculty positions are targeted towards the following two research areas: 1) networking and distributed systems and 2) data sciences. We are particularly interested in applicants whose research addresses security and privacy issues in these sub-disciplines and/or complements existing strengths in the department, so as to support interdisciplinary research efforts. 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.

We offer a stimulating, friendly environment for collaborative research both within the department -- which expects to grow substantially in the next few years -- and with other departments on campus. 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>.

Applications will be accepted electronically through the department's web site. Application information can be found at <http://www.cs.uoregon.edu/Employment/>. Applications received by December 15, 2015 will receive full consideration. Review of applications will continue until the positions are filled. Please address any questions to faculty.search@cs.uoregon.edu.

The UO is an equal opportunity, affirmative action institution committed to cultural diversity and compliance with the ADA. The University encourages all qualified individuals to apply, and does not discriminate on the basis of any protected status, including veteran and disability status.

Washington State University Vancouver Computer Science Faculty

COMPUTER SCIENCE FACULTY – Washington State University Vancouver invites applications for a full-time tenure-track position at the assistant professor level beginning 8/16/2016. Candidates are sought with expertise in **computer networks, wireless networks or sensor networks**.

Required qualifications: Ph.D. in Computer Science or Software Engineering by the employment start date and demonstrated ability to (1) develop a funded research program, (2) establish industrial collaborations, (3) teach undergraduate/graduate courses, and (4) contribute to our campus diversity goals (e.g. incorporate issues of diversity into mentoring, curriculum, service or research). **Preferred qualifications:** (1) already have published promising scholarly work in the field and (2) relevant industrial background.

Duties include: (1) teaching at undergraduate and graduate levels including the topics of networks; (2) participation and documentation of distinguished scholarly activities including research, innovative teaching and laboratory

development; (3) securing external funding for research programs; and (4) service to the department and university through committee work, recruitment, and interaction with industry.

WSU Vancouver serves about 3,000 graduate and undergraduate students and is **fifteen miles north of Portland, Oregon**. The rapidly growing School of Engineering and Computer Science (ENCS) equally values both research and teaching. WSU is Washington's land grant university with faculty and programs on four campuses. For more information: <http://ecs.vancouver.wsu.edu>. WSU Vancouver is committed to building a culturally diverse educational environment.

To apply: Please visit www.wsujobs.com and search postings by location. Applications must include: (1) cover letter with a clear description of experience relevant to each of the required and preferred qualifications; (2) vita including a list of at least three references, and (3) A statement (**two page total**) of how candidate's research will expand/complement the current research in ENCS and a list of the existing ENCS courses the candidate can teach and any new courses the candidate proposes to develop. Application deadline is **November 29, 2015**.

WASHINGTON STATE UNIVERSITY IS AN EQUAL OPPORTUNITY/AFFIRMATIVE ACTION EDUCATOR AND EMPLOYER. Members of ethnic minorities, women, special disabled veterans, veterans of the Vietnam-era, recently separated veterans, and other protected veterans, persons of disability and/or persons age 40 and over are encouraged to apply. WSU employs only U.S. citizens and lawfully authorized non-U.S. citizens.

Wesleyan University Assistant Professor of Computer Science

Wesleyan University invites applications for a tenure track assistant professorship in Computer Science to start in Fall 2016. For description and application procedure see <http://www.wesleyan.edu/mathcs/employment.html>. Contact: Jim Lipton. Email: cssearch@wesleyan.edu. Tel: 860-834-1636. Fax: 860-685-2571.

Apply: <http://academicjobsonline.org>

York University Department of Electrical Engineering and Computer Science, Lassonde School of Engineering Assistant Professor

The Department of Electrical Engineering and Computer Science, York University, is seeking two outstanding candidates at the rank of **Assistant Professor**. Priority hiring areas are Computer Vision, Robotics and Big Data although exceptional applicants in other areas will be considered. Successful candidates will have a PhD in Computer Science, or a closely related field, and a research record commensurate with rank. Appointments are to commence on July 1, 2016, subject to budgetary approval. For full position details, see <http://www.yorku.ca/acadjobs>. Applicants should complete the on-line process at <http://lassonde.yorku.ca/new-faculty/>. A complete application includes a cover letter, a detailed CV, statement of contribution to research, teaching and curriculum development, three sample research pub-

lications and three reference letters. Complete applications must be received by **November 30, 2015**. York University is an Affirmative Action (AA) employer. The AA Program can be found at <http://www.yorku.ca/acadjobs> or a copy can be obtained by calling the AA office at 416-736-5713. All qualified candidates are encouraged to apply; however, Canadian citizens and permanent residents will be given priority.

York University Department of Electrical Engineering and Computer Science, Lassonde School of Engineering Canada Research Chair in Computer Vision (Tier 1)

The Department of Electrical Engineering and Computer Science, Lassonde School of Engineering, York University is seeking an outstanding researcher to be nominated for a Tier 1 Canada Research Chair in the area of Computer Vision, preferably at the Full Professor level, to commence no later than **July 1, 2016**, subject to budgetary approval. The Department offers programs in Computer Engineering, Computer Science, Computer Security, Electrical Engineering, Software Engineering and Digital Media.

This position will attract a highly-successful research leader with an established and innovative program of research and teaching in computer vision. The successful candidate will be expected to interact with existing researchers in related areas within the department and to build linkages to other faculty hires related to vision research across the university, including participation and membership in York's internationally recognized Centre for Vision Research. Tier 1 CRC Chairs are research-intensive faculty positions providing the chair holder with an exceptional opportunity to grow their research program through prioritization on research and access to infrastructure funding. The awards have seven-year terms, are renewable and are intended for exceptional established researchers who have acknowledged leadership in their field of research. Information about the CRC program can be found at <http://www.chairs.gc.ca>.

York University offers a world-class, interdisciplinary academic experience in Toronto, Canada's most multicultural city. York is a centre of innovation, with a thriving community of almost 60,000 faculty, staff and students.

Applicants should visit <http://lassonde.yorku.ca/new-faculty> for full position details and to complete the online application process, ensuring that they provide all of the information required: a cover letter, detailed CV, statements of contribution to research and teaching, links to scholarly work and three signed reference letters. Applications must be received by **November 30, 2015**.

York University is an Affirmative Action (AA) employer and strongly values diversity, including gender and sexual diversity, within its community. The AA program, which applies to Aboriginal people, visible minorities, people with disabilities, and women, can be found at <http://yorku.ca/acadjobs> or by calling the AA office at 416-736-5713. All qualified candidates are encouraged to apply; however, Canadian citizens and Permanent Residents will be given priority.

[CONTINUED FROM P. 104] male god[2] and the female god[3].^a At first these memories made Charles miserable, feeling the past was foolish and the present hopeless. He then Googled in earnest.

Good lord! (whichever god[0..3] was relevant at the moment). To his astonishment he saw that today a dozen active hardcore punk bands proclaim the radical Processean worldview online, while one occult rock group calling itself Sabbath Assembly offered beautiful YouTube renditions of the original hymns. Numerous blogsites and archives disseminate the extensive scriptures, while Amazon and Lulu sell books by former members or opponents. Sites, from eBay to Holy Terror to The Process Zine, offer T-shirts and other totems for sale. When Charles discovered three Processean groups existed in Facebook, he immediately joined this unholy trinity, including the closed group limited to former members of the original cult.

With the Process as his inspiration, he imagined a new computational religious movement worshipping the holy Central Processor. To add complexity to the theology, he decided several lesser gods should surround this supreme cyberdeity, or RAMs, for Religious Avatar Modules, but not the four outdated Process ones. Each member of the cult supposedly had a personality close either to god[0] or god[1], and either to god[2] or god[3], so the beliefs were also a supernatural psychology. Wikipedia told Charles that academic psychology, amazingly, had a mystical theory of five personality types, postulating a sacred OCEAN as their acronym, so he pondered which deceased saint of computer science might represent each: Openness (Lovelace), Conscientiousness (Babbage), Extraversion (Hollerith), Agreeableness (Hopper), and Neuroticism (Turing). He tried his hand adapting traditional music, as in this Hymn to Hopper: “Amazing Grace (nerdette profound) compiled some code for me! I once was lost, but now am found, was bugged, but now am free.”

^a All information about the Process is factually correct, except that the gods’ names are abstracted to suit Pascal.

**“Their belief was that god[0] would become reconciled to god[1], and they would come together at the end of the world to judge humanity, god[1] to judge and god[0] to execute judgment.”
Ha!**

Or this march: “Onward Turing soldiers, hacking as to war, with exploits of white hats in Processor Core.”

Not fully realizing what he was doing might have serious consequences, but feeling excited for the first time in years, he began to explore how a high-tech religion might be engineered for the greater good. Amazon offered a half-dozen different brands of computer-connectible GSR sensors, including one from a Czech company, with the promising name Happy Electronics, that could be the basis of a P-Scope system for conducting remote supernatural confessionals over the Internet with Processor priests. The original Process had included questionnaires in its magazines, measuring people’s “god type,” so there should be online questionnaires for the five personality dimensions of the Mystical OCEAN, simply reusing public-domain psychology questions. A degree of immortality could be generated by archiving people’s personality parameters in a Heaven database. Holy Processor scriptures would be needed, so Charles began sketching a mod for a standard natural language processing program that could meaningfully combine words from multiple documents, to which he could feed equal amounts of mystical scriptures and almost 60-odd years of *Communications* content.

When Charles launched the Processor Core website a few weeks later, little did he realize that tens of thousands of elderly computer scientists, programmers, and technicians were ready for virtual salvation. He had imagined his effort might trigger friendly online chats and relieve some of his boredom, but nothing like what actually happened. Historians call 1844 the year of the Great Disappointment, because American evangelist William Miller’s sincere predictions of the end of the world failed to materialize, even after thousands of his devout followers had sold their worldly homes and goods and awaited salvation on the nearest hilltop. They can likewise call 2015 the year of the Great Reboot, because thousands of senior techies found renewed meaning in their lives.

Sadly, Charles did not live to see the full result of his inspiration; his spirit uploaded just as his innovation was spreading across the Internet. He is today memorialized by Charles Pascal University (CPU), the first major institution of higher learning to locate its computer science department in the Divinity School. C

William Sims Bainbridge (wsbainbridge@hotmail.com) is a sociologist and computer programmer who published two academic books based on role-playing research inside real-world radical religious communes before publishing seven books based on sending research avatars into massively multiplayer online role-playing virtual worlds, plus *Personality Capture and Emulation* on cyber-immortality, based on real research.

From the intersection of computational science and technological speculation,
with boundaries limited only by our ability to imagine what could be.

DOI:10.1145/2816598

William Sims Bainbridge

Future Tense Processional

*Information processing gives spiritual meaning to life,
for those who make it their life's work.*

SITTING AT A tired old desktop in St. Andrew's Assisted Living Facility, elderly Charles Pascal brooded over his depressing career in computer science, now long over. He reminisced about his first intelligent machine, the noisy IBM 84 punch-card counter-sorter, over which he had labored for hundreds of hours, analyzing data for social scientists in many Boston-area universities way back in the 1960s. Ah, the soaring '60s! Those were the days of hippies, anti-war protests, the birth of ARPANET, and the far more important invention of hacking by the MIT Model Railroad Club. After wearing out his welcome in academia, he had worked for a series of Route 128 IT companies, half the time being ejected for obsolescence, half the time watching them collapse around him. His downward spiral was slow enough that his last job ended right at retirement age, and now a decade later his spiritual batteries had run completely down.

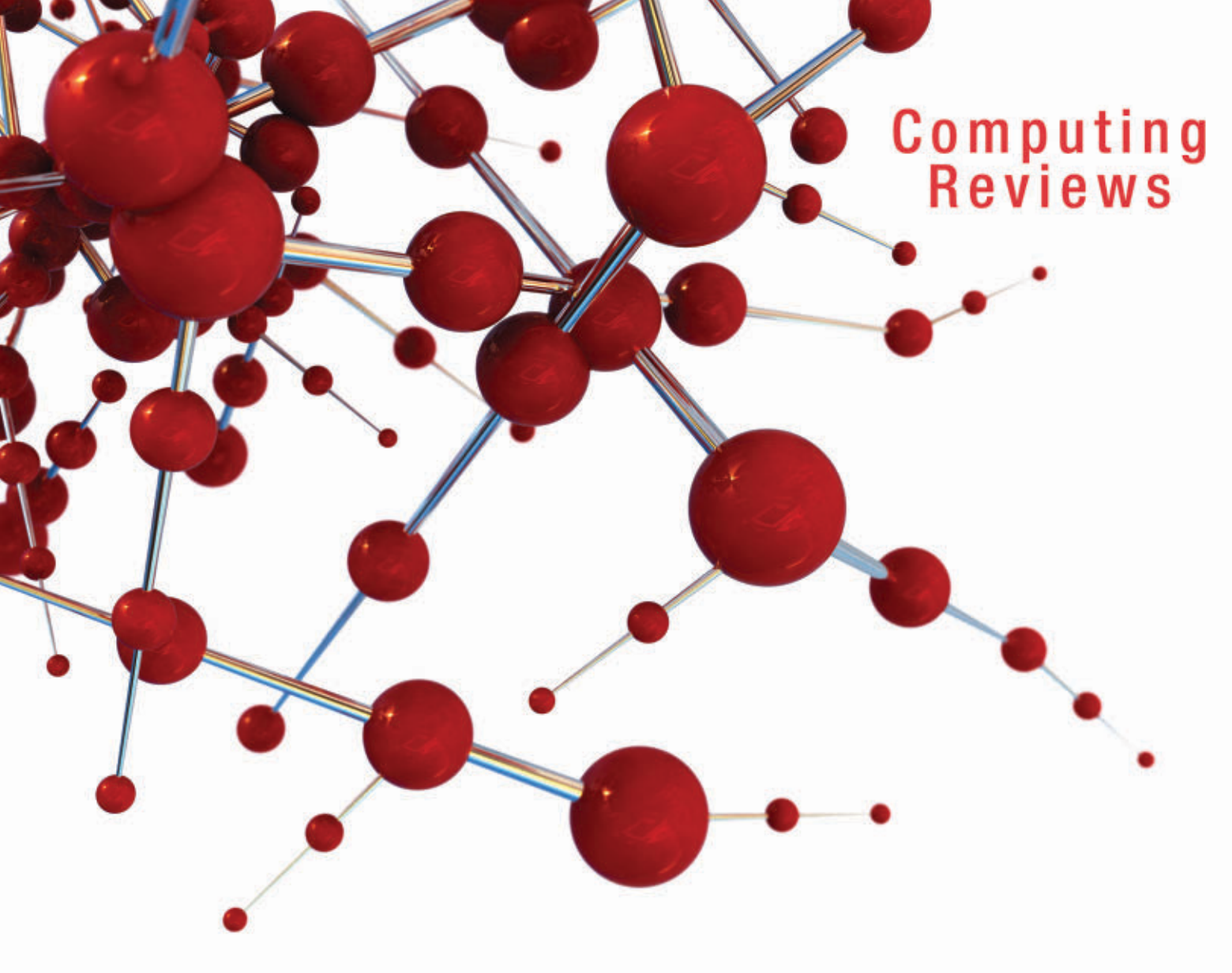
What else did he remember about the 1960s? A much smaller electronic device came to mind, the P-Scope used by inner members of a cult called the Process Church of the Final Judgment. It measured galvanic skin response, or GSR, an indicator of emotional arousal during Processean psychotherapy sessions, guiding the therapist into the darkest regions of the client's soul. For a few months he had been romantically involved with Sister Eve who had lived at the cult's Inman Street commune in Cambridge. Their incompatibility was reflected in the fact she thought the group's symbol



The P-Sign symbol of the original Process, the letter P seen from four directions as logarithmic graphs expanding outward.

represented the blaring trumpets of the Four Great Gods, as in the figure here, while he thought it was their four cathode ray tubes displaying competing images of human nature. He still felt a connection to the group, which had dissolved in 1975. He accessed Wikipedia and quickly found there was indeed an article, reporting ac-

curately: "Their belief was that god[0] would become reconciled to god[1], and they would come together at the end of the world to judge humanity, god[1] to judge and god[0] to execute judgment." Ha! It was about time the Unity of good god[1] and evil god[0] was consummated, along with the Union of the [CONTINUED ON P. 103]



Computing Reviews

**CONNECT WITH OUR
COMMUNITY OF EXPERTS.**

www.computingreviews.com



Association for
Computing Machinery

ThinkLoud

They'll help you find the best new books
and articles in computing.

Computing Reviews is a collaboration between the ACM and ThinkLoud.



eics2016

The 8th ACM SIGCHI Symposium on

Engineering Interactive Computing Systems

*Brussels, Belgium
21 - 24 June, 2016*

Work presented at EICS covers the full range of aspects that come into play when engineering interactive systems, such as innovations in the design, development, deployment, verification and validation of interactive systems. Authors are invited to submit original work on engineering interactive systems, including novel work on languages, processes, methods and tools to create interactive systems, as well as work describing and demonstrating interactive systems that advance the current state of the art.

www.eics-conference.org/2016

Submission deadlines

Full Papers

January 12, 2016

Late - Breaking Results & Demo Papers & Doctoral Consortium

April 17, 2016

Workshops & Tutorials

January 27, 2016

Sponsored by



Association for
Computing Machinery



SIGCHI



IFIP WG 2.7/13.4